TESIS DOCTORAL

Análisis bioinformático del transcriptoma de pino

Noé Fernández Pozo

Universidad de Málaga
Departamento de Biología Molecular y Bioquímica
Facultad de Ciencias
Plataforma Andaluza de Bioinformática
Edificio de Bioinnovación

Málaga. Mayo de 2012

AUTOR: Noé Fernández Pozo

http://orcid.org/0000-0002-6489-5566

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

Análisis bioinformático del transcriptoma de pino

Memoria presentada por:

Noé Fernández Pozo

Para optar al grado de Doctor en Ciencias Biológicas por la Universidad de Málaga

Tesis realizada bajo la dirección del Dr. M. Gonzalo Claros Díaz en la Plataforma Andaluza de Bioinformática y el Departamento de Biología Molecular y Bioquímica de la Universidad de Málaga

Fdo. Noé Fernández Pozo

$V^o.B^o$. DIRECTOR DE LA TESIS DOCTORAL:

Fdo. M. Gonzalo Claros Díaz

Málaga, 28 de Mayo de 2012

D. M. Gonzalo Claros Díaz, Investigador de la Plataforma Andaluza de Bioinformática y el Departamento de Biología Molecular y Bioquímica de la Universidad de Málaga.

CERTIFICA:

Que Don Noé Fernández Pozo, Licenciado en Biología, ha realizado bajo mi dirección en el Departamento de Biología Molecular y Bioquímica y la Plataforma Andaluza de Bioinformática de la Universidad de Málaga, el trabajo de investigación recogido en la presente memoria de Tesis Doctoral que lleva por título: "Análisis bioinformático del transcriptoma de pino".

Tras la revisión de la presente Memoria se ha estimado oportuna su presentación ante la Comisión de Evaluación correspondiente, por lo que autorizo su exposición y defensa para optar al grado de Doctor en Biología.

Y para que así conste, en cumplimiento de las disposiciones legales vigentes, firmo el presente certificado.

Málaga, 28 de Mayo de 2012

El Director de la Tesis,

Dr. D. M. Gonzalo Claros Díaz

Proyectos

Este trabajo de tesis se enmarca bajo los siguientes proyectos de investigación:

- Estudios de genómica funcional en plantas de interés forestal (2006-2009; Proyecto de Excelencia de la Junta de Andalucía, AGR-663). IP: Francisco M. Cánovas.
- Biología de Sistemas: de la Genómica a la Biocomputación (2007-2010; Plan Propio de la UMA). IP: Enrique Viguera Mínguez.
- Participación en la primera fase de la iniciativa internacional para la secuenciación del genoma de pino. (2008; MEC BIO2007-29814-E). IP: Francisco Cánovas.
- Genomic tools in maritime PINE for enhanced biomass production and SUSTAINable forest Management (SUSTAINPINE). (2010-2013; MICINN & FP7-PLANT-KBBE Scientific Advisory Board PLE2009-0016). IP: Francisco Cánovas.
- Ingeniería biomolecular de la madera de pino mediante aproximaciones de genómica functional (2010-2012; MICINN, AGL2009-12139-C02-02). IP: Francisco R. Cantón.
- Desarrollo de herramientas bioinformáticas para los estudios genómicos y transcriptómicos a partir de datos de secuenciación de lecturas cortas de alto rendimiento para las especies que no tienen un organismo modelo de referencia (NEOGEN) (2011-2013; Proyecto de Excelencia de la Junta de Andalucía, CVI-6075). IP: M. G. Claros.

Artículos

Los siguientes artículos justifican este trabajo de tesis:

- Fernández-Pozo et al., 2011. **EuroPineDB: a high-coverage web database for maritime pine transcriptome.** BMC Genomics, 12: 366.
- Fernández-Pozo et al., 2011. **Programming languages for bioinformatics.** Encuentros en la Biología. Vol. 4, núm. 134, 31-32.
- Fernández-Pozo et al., 2012. **GeNOTE: a web tool for annotation of non-model, eukaryotic, unfinished sequences**. En A.T. Freitas and A. Navarro (Eds.), *Bioinformatics for Personalized Medicine*, series *Lecture Notes in Computer Science*, Vol. 6620, pp. 66-71. Springer-Verlag, Berlín-Heidelberg.
- Sanz-Santos et al., 2011. **Gene expression pattern in swine neutrophils after lipopolysaccharide exposure: a time course comparison.** BMC Proceedings 2011, 5(Suppl 4):S11.
- Falgueras et al., 2010 **SeqTrim: a high-throughput pipeline for preprocessing any type of sequence read.** BMC Bioinformatics 11: 38

Agradecimientos

En primer lugar quiero agradecerle al Dr. M. Gonzalo Claros, el director de este trabajo, todo lo que ha hecho por mí durante todos estos años. Siempre recordaré, que fue él quien me dio la oportunidad de empezar esta aventura en la investigación científica, y sobre todo ahora, me alegro de que me animara a tomar la senda de la bioinformática, un campo que mira hacia el presente y futuro de la biología. Quiero agradecerle también, la paciencia y el apoyo que me ha ofrecido siempre, y especialmente durante la escritura de este manuscrito.

Al Dr. Francisco M. Cánovas, gracias por acogerme dentro del grupo y encontrar financiación en estos momentos tan difíciles. Nunca olvidaré esos partidos de baloncesto intra- e interdepartamentales.

A la Dra. Concepción Ávila, el Dr. Francisco Ruiz Cantón, y de nuevo, el Dr. Francisco M. Cánovas, muchas gracias por contar conmigo cuando se requería de la bioinformática, y gracias por acogerme en el grupo.

A los compañeros del PTA, Rocío, Darío, Rafa, Hicham, Pedro, muchas gracias por todo, siempre recordaré las charlas de los desayunos, en especial el hecho de que Rafa es un experto en todos los temas. Gracias a todos por estar ahí siempre que os he necesitado, gran parte de lo que soy como bioinformático es gracias a vosotros. A Dario gracias por todo lo que me has enseñado, gracias por tu paciencia y por compartir tu sabiduría. Gracias a Juan Falgueras por su ayuda con Perl.

A los compañeros del grupo Nandi, Canales, Pepi, Vanesa, Blanca, Cañas, Marco y todos los demás gracias por acogerme y espero que todo os vaya muy bien. A Sara, David, Said, gracias por los ratos que pasamos juntos, siempre recordare aquellos días que nos íbamos al bosque a doblar pinos y tomar muestras. A Vicky, Juan Jesus, Eva, Juanjo, espero que todo os vaya muy bien, siempre recordaré esas tardes de baloncesto.

A mis compañeros de la carrera, que siempre me han apoyado, Alejandro, Benji, Lorena, Laura, Eli, Iván, Jose Antonio, Salvi, Manolo, Paco, Sergio, y especialmente a Salvador García Bravo por realizar la ilustración de la portada, gracias por todo.

A mis amigos no biólogos, Alfonso, Gustavo, Juan, Paco, Manu, Carmen, Serralvo, Edu, Rosa, Gon, Rocío, Pepe, Bego, Miguel y Fernando, por los momentos que hemos compartido y por creer en mí, muchas gracias.

A Pili, mi mujer y mejor amiga, por quererme y aguantarme todos estos años y en especial estos últimos meses de escritura, por todo lo que haces por mí, muchas gracias.

A mi familia, a mi abuela Luisa, a mis hermanas Séfora y Blanca, y en especial a mis padres, Salvador y Victoria, que sin ellos nada de esto hubiese sido posible, gracias por apoyarme y creer en mí.

A todos vosotros, espero que todo os vaya muy bien en lo personal y en lo profesional, y que la salud os respete para que disfrutéis de la vida durante muchos años, seguid así, con gente como vosotros el mundo es un sitio mejor.

A mi abuela Victoria
1922-2012

Acrónimos y Abreviaturas

Aa aminoácido

ADN ácido desoxirribonucleico

ADNc copia en ácido desoxirribonucleico

ADNg acido desoxirribonucleico genómico

ADNr acido desoxirribonucleico ribosómico

API interfaz de programación de aplicaciones (del inglés *Application Programming Interface*)

ARNc copia en ácido ribonucleico

ARNm ácido ribonucleico mensajero

ARNr ácido ribonucleico ribosómico

BAC cromosoma artificial bacteriano

BLAST *Basic Local Alignment Search Tool*

BMBP Biología Molecular y Biotecnología de Plantas

CPU unidad central de procesamiento

DDBJ Banco de Datos de ADN de Japón

ddNTP didesoxinucleótidos trifosfato

EBI European Bioinformatics Institute

EC Enzyme Commission

EMBL Laboratorio Europeo de Biología Molecular

EST etiquetas de secuencias expresadas (del inglés *Expressed Sequence Tag*)

FC veces de cambio

FDR tasa de positivos falsos (del inglés *false discovery rate*)

FTP protocolo de transferencia de archivos (del inglés *File Transfer Protocol*)

FWER *familywise error rate*

GED genes expresados diferencialmente

GO Gene Ontology

indels inserciones y deleciones

MID identificadores multiplexados

NCBI National Center for Biotechnology Information

NGS secuenciación de nueva generación (del inglés *next generation sequencing*)

nt nucleótido

OLC *overlap-layout-consensus*

ORF marco abierto de lectura

PAB Plataforma Andaluza de Bioinformática

pb pares de bases

Poli-A poliadenosinas

Poli-T politimidinas

ROR Ruby-on-Rails

SCBI Centro de Supercomputación y Bioinformática de la UMA

SNP polimorfismos mononucleotídicos

SSH hibridación sustractiva por supresión

SSR repeticiones de secuencias simples

WGA secuenciación de todo el genoma (del inglés *whole genome sequencing*)

Resumen y contextualización

Las especies del género *Pinus* son de gran interés económico y ecológico en todo el mundo, y especialmente en el hemisferio norte. Sin embargo estas especies carecen de un genoma secuenciado debido a la complejidad y al gran tamaño de su genoma, repleto de repeticiones y elementos transponibles, y por ser mucho mayor que el de otros organismos modelo como el ser humano. Por estos motivos, la primera aproximación para el estudio de estos organismos, es el análisis de sus genes a través del transcriptoma, algo que en estos momentos, con las nuevas técnicas de secuenciación, es mucho más accesible. El transcriptoma del pino, en especial de *Pinus pinaster*, se estudia en este trabajo mediante el análisis de la expresión de sus genes en diferentes condiciones y tejidos, a través de micromatrices de ADNc y de la secuenciación y caracterización de estos genes. Para ello, se desarrollan una serie de herramientas bioinformáticas útiles para el análisis de micromatrices de ADNc, y para el preprocesamiento y anotación de secuencias, adaptadas tanto a las tecnologías de nueva generación como a las anteriores. También se propone un flujo de preprocesamiento, ensamblaje y anotación de transcriptomas para especies no modelo que carecen de un genoma de referencia, en el que se incluyen estas herramientas y se demuestran sus beneficios a la hora de obtener el transcriptoma del pino. Para poner la información de las micromatrices y de las secuencias a disposición de la comunidad científica se ha desarrollado una base de datos que ha sido de gran utilidad para diseñar una micromatriz de más de 8000 puntos. El funcionamiento de esta base de datos se ha ido mejorando y simplificando hasta proponer un modelo para transcriptómica de rápido desarrollo, en la que hemos incluido la última versión del transcriptoma de *Pinus pinaster*, obtenida con las herramientas desarrolladas en este trabajo y el flujo propuesto. En la última versión del transcriptoma de pino hemos agrupado los genes con anotaciones y genes específicos de la especie, para obtener un transcriptoma fiable de unos 30 000 genes.

Este trabajo de doctorado se ha realizado en el grupo de investigación Biología Molecular y Biotecnología de Plantas (IBMBP) del catedrático Francisco M. Cánovas Ramos, bajo la dirección del profesor M. Gonzalo Claros Díaz. El objetivo final era abrir una nueva línea de investigación bioinformática para el grupo de investigación, principalmente encaminado hacia la profundización en el conocimiento del transcriptoma y del genoma de *Pinus pinaster*, así como establecer las herramientas bioinformáticas que resultarán más útiles para dichos estudios. Al tratarse de un trabajo pionero para el grupo, en la presente memoria se incluyen algunos ejemplos de código de programación para ayudar a la consolidación de esta línea de trabajo.

En gran parte, este trabajo ha sido posible gracias al personal y la infraestructura de la Plataforma Andaluza de Bioinformática (PAB), donde es posible acceder a recursos informáticos sin los que parte del trabajo hubiera sido de difícil resolución, además de aprender técnicas de programación, que han permitido que los programas desarrollados en este trabajo puedan ser ejecutados en paralelo. Además, a través de la PAB surgieron otras colaboraciones mencionadas en este trabajo, entre las que cabe destacar por mi participación, el desarrollo de una base de datos de lenguado, desarrollada en colaboración con el Instituto de Investigación y Formación Agraria y Pesquera (IFAPA), y el análisis de micromatrices realizado en colaboración con el grupo de investigación de Genómica y Mejora Animal del Departamento de Genética de la Facultad de Veterinaria de la Universidad de Córdoba.

Índice general

Parte I

Introducción

Capítulo 1

El pino, un modelo de leñosa

El género *Pinus* se encuentra dentro de las coníferas (orden *Coniferales* o *Pinales*, según la clasificación), donde están algunos de los organismos más longevos del planeta, con especies como *Pinus longaeva* y *Taxus baccata* en las que algunos individuos alcanzan los 5000 años, y *Picea abies* con individuos de más de 8000 años en algunas poblaciones de Suecia. A su vez, las coníferas se engloban dentro de las gimnospermas, junto con las cicas, la división *Gnetophyta* y la especie *Ginkgo biloba*. Dentro de estos grupos, *Pinus pinaster* y *Pinus sylvestris* son dos de las principales especies modelo en Europa. En la figura 1.1 puede observarse la distribución de *Pinus pinaster*, que es el tipo de pino de mayor interés en el sur de Europa.

1.1. Interés económico y ecológico del pino

Los bosques constituyen aproximadamente un 82 % de la biomasa terrestre, y son una fuente de materia prima para una gran cantidad de productos esenciales para el ser humano, como materiales de construcción, papel, leña y carbón para calentarse, cocinar y producir energía, además de alimento como los piñones. Los bosques además aportan varios servicios ecológicos, entre los que cabe destacar los hábitats que permiten conservar la biodiversidad, el reciclaje del CO_2 y la regulación del clima. Sin embargo, durante los últimos años el cambio climático y la deforestación son dos de los temas medioambientales de mayor preocupación [1]. Se estimó que durante la última década se perdieron unas 13 hectáreas de bosques por año en el mundo, lo que repercute directamente en ciclo del carbono y en la pérdida de biodiversidad [71]. Por ello es muy importante que se aumente el conocimiento científico sobre estas especies, para poder realizar una explotación sostenible de sus recursos y frenar así el calentamiento global y la deforestación.

Los pinos se encuentran principalmente en el hemisferio norte [127], donde son muy valorados por su madera, porque se utiliza para la construcción de edificios y muebles, postes de teléfono, vías de tren y mástiles de embarcaciones entre otras cosas. Los pinos se utilizan a menudo en las reforestaciones por su rápido crecimiento en comparación con otras especies arbóreas, porque forman grandes bosques que sirven de hábitat para muchas especies y son capaces de crecer en zonas arenosas, donde se utilizan para retener el suelo o crear sustrato. Además, se utilizan derivados de su resina para la producción de papel, adhesivos, tintas de impresión, compuestos de goma, revestimientos superficiales, barnices, pinturas, esmaltes, preparación de materiales de limpieza, controles biológicos para luchar contra plagas de invertebrados y otros fitopatógenos, construcciones, embarcaciones y precursores de fármacos.

En el género *Pinus* hay algunos rasgos de interés económico y ecológico en los que la herencia de los genes depende de un solo gen (es decir, es una herencia de tipo mendeliana). A modo de ejemplo, valgan los genes más importantes de resistencia a la roya vesicular del pino blanco (*Cronartium ribicola*). Sin embargo, la gran mayoría de ellos son rasgos complejos y cuantitativos. Estos incluyen el crecimiento, las propiedades de la madera y la resistencia a enfermedades, insectos y estrés abiótico [163].

1.2. El genoma del pino

Las gimnospermas se encuentran entre las plantas con los genomas más complejos y grandes de todos los organismos [159], y el pino no es una excepción, como se muestra en la figura 1.2, donde se compara el gran tamaño del genoma del pino con otras plantas, como *Arabidopsis thaliana* y la caña de azúcar, *Saccharum officinarum*. El genoma de *Pinus pinaster* es más de 200 veces más grande que el de *Arabidopsis thaliana*, especie utilizada como modelo para el estudio de las plantas. Este excepcional tamaño no se explica mediante fenóne-

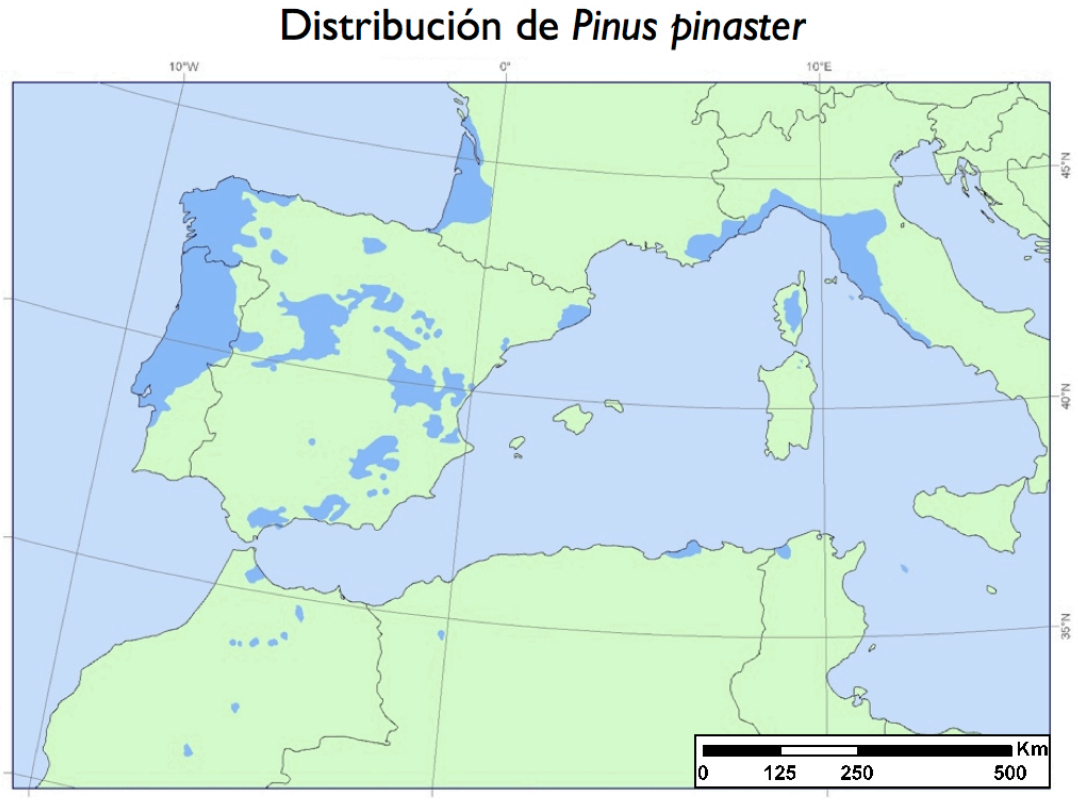


Figura 1.1: *Mapa de distribución del pino marítimo (*Pinus pinaster*) en Europa. Figura tomada de EUFORGEN 2009,* `www.euforgen.org`

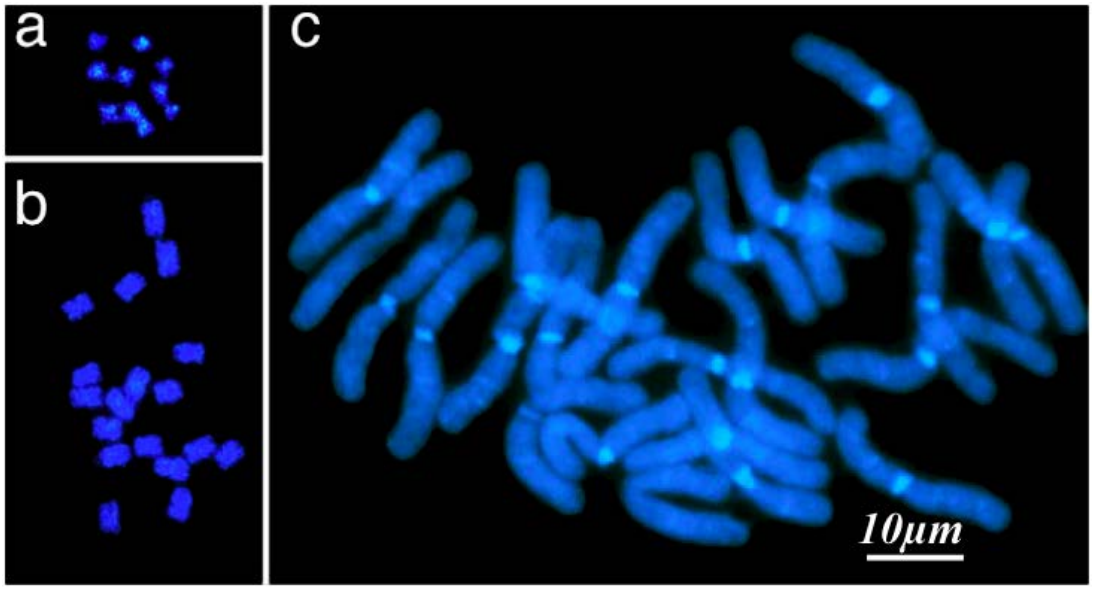


Figura 1.2: *Comparación del tamaño de los cromosomas de varias especies de plantas. (a) Arabidopsis (genoma haploide, 140 Mbp de ADN aproximadamente; 2n = 10); (b) caña de azucar (750 Mbp; 2n = 18); y (c) pino (30 300 Mbp en Pinus pinaster; 2n = 24). Figura tomada de [195]*

mos de poliploidía, que sí son muy frecuentes entre las angiospermas [171, 48, 92], porque son raros en las coníferas [163]. Todos los miembros del género *Pinus* son diploides y tienen 12 parejas de cromosomas (2n = 24) de tamaño parecido [127]. En el caso de *Pinus pinaster*, su tamaño se estima en 30 300 Mbp aproximadamente (`http://dendrome.ucdavis.edu/NealeLab/crsp/overview.php`),

unas diez veces más grande que el genoma humano (3 000 Mpb aproximadamente). Sin embargo, poco se sabe del número de genes que contiene, que en otras plantas nunca superan los 40 000 [211] y en el ser humano está entre 20 000 y 25 000 [52]. El incremento del número de genes en las plantas con respecto a otros organismos, puede deberse su estilo de vida y a las constantes duplicaciones del genoma que se vienen constatando [80, 54, 55, 145]. Las plantas están fijas al sustrato y no pueden huir de las amenazas que les afectan, por lo que han ido desarrollando diferentes estrategias para enfrentarse a herbívoros, mamíferos, insectos o patógenos (virus, bacterias y hongos), a las variaciones climáticas, a las especies vecinas con las que compiten y a otras formas de estrés [211]. Así pues, aunque el genoma de pino seguramente tendrá más genes que el genoma humano, es evidente que dado que su genoma es mucho más grande, la mayor parte del ADN no será codificante.

En el ADN no codificante de las plantas abundan los retrotransposones LTR (con repeticiones terminales largas) [159, 80]. Se ha descrito que en las angiospermas con genomas haploides de mas de 2000 Mb, más del 50 % del ADN nuclear está compuesto por retrotransposones LTR y otras repeticiones [24], llegando a constituir más del 70 % del genoma nuclear en el maíz [25]. La gran cantidad de repeticiones que aparecen en el ADN no codificante van a afectar directamente a los procesos de ensamblaje cuando se pretenda secuenciar las especies vegetales. No es de extrañar que una de las teorías propuestas para explicar el motivo del gran tamaño del genoma del pino sea la acumulación de gran cantidad de elementos transponibles, de unidades de repeticiones de ADNr, de familias de genes y de pseudogenes [79]. Los elementos móviles podrían contribuir a la inversión, pérdida o fusión de genes en el genoma. Algunas teorías indican que los periodos de mayor actividad de los retrotransposones han afectado a la evolución de los genomas de las plantas actuales. Esta expansión puede verse ligada a periodos de estrés, considerándose así como un modo de variación que sometido a la selección natural puede actuar como método de diversificación para formar nuevas especies [159].

En un estudio realizado sobre ADN genómico de *Pinus taeda* repartido en 10 BAC que representaban casi 1 Mb [127] se detectaron 3 genes codificantes y 15 pseudogenes, además de una gran cantidad y diversidad de elementos repetitivos, desde áreas ricas en AT y GC a repeticiones complejas como retrotransposones LTR y de otros tipos, transposones de ADN, retrovirus endógenos y otros elementos repetitivos [127]. Los análisis realizados en nuestro laboratorio sobre secuencias genómicas en BAC sugieren además que los genes podrían estar más aislados que en otras especies vegetales y que abundan los elementos repetitivos y pseudogenes, pero que el tamaño de los genes no parece ser muy distinto al de otras plantas, con la salvedad de que hay genes en los que algún intrón puede llegar a tener varias kilobases (datos no publicados).

En el caso de los pseudogenes, suelen retener cierta similitud con los genes de los que provienen, pero no son funcionales debido a la acumulación de mutaciones. Esto provoca codones de parada prematuros, cambios de fase o alteraciones que afectan a las regiones reguladoras o a los sitios de ayuste. Estas mutaciones generalmente ocasionan la pérdida de la capacidad de transcribir o traducir al gen correspondiente, o provocan la síntesis de proteínas afuncionales [79]. En el pino también aparecen pseudogenes con mucha frecuencia [127, 79], hecho que se ha demostrado también en nuestro laboratorio [21].

En conclusión, el gran tamaño del genoma, la gran cantidad de secuencias repetitivas, y la complejidad de los rasgos de interés económico y ecológico hacen que el estudio del genoma de los pinos merezca la pena abordarse primero desde el punto de vista transcriptómico, y posteriormente pasar a una estrategia genómica.

1.3. Hacia la genómica de pino

En la actualidad existe un común acuerdo de que el conocimiento de las secuencias de los genomas de plantas ayudarán a [111]:

1. Descifrar el conjunto de genes característicos de las plantas.

2. Identificar genes de importancia agronómica.

3. Encontrar polimorfismos mononucleotídicos (SNP) y otros marcadores moleculares que ayuden a la mejora genética.

4. Conocer las variaciones específicas de diferentes linajes y su disposición en el genoma.

5. Comprender la evolución del genoma y su diversificación para conocer la forma en la que los diferentes linajes han logrado adaptarse a los eventos de duplicación de todo el genoma.

6. Conocer cómo se relaciona el genoma con la variación morfológica y fisiológica (la variación que ha contribuido a la colonización de gran parte del planeta por las plantas superiores).

La mayoría de los proyectos de secuenciación de plantas se están realizando con angiospermas, principalmente dentro del grupo de las eucotiledóneas, donde se encuentran las plantas de cultivo más importantes desde un punto de vista económico. Se suelen utilizar de modelo especies como *Arabidopsis thaliana*, maíz, arroz, trigo y tomate. Sin embargo, el conjunto de todos los grupos dedicados a la investigación en árboles forestales está formado por no más de unas mil personas en todo el mundo que estudian decenas de especies [163], a las que además se suelen dedicar presupuestos de investigación de menor cuantía que a las angiospermas. Afortunadamente, en los últimos años se ha empezado a invertir más en los proyectos destinados a la secuenciación de genomas de coníferas. Prueba de ello es que en 2010, la fundación Knut y Alice Wallenberg aportó 10 millones de dólares para secuenciar el genoma de *Picea abies*. Un proyecto que se está llevando a cabo por grupos de investigación de Suecia principalmente, en colaboración con algunos grupos de Canada, Bélgica e Italia, y en el que esperan tener un borrador del genoma de *Picea abies* para 2013 [219]. Otra gran inyección de dinero se llevó a cabo en 2011 por el departamento de agricultura de los Estados Unidos (USDA), realizando una inversión de 14,6 millones de dólares para secuenciar los genomas de tres coníferas, *Pinus taeda*, *Pinus lambertiana*, y *Pseudotsuga menziesii*. Por su parte, la Unión Europea ha comenzado también a financiar proyectos relacionados con *Pinus pinaster* y *Pinus sylvestris* (`http://cordis.europa.eu/fp7/kbbe/`).

Por otro lado, para conocer los genes del pino basándose en la similitud de su secuencia se han invertido esfuerzos en el desarrollo de un gran número de EST de pino y organismos cercanos [11, 33, 121, 141]. Por ejemplo, si se realiza una búsqueda de las secuencias del género *Pinus* en la base de datos de EST del NCBI, aparecen 473 298 EST de pino, su mayoría de *Pinus taeda* (328 662), seguido de *Pinus pinaster* (34 649). La misma búsqueda para EST del género *Picea* encuentra 571 751 secuencias, de las cuales, 200 000 fueron aportadas por un solo proyecto [181]. Como se verá en este trabajo, estas secuencias son particularmente útiles para la comparación y validación de los ensambles del transcriptoma de pino, así como para la creación de modelos de genes para coníferas [144]. Aunque estas secuencias sirven de guía para confirmar que los genes encontrados también se expresan en dicho transcriptoma, aún queda mucho por hacer, ya que lo habitual es que las EST no estén anotadas y se desconozca el producto génico que codifican. Por eso, para encontrar secuencias con anotaciones revisadas, suele recurrirse a búsquedas en organismos más alejados desde un punto de vista evolutivo, como *Arabidopsis thaliana*, Maíz (*Zea mays*) o arroz (*Oryza sativa*).

Los pinos se separaron de las angiopermas hace 200-300 millones de años, y esta distancia hace que no se obtengan buenos resultados cuando se usan especies distantes como el chopo, la vid o el arroz como genoma de referencia [127]. La ausencia de un genoma de referencia y las otras características del genoma de pino mencionadas en el apartado 1.2 hace que los estudios genómicos de estas especies sean caros y laboriosos. Por eso, y como alternativa, el análisis a gran escala de ADNc en forma de EST constituye una estrategia muy utilizada para conocer los genes y sus secuencias. Debido a la mencionada complejidad de los genomas de coníferas, su estudio se ha basado principalmente en el análisis de su transcriptoma [144]. Además, las especies no modelo sin un genoma de referencia adecuado requieren diferentes estrategias para conocer el genoma, como por ejemplo las técnicas de nueva generación de secuenciación, que gracias a la reducción de costes que conllevan están abriendo la puerta a la caracterización de un gran numero de especies no modelo, tanto en las plantas como en animales y otros organismos [201].

Capítulo 2

Análisis de la expresión génica con micromatrices

2.1. Visión general

Antes de que los secuenciadores de nueva generación revolucionaran la forma de analizar la expresión génica, las micromatrices (*microarrays*) habían sido otra técnica que había cambiado la forma de analizar la expresión de los genes en diferentes condiciones experimentales [146]. Las micromatrices permiten la comparación masiva de miles de genes en una, dos o más situaciones diferentes mediante plataformas tecnológicas diferentes. Hay varios tipos de micromatrices, de proteínas-anticuerpos [94], para conocer los patrones de expresión de las proteínas y su función en relación a procesos biológicos, micromatrices de tejidos [125], que permite la comparación de la expresión de proteínas, ARN o ADN en muestras de diferentes tejidos, micromatrices de ADN genómico [200], para comparar por ejemplo dos genómas y observar que regiones se hay de más o de menos [76], o el número de copias de algún gen de interés [176], micromatrices de SNP [152], para detectar mutaciones o polimorfismos, y micromatrices de expresión [114]. En este trabajo, de todos los tipo de micromatrices mencionados anteriormente, solo se han utilizado micromatrices de expresión, por lo que este capítulo se centrará únicamente en este tipo de micromatrices.

En los experimentos de expresión con micromatrices, el fundamento principal consiste en cuantificar la hibridación de las sondas a sus dianas:

- Las **sondas** consisten en una muestra genérica lo más completa posible del transcriptoma del organismo u organismos que se desea estudiar. Están unidas covalentemente a un sustrato, que puede ser un microchip o un portaobjetos de microscopio (*slide*) que tiene un tratamiento especial que permite su fijación.

- Las **dianas** son las diferentes muestras que se quieren comparar y que están marcadas (con fluorescencia o radiactividad) para poder detectar el nivel de hibridación a las sondas. Cada muestra diana contendrá tan solo un subconjunto de los genes del transcriptoma.

El objetivo final del uso de micromatrices para la expresión génica consiste en averiguar qué genes modifican sus patrones de expresión en las diferentes condiciones de las muestras (dianas), siendo el nivel de expresión directamente proporcional a la cantidad de diana que ha quedado hibridada con la sonda.

Una posibilidad son las micromatrices de ADNc que permiten comparar a la vez la expresión de los genes en dos situaciones diferentes; se obtienen expresiones relativas, no absolutas. La alternativa son las micromatrices de oligonucleótidos de alta densidad, como por ejemplo las desarrolladas por Affymetrix, en las que se analiza la expresión de los genes en una sola condición experimental, que además permiten la comparación de todas las muestras entre sí, con lo que se obtienen datos de expresión absoluta y datos de expresión relativa.

Las micromatrices de expresión han producido una gran cantidad de información acerca de cómo se comportan los transcriptomas en diferentes tipos celulares [115] y tejidos [41], de cómo cambia la expresión de los genes en las diferentes etapas del ciclo celular de un organismo [209], y entre individuos sanos y enfermos [8, 85] [146]. Esta técnica ha resultado muy útil en las plantas, donde se han estudiado desde qué genes se expresan diferencialmente entre sequía y salinidad [199] hasta los específicos de la defensa de la planta [194, 227].

La llegada de las nuevas técnicas de secuenciación está relegando este tipo de experimentos a un segundo plano. Pero hay que reconocer que gracias a la aplicación de las micromatrices en las especies forestales como el pino se ha profundizado más en el conocimiento [163], por ejemplo, de la formación de la madera [235], de la pared celular [230], de la

defensa contra insectos [182], del desarrollo de la raíz [31], de las variaciones debidas a las estaciones [234], de la variación entre especies y tejidos [220], o de cómo responden las raíces frente a diferentes tipos de hongos [3], como por ejemplo *Laccaria bicolor* [97]. Es más, en el campo de las coníferas se han desarrollado varias micromatrices de gran tamaño: con 26 496 puntos de secuencias de *Pinus taeda* [142], con 21 840 puntos de secuencias de varias especies de *Picea* [99] y con 11 394 puntos de secuencias de *Pinus pinaster* (micromatriz comercial del Instituto Austriaco de Tecnología; `http://www.picme.at/index.php/products/arrays`).

A continuación se describen las dos plataformas de micromatrices más populares, las de alta densidad de oligonucleótidos y las de ADNc. Los estudios que han comparado ambas tecnologías muestran que ambas plataformas, con sus ventajas e inconvenientes, son igualmente fiables en cuanto a la calidad de los datos que obtienen y la precisión con la que miden las variaciones en la expresión [172, 173].

2.2. Micromatrices de ADNc

También se denominan «micromatrices de dos colores». Sirven para comparar simultáneamente dos conjuntos de secuencias, cada uno marcado con un fluoróforo diferente, que suelen ser el Cy3 y el Cy5 (ambos derivados de la cianina), que emiten a 570 nm y 670 nm respectivamente, de ahí el nombre de micromatrices dos colores. Por lo general se asigna un color a cada uno de estos fluoróforos: verde para Cy3 (canal G, del inglés *green*), y rojo para Cy5 (canal R, del inglés *red*), a pesar de que no sean esos los colores a los que realmente emiten los fluoróforos.

Al realizar la hibridación, las dianas de un mismo gen de cada muestra compiten por unirse a la sonda que contiene la secuencia complementaria del gen. De este modo, los genes que se expresen más en una muestra que en otra se hibridarán con la sonda en mayor cantidad, y el punto de la micromatriz en el que está impresa esa sonda mostrará una mayor señal a la intensidad de onda del fluoróforo con el que fue marcado. Esta luz emitida por cada fluoróforo se escanea y con un programa informático se generará una imagen de la micromatriz hibridada, en la que los puntos con mayor intensidad a 570 nm se mostrarán en verde, los de 670 nm en rojo, aparecerán en amarillo cuando la cantidad de ambos fluoróforos es aproximadamente la misma, o aparecerán en negro cuando no se expresa en ninguna de las dos condiciones analizadas (véase la imagen central de la figura 2.1a).

La gran ventaja de estas micromatrices de dos colores es que se pueden realizar dos hibridaciones a la vez, con lo que se elimina la variabilidad técnica debida a las diferencias entre las hibridaciones, aunque como contrapartida introduce una variabilidad técnica debida a que la incorporación de ambos fluoróforos puede ser diferente, y a que el escáner puede no detectar con la misma fiabilidad ambas longitudes de onda [122].

Preparación de la micromatriz: las sondas se obtienen por amplificación por PCR de los insertos procedentes de genotecas de ADNc, o bien mediante síntesis química de oligonucleótidos, como por ejemplo en la tecnología de Agilent. Las sondas se imprimen en sitios específicos de la superficie de cristal de un portaobjetos de microscopio utilizando robots de alta precisión. Las hebras codificantes del ADNc se unen a la superficie del cristal mediante un enlace covalente, gracias a entrecruzadores químicos (figura 2.1a).

Preparación de las muestras: como diana se utiliza ADNc monocatenario formado a partir del ARN procedente de dos condiciones experimentales diferentes. Se marcan con nucleótidos conjugados a fluoróforos diferentes, uno diferente para cada condición experimental en cada hibridación. Ambas muestras se mezclan y se hibridan de modo competitivo con las secuencias fijadas en la micromatriz (figura 2.1a). La hibridación se escanea después a dos longitudes de onda diferentes, las correspondientes a cada fluoróforo, y se obtienen la intensidad y la relación de la abundancia de cada sonda impresa en las dos condiciones experimentales [198].

2.3. Micromatrices con alta densidad de oligonucleótidos

La plataforma más conocida de este tipo son las micromatrices desarrolladas por Affymetrix. Es un producto comercial, en el que se imprimen unos 300 000 oligonucleótidos en un microchip y se hibridan con las secuencias que se quieren estudiar. Se utiliza un conjunto de secuencias como control y posteriormente se hibridan tantos grupos de secuencias como se desea estudiar. Las condiciones experimentales pueden ser luego comparadas entre sí o a través de la muestra de control. En las micromatrices de oligonucleótidos, cada gen está representado normalmente por 16-20 parejas de sondas. Uno de los componentes de estas parejas se conoce como *perfect match* (PM) y el otro como *mismatch* (MM). Cada sonda PM tiene junto a ella una MM, la PM

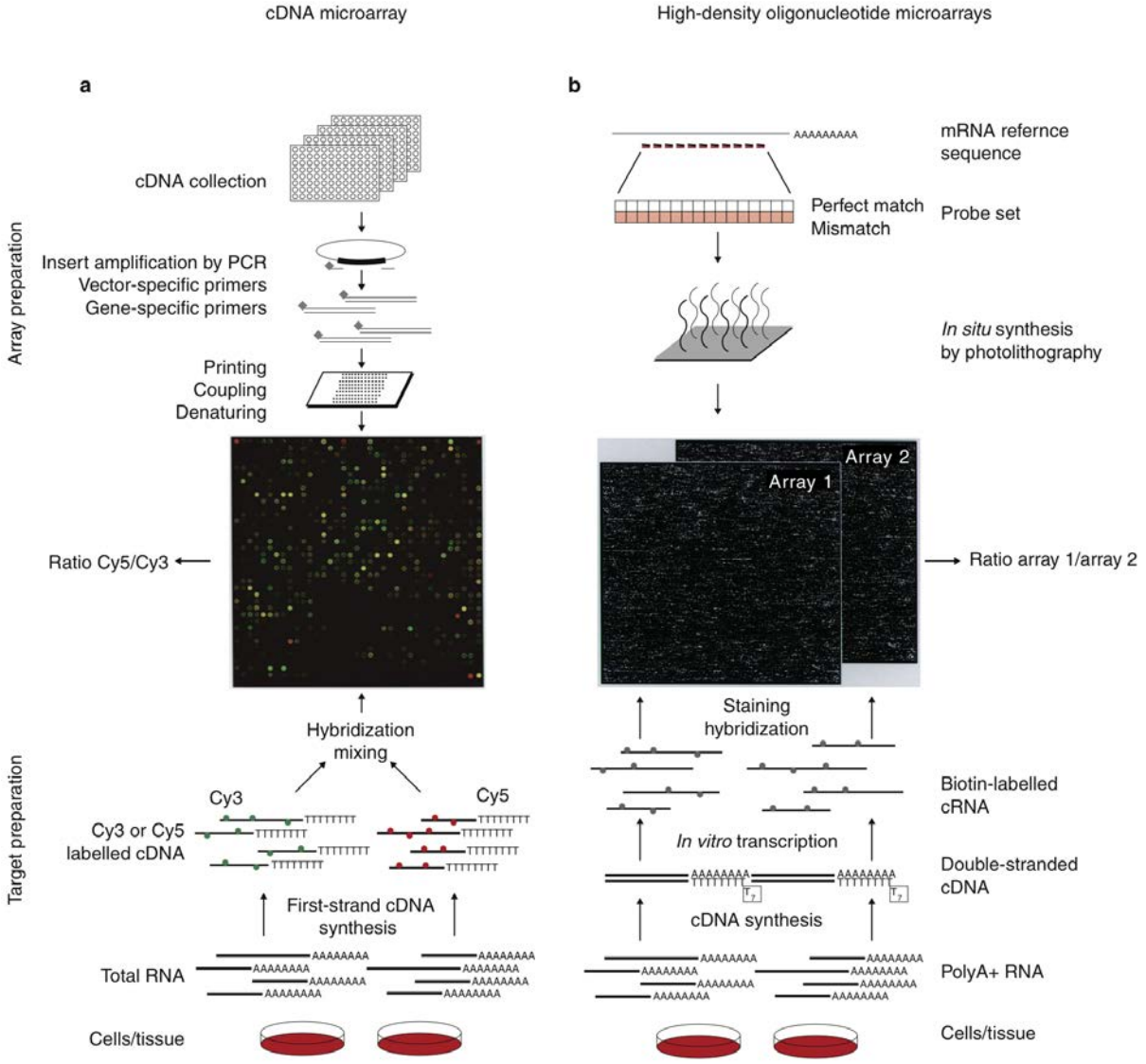


Figura 2.1: *Preparación de las sondas y de las dianas (muestras) para* **(a) Micromatrices de ADNc** *y,* **(b) Micromatrices con alta densidad de oligonucleótidos**. *Imagen tomada de [198]*

es idéntica a la región de la secuencia que se quiere hibridar, y la MM suele contener un nucleótido diferente en su parte intermedia, usualmente en la posición 13. Estas sondas MM, están diseñadas con la intención de cuantificar las uniones inespecíficas [109]. Sin embargo, posteriores estudios [228, 109], han determinado que la señal de las sondas MM no representa de un modo preciso las hibridaciones inespecíficas, sino que puede ser una fuente de errores en el análisis, y proponen otros métodos para el análisis de micromatrices de oligonucleótidos como RMA (*robust multi-array average*) [109], que utiliza las sondas PM únicamente, lo que conlleva una menor varianza y potencialmente a la detección de un mayor número de genes candidatos [228].

Preparación de la micromatriz: las sondas son oligonucleótidos cortos, generalmente de unas 25 pb, de modo que cada gen de la micromatriz queda representado por unos 16-20 oligonucleótidos diferentes. La síntesis de oligonucleótidos se realiza *in situ*, de manera que se genera una matriz de alta densidad que puede llegar a contener 300 000 sondas (figura 2.1b).

Preparación de las muestras: la diana será ARNc preparado a partir de la primera cadena de ADNc que se obtiene del ARNm. Durante la síntesis *in vitro* del ARNc se incorporan nucleótidos marcados con biotina. La hibridación de cada muestra (figura 2.1b) se realiza por separado y el fluoróforo se adhiere a la diana al ir unido a estreptavidina. La

intensidad de la señal recogida por el escáner sirve para calcular la cantidad relativa de cada uno de los genes de la micromatriz [198].

2.4. Variabilidad y réplicas

Como en este trabajo los resultados que se presentan son únicamente de micromatrices de dos colores, solo se describirán los aspectos relacionados con los experimentos que se realicen con esta plataforma, algunos de los cuales también serán útiles para las micromatrices de oligonucleótidos.

Las micromatrices no siempre constituyen la estrategia experimental adecuada: Hay que tener en cuenta que las hibridaciones de las micromatrices no tienen por qué resolver cualquier problema biológico. Las micromatrices sirven para conocer los patrones de expresión del conjunto total de genes del experimento, no para conocer la respuesta específica de unos pocos genes concretos, para lo cual hay técnicas más apropiadas como la PCR en tiempo real (RT-PCR) [122].

No vale con una simple hibridación: Por más caras que resulten, las hibridaciones de las micromatrices han de realizarse varias veces, como cualquier otro experimento. Las repeticiones serán de dos tipos: técnicas y las biológicas. Las réplicas técnicas dan precisión al experimento puesto que sirven para evaluar la significación estadística, dar mayor precisión numérica a los resultados, y para corregir la variabilidad —natural y sistemática— del ensayo. Así se podrá estimar y reducir todo aquello que introduzca variabilidad por la obtención de los datos, por la preparación de las muestras, por la hibridación en el laboratorio [10], o por la incorporación de los fluoróforos. Hay que tener en cuenta que son muchas las fuentes de variación técnica y que la suma de las pequeñas variaciones que se dan en cada caso puede repercutir en los valores de expresión finales. De hecho, se puede introducir variabilidad en todas las etapas del proceso:

- El ARNm obtenido puede presentar pequeñas variaciones debido a las condiciones del material biológico, a las diferencias entre los individuos o incluso dentro del mismo individuo, al método de extracción, a la eficacia de la retrotranscripción, al rendimiento de la PCR, y al marcaje con los fluoróforos.
- Durante la impresión de las micromatrices pueden producirse variaciones por anomalías en las puntas, por la impresión de diferente cantidad de ADN en cada vez (incluso con la misma punta), por el lote de portaobjetos, por la longitud de los fragmentos fijados, por el rendimiento de la unión química del ADN al soporte, o incluso por la humedad ambiental.
- La hibridación puede presentar variabilidad debido a la diferencia de sensibilidad entre los fluoróforos, a una distribución desigual de la diana sobre la micromatriz, al lavado, a la temperatura del experimento y del entorno, al experimentador, e incluso al momento del día.
- Incluso la adquisición de los datos puede incluir variabilidad debida al propio escáner, al decaimiento desigual de los fluoróforos, a la ganancia del fotomultiplicador y a la manera en que el programa localiza las sondas hibridadas.

Cada diana hay que marcarla al menos una vez con cada fluoróforo: Uno de los sesgos que se puede introducir inadvertidamente en el experimento se debe a la eficacia del marcaje y lectura de los fluoróforos. Por eso conviene realizar al menos una réplica técnica en la que cada muestra (diana) esté marcada con el fluoróforo contrario al que se ha utilizado la vez anterior, es decir, si un tratamiento se marca con Cy3 y su control se marca con Cy5, se realizará una réplica técnica de intercambio de fluoróforos de manera que ahora el tratamiento se marcará con Cy5 y el control con Cy3.

Las mejores réplicas son las biológicas: Si se invierte demasiado en las réplicas técnicas se puede estar seguro que los valores obtenidos son correctos y libres de sesgos técnicos. Sin embargo, no se puede decir que los genes que muestren expresión diferencial se deban a las diferencias entre el tratamiento y el control, o se deban a las muestras concretas que se tomaron. La pregunta «¿Obtendré los mismos genes si repito el experimento con otras muestras?» solo puede resolverse con réplicas biológicas, porque sirven para evaluar la variabilidad del sistema biológico y controlar la variación al azar al preparar la muestra de ARN. Además, utilizar distintos individuos (uno para cada réplica) disminuye el sesgo que introduce el genotipo de cada individuo. Los estudios realizados indican que para que la búsqueda de genes expresados diferencialmente sea biológicamente significativa, se deberá de disponer de al menos 5 grados de libertad, esto es, que la suma de muestras biológicas diferentes menos las condiciones utilizadas debe ser mayor de 5 [10]. Por ejemplo, si se quiere analizar un tratamiento frente a un control (2 condiciones experimentales) se necesitarán al menos 8 muestras biológicas para que $8-2>5$. Se consideran réplicas biológicas los ARN

obtenidos de diferentes extracciones de muestras diferentes, a ser posible y si la estrategia experimental lo permite, de individuos diferentes. Una buena máxima a seguir pasa por invertir el máximo esfuerzo y dinero en las réplicas biológicas y no en las técnicas.

Agrupamiento de muestras, una forma de promediar sin estadística: Otro modo de considerar la variabilidad biológica consiste en el agrupamiento de muestras biológicas (*pools*), de manera que en una única hibridación ya se promedian varias muestras biológicas. Esto reduce los costes del experimento y resulta beneficioso cuando el objetivo principal es identificar la expresión diferencial de los genes, cuando la variabilidad biológica es alta con relación a errores de medición, y sobre todo cuando las muestras biológicas son difíciles de obtener [119]. Sin embargo, la realización de agrupamientos de ARNm no es siempre ventajosa, ya que se pierde la información de la variación entre individuos [10].

El diseño experimental es crucial: Con un buen diseño experimental se podrá distinguir la mayor cantidad posible de genes que se expresan diferencialmente, sin que se vean afectados por los que muestran variación debido a la acumulación de errores técnicos (preparación de muestras, marcaje e hibridación). El diseño también afectará seriamente a la eficiencia y a la potencia estadística de los análisis que hay que realizar posteriormente cuando se analizan los datos de la hibridación. También se verá afectada la eficacia para estimar y corregir los errores potenciales debidos al marcaje con los fluoróforos [122].

2.5. Análisis de micromatrices

Los resultados obtenidos tras escanear la intensidad de emisión de los fluoróforos de las dianas que se hibridaron a cada una de las sondas impresas en la micromatriz, quedan recogidos en ficheros de texto tabulado organizados en columnas de datos. La manera de analizar estos datos variará según la plataforma utilizada, puesto que cada una los proporciona en distintos formatos. Como en este trabajo los resultados que se presentan son únicamente de micromatrices de expresión de dos colores, este apartado se centrará únicamente en cómo se realiza para este tipo de datos.

2.5.1. Representaciones gráficas

Un sistema de dos fluoróforos produce dos valores de intensidad por punto, uno por cada canal (R y G), y cada intensidad se correlacionará con la cantidad de ARNm producida por el gen. Para relacionar los valores de intensidad de ambos canales se suelen utilizar varios tipos de representaciones (figuras 2.2a y 2.2b). Una de las más usadas son las *gráficas MA* (figura 2.2a), donde se representan los valores de M frente a los valores de A. M es el logaritmo en base 2 de la relación de intensidades de los canales rojo (R) y verde (G), como $M = log_2(R/G)$, y puede considerarse como el logaritmo del valor de expresión relativa de los genes. A indica la intensidad de la señal, tal que $A = log_2\sqrt{(R \times G)}$. Con estas variables se consigue que los valores de sobreexpresión ($M > 0$) y subexpresión ($M < 0$) sean simétricos: una sobreexpresión del cuádruple es $M = 2$ y una sobreexpresión a la cuarta parte es $M = -2$, en lugar de 4 y 0,25, respectivamente. En las gráficas MA (figura 2.2a), los valores de M quedan en el eje y, de modo que los puntos con valores de $M > 0$, serán aquellos que muestren niveles de expresión mayores en el canal R que en el G, y del modo contrario, los puntos con mayor expresión en el canal G quedarán representados por debajo del valor de $M = 0$. En el eje x se representan los valores de A, que indican la intensidad de la señal, situándose los valores de menor intensidad a la izquierda, y los de mayor intensidad a la derecha de la gráfica MA.

También se suelen representar los valores de intensidad de cada micromatriz en *representaciones por cajas* (figura 2.2b1) para comparar la homogeneidad de los valores de la intensidad de la señal o del ruido de fondo entre las réplicas. Estas representaciones por cajas son también útiles para comparar los valores de M entre las muestras (figura 2.2b2). La homogeneidad se pone de manifiesto cuando las diferentes cajas tienen tamaños similares y están centradas en torno a $M = 0$.

Otras representaciones habituales en el análisis de micromatrices son las *gráficas en volcán* y los *mapas térmicos* (figura 2.2c y 2.2d, respectivamente) que son útiles para poner de manifiesto los *g*enes que se *e*xpresen *d*iferencialmente (GED) en los experimentos de micromatrices. Las gráficas en volcán separan la nube de puntos situando los genes subexpresados a la izquierda, y los genes sobreexpresados a la derecha. Además, permiten localizar el valor de P con el que se hace el corte. Todos los puntos que aparezcan por encima del valor de corte de P ajustado muestran una expresión diferencial estadísticamente significativa. Los mapas térmicos agrupan los GED según sus patrones de variación y lo ponen de manifiesto con gradación de color desde la subexpresión a la sobreexpresión. Algunos de los métodos de agrupación más comunes son las distancias euclídeas [103], aunque también hay otros disponibles

como las correlaciones de Pearson y Spearman, y las *k-means*

2.5.2. Herramientas bioinformáticas

Han sido muchas las herramientas desarrolladas para el análisis de micromatrices, tanto de acceso libre como comerciales, cada una de ellas con sus puntos fuertes y sus debilidades. Las que contienen los algoritmos punteros para el análisis de micromatrices pertenecen al grupo de herramientas de acceso libre, y algunas de ellas son TM4, GEPAS y Bioconductor (el paquete de R) [149]:

- TM4 está formada por cuatro aplicaciones principales, MADAM (Microarray Data Manager), TIGR_Spotfinder, MIDAS (Microarray Data Analysis System), y Mev (Multiexperiment Viewer) [103], y dispone de una interfaz gráfica para facilitar a los usuarios su manejo, además de una base de datos para guardar los experimentos [189].
- GEPAS (`http://gepas.bioinfo.cipf.es/`), es una herramienta web que permite el uso de los paquetes de R para el análisis de micromatrices sin necesidad de tener conocimientos del lenguaje o de programación [157] .
- Bioconductor [82] es una colección de librerías de R con funciones estadísticas muy potentes para el análisis de micromatrices y la generación de gráficos. La realización de los análisis de micromatrices utilizando Bioconductor directamente en la consola de R requiere de ciertas habilidades de programación, pero permite la mayor flexibilidad y potencia para ajustar el análisis a las necesidades del experimento, algo que queda limitado cuando se utilizan programas como TM4 o GEPAS.

El análisis de los datos de micromatrices de dos colores se divide en las siguientes etapas esenciales: corrección del fondo, normalización, y detección de la expresión diferencial. Puesto que en este trabajo se utilizaron para ello exclusivamente las funciones de Bioconductor, todas las explicaciones se limitarán a cómo se hace utilizando este paquete.

2.5.3. Corrección del fondo

Siempre conviene observar el aspecto de los datos sin modificar porque indicará si la señal y el ruido de fondo son homogéneos, o si muestran algún tipo de patrón que sugiera la existencia de problemas durante el lavado o la hibridación, o durante al impresión de las sondas. Después hay que corregir la intensidad de cada sonda en función del ruido de fondo. Se ha comprobado que esta corrección no mejora la detección de genes expresados diferencialmente, pero si no se corrigiera, se subestimarían la amplitud del cambio de la expresión génica [179]. Existen distintos métodos para corregirla. El más extendido es la simple resta, pero tiene el enorme problema de devolver valores negativos a intensidades bajas. Se ha demostrado que la resta no es un método óptimo, sino que el método `normexp` (de *normal-exponential*, y es una adaptación del método `RMA` desarrollado para chips de Affymetrix) implementado en Bioconductor [82] es el más recomendable [184, 179]. `Normexp` compensa el ruido de fondo sin que aparezcan valores negativos, lo que disminuye la variabilidad a intensidades bajas (figura 2.3).

2.5.4. Normalización

Los datos con el fondo corregido tienen que ser normalizados para corregir las variaciones sistemáticas (técnicas) que se mencionaron antes (apartado 2.4) y las diferencias de distribución de las intensidades, de modo que los datos normalizados sean comparables dentro de la micromatriz y entre micromatrices. En principio, los cambios que se obser-

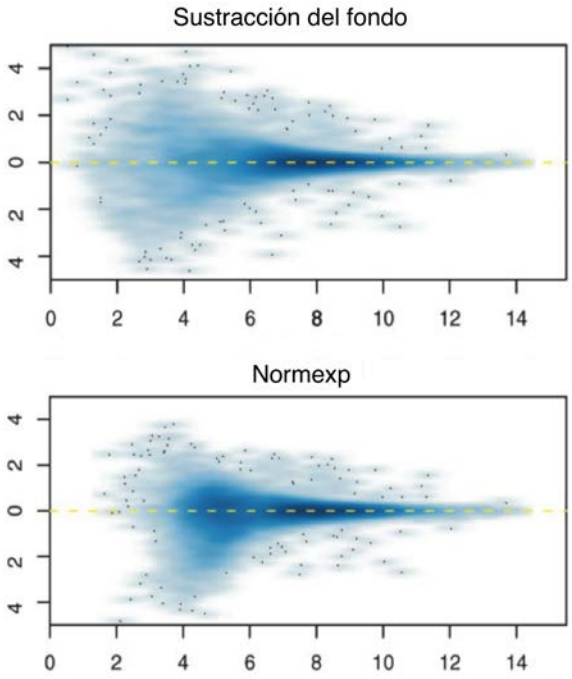


Figura 2.3: *Gráficas MA tras corregir el fondo mediante sustracción y mediante* `normexp`*. Nótese las diferencias en la nube de puntos a intensidades bajas (a la izquierda del eje x). Imágenes tomadas de [184].*

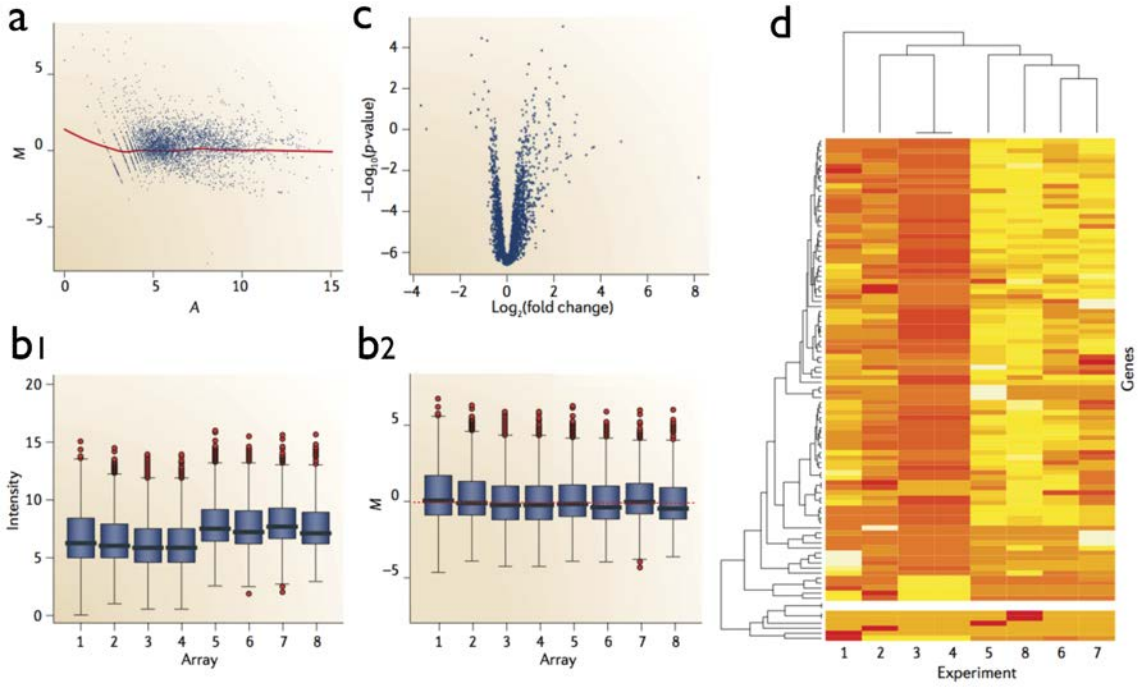


Figura 2.2: *Imágenes típicas para representar los datos de micromatrices. a) gráfica MA, b1) representación de la intensidad por cajas, b2) representación de M por cajas, c) gráfica en volcán, d) Mapa térmico (*heat map*). Imagen reconstruida a partir de una figura de [10]*

ven sólo se deberían a la variación biológica [45], aunque hay que tener en cuenta que toda manipulación matemática, como la normalización, altera los datos de expresión diferencial, puesto que es capaz de introducir nuevos artefactos o sesgos en ellos [45]. Así pues, es deseable que los datos iniciales sean lo más homogéneos y repetitivos posible para que la posterior normalización no introduzca cambios drásticos en estos valores que haga peder información biológica. Por ejemplo, una micromatriz sesgada al verde o al rojo será menos informativa antes y después de la normalización que una que esté centrada en el amarillo.

Para poder normalizar los datos hay que suponer que la cantidad de ARNm con que se hibrida la matriz es la misma en todas las muestras, que la cantidad de genes sobreexpresados es prácticamente idéntica a la de los subexpresados, y que ambas cantidades sean mucho menores (menos del 10 %) que la de los genes invariantes. En consecuencia, la intensidad total de los dos canales debe ser igual [45]. A continuación se explican brevemente los métodos de normalización más usados:

Loess: se basa en el uso de una regresión no lineal de los valores de M que no modifica los valores de A porque supone que la mayoría de las sondas de la micromatriz no se expresa de modo diferencial ($M = 0$). Por eso, al representar los datos normalizados en gráficas MA, el valor medio de M es igual a cero, dejando la mitad de los datos a cada lado de $M = 0$ (figura 2.2c). Así, los puntos por encima de $M = 0$ se están sobreexpresando y los que quedan por debajo se están subexpresando. Se ha comprobado que los errores debidos a la distribución espacial de los puntos en la matriz se deben en gran medida a las agujas de impresión. Por eso se obtienen mejores normalizaciones cuando se ajusta por `loess` cada bloque impreso con la misma aguja (método `Print-tip-loess`), que cuando se normaliza toda la micromatriz a una única curva loess [45].

Scale: si los datos entre micromatrices muestran mucha variabilidad a pesar de estar normalizados, conviene realizar una segunda normalización entre ellas para poder comparar los resultados. El método `Scale` es el más eficaz en hacer que los datos sean equivalentes y comparables entre micromatrices, según una estimación de la mediana de la desviación absoluta. No obstante se ha comprobado que este método es de los que más altera la información biológica de los datos, y conviene evitarlo siempre que sea posible [149].

Quantile: se trata de un método no paramétrico, que se basa en los cuantiles, para corregir la intensidad de los canales R y G de las sondas entre las

matrices para dar la misma distribución a cada matriz. Se fundamenta en la media geométrica de cada canal por separado para obtener el mismo perfil de densidad para cada réplica. Este método supone que la distribución de la intensidad de los genes es aproximadamente la misma en todas las muestras. Sin embargo, no corrige las diferencias debidas a la distribución espacial de los puntos y produce cambios muy agresivos en los datos. Aunque resulte muy apropiado para las matrices de Affymetrix se ha descrito que produce resultados poco fiables en las de dos colores [45], siendo preferible la compensación por intercambio de fluoróforos que la modificación de los valores de R y G para que coincidan.

VSN: su nombre procede del acrónimo de *variance stabilization normalization* (normalización mediante estabilización de la varianza). Se basa en un principio completamente diferente a los anteriores: procura calibrar los errores debidos a factores experimentales, como la eficiencia de marcaje de cada fluoróforo, mediante transformaciones para la estabilización de la varianza basadas en glog (*generalised logarithmic transformations*) [185]. Glog asume que las intensidades sin procesar pueden considerarse como la suma de tres componentes: el ruido medio de fondo, el verdadero nivel de expresión, y variaciones en la expresión debidas a errores [45].

Mediante sondas de expresión conocida: todos los métodos anteriores normalizan toda la micromatriz usando todos los datos; pero también se ha propuesto que se puede normalizar con alguno de los métodos anteriores basándose en tan solo un grupo de genes invariantes, como los de mantenimiento, o los *spikes*, que son genes cuya expresión conoce de antemano el investigador [72]. Este tipo de normalización es particularmente útil cuando alguna de las condiciones que debe reunir para la normalización no se cumple (por ejemplo, la mayoría de las sondas muestran expresión diferencial). Las sondas de genes de mantenimiento han de corresponder a genes que realizan funciones básicas en las células y que se sepa que se expresan de forma prácticamente constante entre las diferentes condiciones analizadas. Suele resultar difícil encontrar genes de este tipo, y se ha demostrado también, en varios casos en los que se han utilizado para la normalización, que varían su expresión [167]. Los *spikes* son transcritos de ARN utilizados para calibrar las medidas de expresión en los experimentos de micromatrices. Cada uno está diseñado para hibridarse a una sonda de control específica que se imprime en la micromatriz, y que producirá valores de expresión conocidos [72]. La utilización de *spikes* como controles requiere añadirlos junto con el ARN antes del marcaje con los fluoróforos y de la hibridación [167].

2.5.5. Detección de la expresión diferencial

Uno de los objetivos que siempre está en los experimentos de micromatrices consiste en conocer los genes expresados diferencialmente (GED) entre las diferentes condiciones, tejidos, fases del desarrollo, etc., de las que se toman las muestras. La búsqueda de GED en las micromatrices conlleva la aplicación de análisis estadísticos en los que se contemple el problema de que hay muchas hipótesis (*multitesting*) dentro de un único experimento. No resulta adecuado utilizar únicamente las veces de cambio (FC, del inglés *fold-change*) para extraer los GED, ya que no se tiene en cuenta la varianza ni se aporta ningún valor estadístico que indique la fiabilidad de los resultados [10]. Por esto, junto con las veces de cambio, es necesario utilizar un valor estadístico que indique si el cambio es significativo en el contexto del experimento. Por ejemplo, los valores de P indican el valor de probabilidad de que un caso analizado cumpla la hipótesis nula (que no hay expresión diferencial), o de que el resultado obtenido sea al azar. El valor de P para cada sonda debe de corregirse para tratar las hipótesis múltiples que son los experimentos de micromatrices (véase más abajo). Se pueden considerar GED aquellos cuyo valor de P ajustado sea menor que α (valor de confianza), es decir, los que rechazan la hipótesis nula. El problema reside en que el valor de P (ajustado o no) no indica la dirección del cambio porque siempre es positivo; por eso hay que combinarlo con el valor de FC para determinar cuáles serán los verdaderos GED. El valor de FC que indica el límite de la expresión diferencial puede variar según los autores, por ejemplo, algunos proponen que un gen debe mostrar un FC $> 2{,}5$ [57], y otros sugieren utilizar un FC > 3 [193]. Pero si se tiene en cuenta que debe utilizarse en combinación con el valor de P, probablemente no haga falta valores de FC tan altos, sino que baste con un FC $> 1{,}5$ [169], puesto que no es necesario que una sonda cambie mucho de expresión para comprobar que su cambio es significativo si se repite de forma coherente en todas las repeticiones.

El valor estadístico t, por ejemplo, sí indica la dirección del cambio, a la vez de la probabilidad de que ese cambio sea significativo (cuanto mayor sea $|t|$, más significativo es el FC). La prueba estadística de la t de Student se utiliza con frecuencia para determinar las diferencias significativas entre dos grupos a través de las diferencias entre medias independientes. En principio sería el tipo de prue-

ba adecuado para detectar los GED. Pero uno de los desafíos que las micromatrices plantean a la estadística es que en un único experimento se analizan varios miles de genes simultáneamente en varias condiciones. Si se analiza la expresión diferencial de cada gen como un suceso independiente de los demás con una prueba t, la incidencia de positivos falsos (FDR, del inglés *false discovery rate*), o sea, de genes considerados como GED que realmente no lo son, es proporcional al número de pruebas realizadas y al valor de α que permite diferenciar las sondas que corresponden a los GED ($P < \alpha$). Con este análisis, la probabilidad de que aparezca un positivo falso es igual al valor de P. Por tanto, si se compara la expresión de 10 000 genes, basándose únicamente en el valor de $\alpha = 0,05$, el 5 % de la muestra (500 genes) podrían ser positivos falsos, es decir, saldrían como GED sin que realmente hubiese una expresión diferencial. Esto obliga a aplicar también los métodos para corregir esta multiplicidad de hipótesis (*multiple testing, multitesting*) mediante el ajuste del valor de P obtenido para cada prueba de t para cada sonda. Existen varios métodos estadístico de corrección del valor de P con distintos niveles de rigor. El método de de Bonferroni [27] es el más drástico, al indicar que entre los GED habrá, como mucho 1 positivo falso; para ello, basta multiplicar el valor de P de cada sonda por el tamaño de la muestra. Al ser tan drástico, en muchas ocasiones no arroja ningún GED, por lo que se suelen preferir otros métodos menos restrictivos, como el de Benjamini y Hochberg [23] o el de Holm y Bonferroni [100]. La distribución esperada de los valores de P tras un análisis corregido por la FDR es como la que aparece en la figura 2.4.

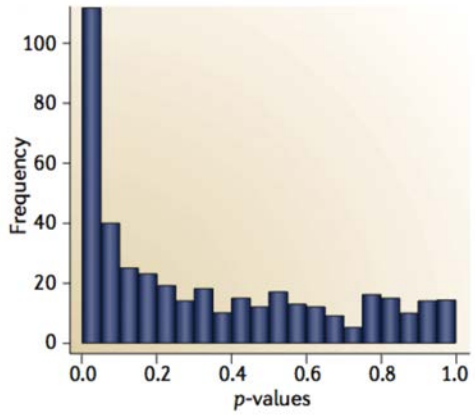


Figura 2.4: *Histogramas de valores de* P. *Imagen tomada de [10]*

Cuando el tamaño de la muestra es pequeño, como suele ocurrir en experimentos de micromatrices, la librería `limma` del paquete Bioconductor proporciona las funciones necesarias para corregir la prueba de la t de Student mediante ajustes bayesianos, y proporcionar lo que se conoce como estadísticas t moderadas (estadístico B). Con `limma` se pueden ajustar los datos de expresión de cada gen a un modelo lineal, lo que sirve para estimar la variabilidad en los datos [205], y los ajustes bayesianos son útiles especialmente cuando hay pocas réplicas (menos de 3), ya que en estos casos una prueba t esparciría mucho los valores de P [95].

Una alternativa a las pruebas de t de Student y de FDR son los análisis no paramétricos por rangos (*rank products*). Con este método se detectan los GED con mayor sensibilidad y selectividad que con los métodos paramétricos basados en la t de Student, sobre todo cuando el tamaño de la muestra es pequeño [30]. Esta técnica es muy potente para identificar cambios relevantes en la expresión biológica, llegando a necesitar menos réplicas para obtener resultados reproducibles [30, 102].

Dados los problemas técnicos y estadísticos de los análisis de las micromatrices, siempre conviene verificar parte de los resultados mediante otra técnica ajena a dichos problemas, como la RT-PCR o la transferencia Northern [35, 36]. Además, en los experimentos de micromatrices los cambios de expresión se comprimen de 2 a 10 veces debido al intervalo dinámico del escáner de las imágenes, por lo que al verificar las sondas por estas técnicas alternativas, casi siempre mostrarán cambios de expresión mayores que los FC obtenidos de las micromatrices.

2.5.6. Análisis funcional

Independientemente de la plataforma que se utilice para analizar las micromatrices, el resultado obtenido será en la mayoría de los casos una lista de genes expresados diferencialmente (GED) [120]. Por lo general, la estrategia seguida para obtener estos GED se basa primero en establecer puntos de corte arbitrarios que consideran sólo unos valores experimentales y descartan otros, aun a sabiendas de que aparecerán algunos positivos falsos y se descartarán otros candidatos (negativos falsos). A continuación, en una segunda etapa independiente se realizará el enriquecimiento de estos GED con términos relevantes desde el punto de vista biológico para tratar de adivinar qué ha pasado para que aparezcan los GED que se han obtenido [60]. Varios autores han señalado que la primera parte de esta estrategia, donde los genes se seleccionan sin tener en cuenta su comportamiento cooperativo, es su punto débil, ya que si los genes se consideran independientemente unos de otros, serán necesarios puntos de cortes muy restrictivos para reducir la tasa de positivos falsos [60]. Además, se ha descrito [162] que los resultados obtenidos por estos métodos se pueden ver

profundamente afectados por el punto de corte seleccionado, lo que alterará las conclusiones biológicas y hará descartar genes con cambios de expresión moderados a pesar de su significado biológico. En resumen, utilizar solo los GED para inferir los cambios biológicos reduce la fuerza estadística del análisis y promueve la inventiva del investigador para generar hipótesis en la que cuadren los GED que ha obtenido.

Por estos motivos se ha desarrollado una nueva generación de herramientas que se inspiran en los criterios de la biología de sistemas y que tienen como objetivo analizar el comportamiento de todos los genes para encontrar grupos de ellos que presenten algún tipo de relación funcional, en lugar de centrarse únicamente en el comportamiento individual de los genes. Estas herramientas se sirven a menudo de anotaciones funcionales como los términos de la *Gene Ontology* (GO) o las rutas KEGG [60, 165], que servirán para estudiar si los genes que comparten anotaciones iguales o muy similares muestran comportamientos también similares. Se trata de un análisis que interpreta los procesos celulares como una red en la que los componentes se están relacionados por sus funciones [162], y que basta con que la mayoría de los miembros de esa red se comporten de forma paralela para que dicha red se considere importante en el experimento. Por tanto, estos bloques de genes relacionados funcionalmente, proporcionan una explicación a nivel molecular de las características estudiadas en los experimentos de micromatrices [60]. Una gran ventaja de este análisis es que al necesitar solo una «mayoría» de genes, la misma función puede salir como interesante en dos experimentos parecidos a pesar de que las listas de GED no sean iguales. Además, este análisis no se ve afectado por un punto de corte elegido arbitrariamente, sino que aprovecha todos los datos del experimento y puede detectar cambios sutiles en muchos conjuntos de genes, que se pasan por alto por los métodos que se centran únicamente en los GED [162].

Algunos ejemplos de programas que sirven para analizar micromatrices basándose en este enriquecimiento biológico son

- GSEA (`http://www.broadinstitute.org/gsea/index.jsp`) [158, 214]: fue el primer programa de este tipo, fácil de usar, y probablemente el más utilizado, aunque hoy en día otros lo superen [162].
- SAFE (`http://www.bioconductor.org/packages/release/bioc/html/safe.html`) [19], que está integrado como un paquete de bioconductor y ofrece pruebas estadísticas más sólidas que GSEA.
- GeneCodis (`http://genecodis.dacya.ucm.es/`) [165], aplicación incluida en un servidor web, integra diferentes fuentes de información para encontrar patrones modulares de anotaciones relacionadas entre sí.
- FatiScan (`http://babelomics3.bioinfo.cipf.es/`) [6], aplicación incluida en un servidor web, permite incluir listas propias de anotaciones sin necesidad de que sean de organismos modelos.

Por desgracia, la mayoría de las herramientas disponibles para el análisis funcional están diseñadas para trabajar con secuencias de humano. Algunas, como GeneCodis, GAzer o GeneTrail, permiten trabajar con otros organismos modelo además de humano, como por ejemplo *Arabidopsis*, ratón o la levadura. Existen pocas opciones cuando se trabaja con especies no modelo [162], como las plantas del grupo gimnospermas. En este sentido cabe destacar FatiScan, que permite realizar análisis funcionales con anotaciones realizadas por el propio investigador, lo que abre la posibilidad de los análisis funcionales a las especies no modelo.

Capítulo 3

Secuenciación de alto rendimiento

La evolución de las técnicas de secuenciación a lo largo de los últimos años ha generado un gran cambio en la investigación científica aplicada a la biología y a la medicina, donde se ha pasado de secuenciar los genes uno a uno de modo manual en los años 80 [42, 218] a proyectos de secuenciación de mil o más genomas, en la actualidad [53, 168, 81]. En la secuenciación manual era posible descifrar varios cientos de nucleótidos por reacción. Posteriormente, al desarrollarse la secuenciación automática con capilares se pudo aumentar la capacidad de trabajo a casi mil nucleótidos por molécula y reacción. En la actualidad, y con el desarrollo de la secuenciación de nueva generación (del inglés *next generation sequencing*) (NGS), se consiguen obtener incluso 600 nt de más de un millón de moléculas a la vez en reacciones a escala nanotecnológica. Para tratar este volumen de secuencias, es esencial la ayuda de la bioinformática.

3.1. Secuenciación clásica (manual)

En 1977 se publicaron a la vez el método de secuenciación química de Maxan y Gilbert [151] y el método de secuenciación de terminación de cadena de Sanger [191], también conocido como método Sanger o método enzimático o secuenciación por didesoxinucleótidos. Este método, esquematizado en la figura 3.1, se basa en la formación de un extremo 3'mediante los didesoxinucleótidos trifosfato (ddNTP) que la ADN polimerasa no puede seguir extendiendo porque los ddNTP carecen de grupo hidroxilo en 3′. La ADN polimerasa sintetiza hebras complementarias a la que se quiere secuenciar, por lo que necesita un cebador, un oligonucleótido diseñado para que se hibride con el extremo 3′ de ésta, además de una mezcla de dNTP (uno de ellos radiactivo) y ddNTP. Para llevar a cabo la secuenciación se realizan cuatro reacciones por separado, cada una con un ddNTP diferente. De este modo se obtienen secuencias truncadas en diferentes posiciones de 3′ del nucleótido correspondiente a cada tubo. Estas moléculas se separaran por tamaño en una electroforesis, utilizando un carril para cada tubo, para visualizar el patrón de bandas del que se puede deducir la secuencia [191].

Frederick Sanger fue premiado con su segundo Nobel de química en 1980 por su contribución en la determinación de las secuencias de los ácidos nucleicos, compartiendo el premio con Walter Gilbert y Paul Berg [164], lo que puede dar una idea de lo que su técnica aportó a la investigación científica, posibilitando la secuenciación, entre otros, del genoma humano [51, 224].

3.2. Secuenciación automática

El método de Sanger se adaptó posteriormente para la secuenciación automática en capilares, sustituyendo los isótopos radiactivos del marcaje de los ddNTP por un marcaje con 4 fluorocromos diferentes. Con ello se consiguió realizar la reacción en un único tubo y migrarla en un solo capilar (figura 3.2a). En uso de un capilar en lugar de un gel plano confería más resolución al método [190]. Además, en las máquinas se incorporaron varios capilares para paralelizar este proceso, de modo que los equipos más productivos de secuenciación de este tipo pueden llegar a secuenciar simultáneamente hasta 96 y 384 secuencias, ya que disponen de un capilar para cada pocillo de una placa de 96 o 384 pocillos [201, 202]. Después de tres décadas de continuo perfeccionamiento, es posible obtener secuencias de unas 1000 pb con una fiabilidad del 99,999 % y con un coste de unos 0,5 $ por kilobase [201].

Las etapas de esta secuenciación (figura 3.2a) son las siguientes:

- Fragmentación del ADN mediante enzimas de restricción o por rotura física.
- Preparación de las muestras por clonación *in vivo* en un vector plasmídico en *E.coli*, y pos-

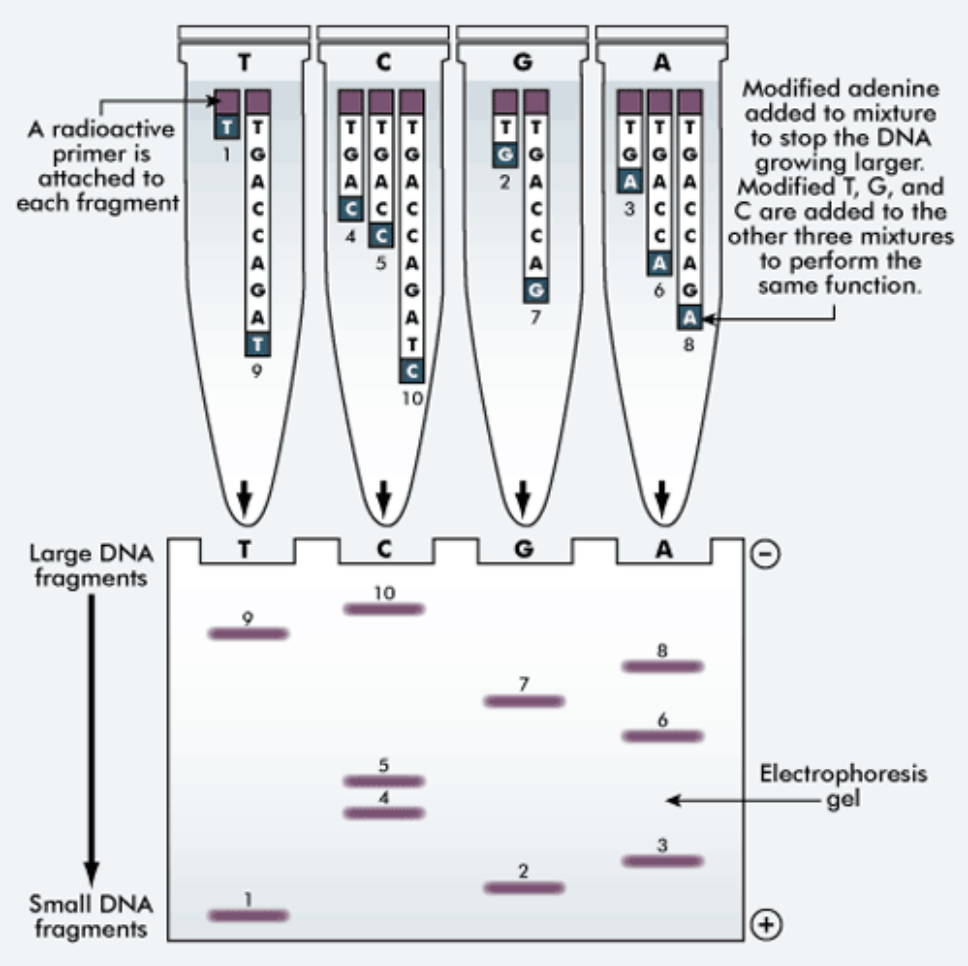


Figura 3.1: *Método de secuenciación por didesoxinucleótidos de Sanger. Imagen tomada de* `http://dna-rna.net/`

terior amplificación y purificación del ADN clonado.

- Elongación de cebadores con la ADN polimerasa para generar los fragmentos de secuencia acabados en un ddNTP que llevará su correspondiente fluoróforo.
- Separación de los fragmentos por electroforesis en los capilares del secuenciador.
- Lectura de la fluorescencia, que tras su análisis por el software informático, asigna una base diferente a cada pico del cromatograma (también denominado electroferograma) generado.

Esta técnica permitió secuenciar los genomas de las principales especies modelo como el ser humano [51, 224], la planta *Arabidopsis thaliana* [16], la mosca *Drosophila melanogaster* [2], y otros muchos proyectos de transcriptómica [212, 143, 135].

La secuenciación del genoma de *Arabidopsis thaliana* dio lugar a una expansión en la investigación de plantas y permitió la explotación de los genes anotados para conocer los ortólogos en otras plantas. También facilitó el camino para la secuenciación de varios genomas de otras plantas modelo y de algunos genomas de plantas de cultivo [74].

Sin embargo, a pesar del gran éxito de la secuenciación de Sanger, esta técnica presentaba ciertas desventajas que limitaban su aplicación a gran escala, principalmente el coste y la necesidad de la clonación de los ácidos nucleicos en vectores y su amplificación en hospedadores. Por ejemplo, la secuenciación inicial del genoma de levadura, de *Arabidopsis* y del genoma humano necesitó consorcios de muchos laboratorios, conllevando un gran esfuerzo económico, ya que cada megabase costó aproximadamente 1330 $ [29].

3.3. Secuencias pareadas

La técnica de secuencias pareadas fue descrita por Edwards y Caskey en 1991 [66], y posterior-

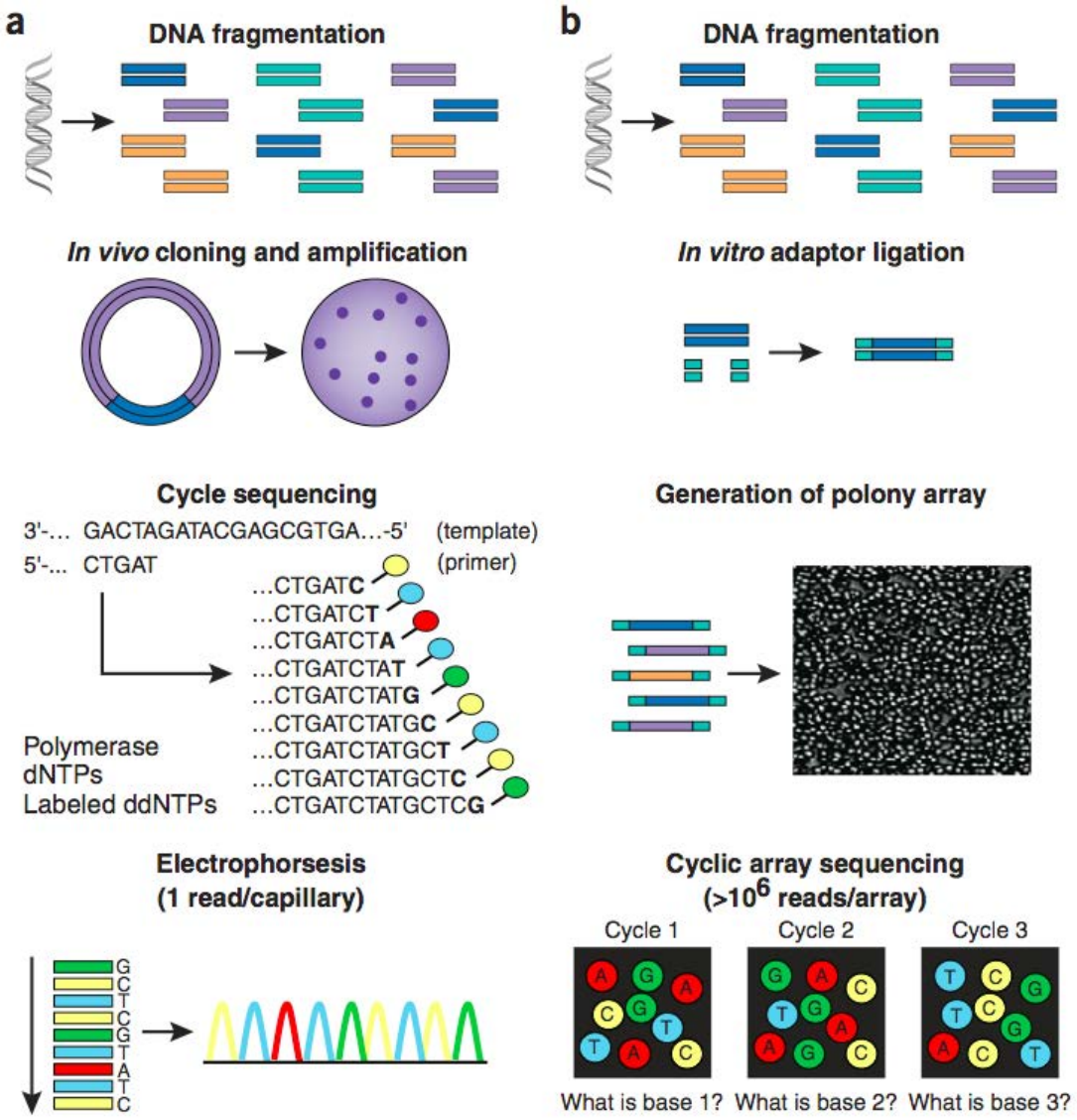


Figura 3.2: *Paralelismo de las etapas a seguir en la secuenciación automática basada en el método de Sanger y en las nuevas tecnologías de secuenciación de alto rendimiento. (a) Secuenciación automática Sanger: las etapas se pueden resumir en fragmentación del ADN, preparación de las muestras* in vivo*, elongación de los cebadores, ordenamiento de los fragmentos por electroforesis en un capilar y lectura de la señal por un receptor. (b) Secuenciación de nueva generación (NGS): sus etapas son fragmentación del ADN, preparación de las muestras* in vitro*, elongación de los cebadores y captura de imágenes de la micromatriz. Imagen tomada de [201]*

mente varios grupos han desarrollado variantes de esta técnica [203]. Se basa en obtener dos secuencias a partir de los extremos de una molécula de ADN de longitud conocida. En el proceso de ensamblaje, las secuencias pareadas contienen la misma información que las secuencias simples y además se conoce también qué posición relativa debe tener con respecto a su pareja. Esta información se utiliza posteriormente para corregir errores de ensamblaje, para reconocer la presencia de repeticiones largas, y para generar supercontigs (también, denominados *scaffolds* figura 3.3). Los *scaffolds* están formados por contigs que están separados por un fragmento de secuencia desconocida, pero que se pueden ordenar gracias a que cada contig contiene un extremo de las secuencias pareadas [203]. Los fragmentos desconocidos pueden deberse a que esa zona no se ha secuenciado o a que es altamente repetitiva. Así pues, esta técnica tiene las mismas ventajas que las secuencias simple y además, permite resolver grandes regiones genómicas en un único *scaffold* [203].

Al comienzo de su desarrollo, las secuencias pareadas se utilizaban en genomas de gran tamaño y

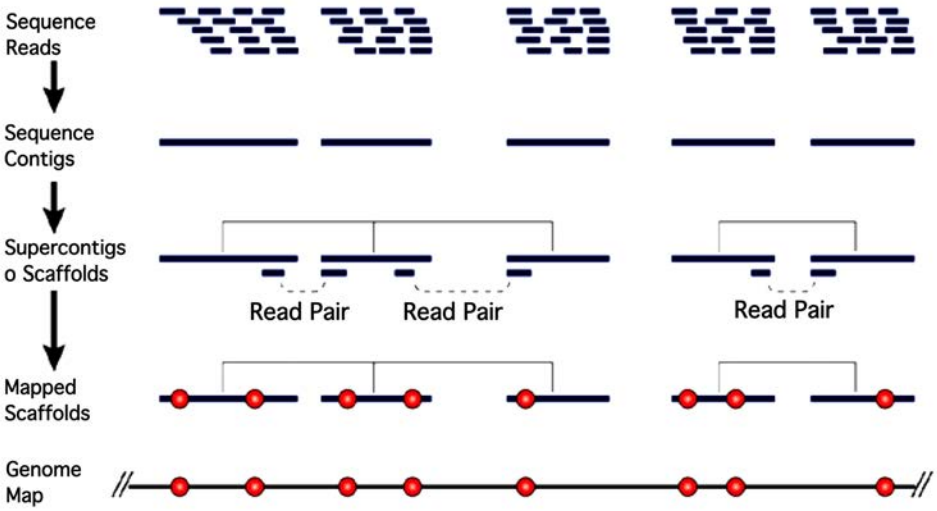


Figura 3.3: *Estrategia de secuenciación utilizando pareadas. A partir de las lecturas generadas en un proyecto de secuenciación de cualquier tecnología o estrategia, se obtienen las secuencias consenso de los contigs mediante el ensamblaje con un programa informático. Las secuencias de los contigs se organizan entonces formando un* scaffold *basándose en la información de las secuencias pareadas para ordenar contigs no solapantes. Finalmente los* scaffold *pueden ordenarse igual que el genoma identificando en ellos marcadores que se conozcan en el genoma, como por ejemplo STS, marcadores moleculares y genes (círculos rojos). Figura tomada de [88]*

con gran cantidad de repeticiones. Por ejemplo, ya se aplicó en la secuenciación del genoma humano [51, 224], antes de la aparición de las NGS. Pero con la llegada de las NGS, las secuencias pareadas se han aplicado tanto para la secuenciación de genomas como de transcriptomas. Por ejemplo, muy recientemente se han aplicado en la secuenciación del genoma de la patata [178] en combinación con secuencias simples de otras tecnologías de NGS, produciendo secuencias genómicas muy fiables, que han sido confirmadas con genotecas de fósmidos y BAC, y datos de expresión procedentes de EST.

3.4. Secuenciación de nueva generación

La llegada de la NGS ha reemplazado a la secuenciación basada en capilares en la mayoría de sus aplicaciones debido a su menor coste por base secuenciada (tabla 3.1) y a que no necesita clonación [29]. Tanto los estudios transcriptómicos como los genómicos han avanzado mucho gracias a estas nuevas tecnologías [213]. Se considera que una de las aplicaciones más importantes de las técnicas de NGS es la secuenciación y caracterización de transcriptomas, tanto en especies modelo como en no modelo [113]. El coste de secuenciación con las tecnologías de NGS se reduce a entre 4 y 90 $ por megabase, según la plataforma utilizada. A continuación se describen las tres tecnologías más usadas: 454/FLX de Roche, Solexa de Illumina y SOLiD de Applied Biosystems.

Tabla 3.1: *Comparación de las principales características de las plataformas de secuenciación de nueva generación con respecto al método de Sanger*

	Longitud	Precisión	Coste
Sanger	1000 pb	99,9 %	1330$/Mb
Roche-454	800 pb	99 %	90$/Mb
Solexa	100 pb	98,5 %	4$/Mb
SOLiD	75 pb	99 %	4$/Mb

3.4.1. Plataforma 454/FLX de Roche

En 2005 aparece el primer método comercial de NGS, un sistema de secuenciación escalable, verdaderamente paralelizable (no como las 96 o 384 reacciones en paralelo máximas de la secuenciación automática), y de mayor capacidad que los de tipo Sanger. Este método se basa en chips de fibra óptica de $60 \times 60\ mm^2$ que contienen aproximadamente 1 600 000 micropocillos (*Picotiter plate*) en los que se realizan las PCR en paralelo, puesto que en cada uno entra una única microesfera que contiene todos los reactivos para la PCR con los cebadores de ADN que van fijados en las microesferas, y una molécu-

la de ADN a secuenciar. Con la PCR se amplifica la molécula a secuenciar, y luego, en la reacción de secuenciación, se añade un nucleótido diferente a cada vez. Si es el nucleótido que necesita la cadena de ADN en formación se emitirá luz por la acción de la luciferasa (figura 3.4). A cada paso se añade un nucleótido diferente y se toma una imagen fotográfica de todo el chip. En función de los pocillos que emitieron luz al añadir cada nucleótido se obtiene la secuencia del ADN que hay en cada micropocillo. De esta forma se consigue secuenciar 25 millones de bases con una precisión del 99 % en 4 horas [147], con un coste de 90 $ por megabase (tabla 3.1).

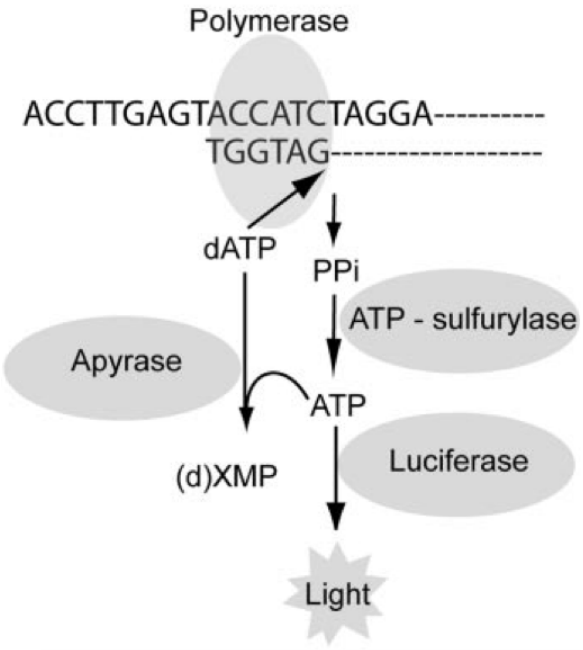


Figura 3.4: *Enzimas acopladas en la polimerización del ADN que intervienen en la pirosecuenciación. Imagen tomada de [5]*

En la figura 3.2.b aparecen las etapas clave de este método, que se resumen a continuación para poderlas comparar mejor con las del método de Sanger:

- Fragmentación física del ADN por nebulización.
- Preparación *in vitro* de las muestras de manera muy sencilla, en la que solo se requiere la ligación de adaptadores a los fragmentos de ADN, sin clonación.
- Elongación de los cebadores en cada uno de los millones de micropocillos en los que se está secuenciando una molécula. En cada paso se añade un desoxinucleótido diferente (dATP, dCTP, dGTP, dTTP) y cada vez que alguno se incorpora a la cadena se libera un pirofosfato, que será sustrato de varias reacciones enzimáticas acopladas (figura 3.4), [5]), en las que la cantidad de pirofosfato liberado se mide en forma de intensidad luminosa.
- Captura de imágenes de la matriz después de cada etapa de reacción con una cámara CCD. Con la información obtenida en este paso se obtienen los flujogramas (también denominados pirogramas) que determinarán la secuencia (figura 3.5).

Se puede decir que se requieren menos preparativos, produce una cantidad mucho mayor de nucleótidos por reacción, es más barata, y el proceso global (gracias a que no incluye la clonación) es más rápido, además de no necesitar mucho espacio en los congeladores a −80 °C para almacenar los clones. El altísimo rendimiento de la técnica permite que se secuencien a la vez muestras de varios experimentos diferentes para optimizar aún más los costes. Cada muestra de cada experimento se identificará posteriormente mediante un análisis bioinformático trivial gracias a que se le añaden por ligación unas etiquetas denominadas identificadores multiplexados (MID), que permiten separar los resultados de cada experimento [213].

El método 454 de Roche es el más costoso de los de nueva generación (tabla 3.1), pero tiene la ventaja de producir las lecturas de mayor longitud, hoy en día muy cercana a la que proporciona la tecnología de Sanger. Las lecturas largas son especialmente útiles cuando se estudian organismos con genomas complejos con un alto número de repeticiones, ya que cuanto más larga sea la secuencia, mayor será la probabilidad de que contenga una repetición corta completa, y no se generarán huecos al ensamblarlas. Así pues, por el hecho de ser la técnica de NGS que produce lecturas de mayor longitud, es la que más se utiliza para el estudio de genomas [222, 22, 223, 89, 231] y transcriptomas de organismos no modelo [166, 18, 7, 183, 78, 14, 215], o cualquier tipo de ensamblaje *de novo*.

Sin embargo, esta plataforma de secuenciación produce un error de secuencia muy característico debido a que la cantidad de luz producida se ha de interpretar para conocer el número de nucleótidos que se incorporaron (figura 3.5). Como la cantidad de luz no se correlaciona de forma lineal con el número de nucleótidos que se incorporaron, al secuenciar homopolímeros de más de 5 a 8 nucleótidos es probable que el algoritmo asignador de bases se equivoque, y la probabilidad de error aumenta cuanto más largo sea el homopolímero, casi siempre volcando menos nucleótidos de los que realmente contiene.

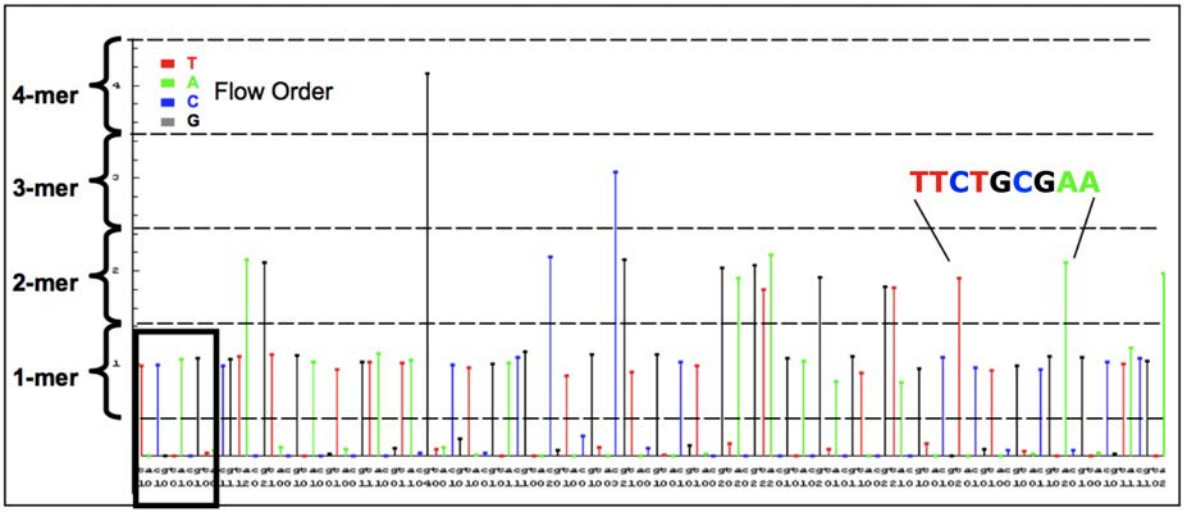


Figura 3.5: *Flujograma de una reacción con 454/FLX. En cada ciclo de secuenciación se añade un nuevo desoxinucleótido en el orden TACG. Cuando el desoxinucleótido añadido se incorpora a la cadena, en el flujograma se indica con una barra, que variará su altura en función del número de desoxinucleótidos incorporados y su color en función del desoxinucleótido añadido en ese ciclo. En el recuadro negro se señala la clave de la reacción, 4 nucleótidos introducidos al principio de cada secuencia para calibrar la intensidad de luz detectada por nucléotido. Finalmente, un programa informático determina la secuencia basándose en la altura de cada emisión de luz en el flujograma. Imagen tomada de* `http://www.my454.com/downloads/news-events/how-genome-sequencing-is-done_FINAL.pdf`

3.4.2. Plataformas Solexa, SOLiD y otras

Las otras dos tecnologías más ampliamente utilizadas en la actualidad son Solexa de Illumina y SOLiD de Applied Biosystems. Ambos métodos generan secuencias más cortas que las obtenidas con el 454 de Roche, pero con una mayor cobertura y a menor coste [201]. Al producir lecturas más cortas, el ensamblajes *de novo* resulta más difícil con estas plataformas [177], pero son muy útiles para otras aplicaciones como la secuenciación de genomas procariotas o sin un alto número de repeticiones, la resecuenciación, el análisis de transcriptomas, la secuenciación de ARN pequeños y para obtener una mayor cobertura, algo que resulta muy útil cuando se desea caracterizar los SNP. A pesar de todo, se han llevado a cabo unos pocos estudios en plantas no modelo utilizando únicamente esta técnica [213], y ha resultado especialmente útil en combinación con otras tecnologías que produzcan secuencias más largas. Esta tecnología tiene un rendimiento de 25-200 Gb por reacción, requiere entre uno y ocho días para completar el proceso y obtiene secuencias con una longitud entre 26 y 150 pb a un coste de 4 $ por megabase. La precisión de las secuencias es de un 98,5 %, pudiendo secuenciar hasta ocho librerías en paralelo (Tabla 3.1).

Las librerías de Solexa se obtienen con métodos que produzcan secuencias flanqueadas por adaptadores. Ambos cebadores de PCR, tanto en la hebra directa como en la reversa, están fijos a un sustrato sólido mediante un conector flexible. De este modo, todos los amplicones procedentes del mismo molde permanecerán inmovilizados y agrupados sobre el soporte. Este método utiliza nucleótidos modificados que actúan como terminadores reversibles de la reacción (ver figura 3.6), de modo en cada paso solo se pueda añadir un nucleótido a la cadena de ADN. Como cada uno de los 4 tipos de nucleótidos tiene unido un fluoróforo diferente, después de cada paso de extensión, se toman imágenes en 4 canales diferentes, uno para cada fluoróforo. A medida que las lecturas producidas son más largas, aumenta la tasa de error, principalmente debido a sustituciones [201]. En el grupo deL Dr. Heinz Himmelbauer en el CRG de Barcelona acaban de valorar que la tasa de errores del reciente modelo HiSeq2000 de Illumina es mayor de la descrita y que también afecta a los homopolímeros (XI Jornadas de Bioinformática, Barcelona, 2012, P31).

Las librerías de SOLiD también hay que generarlas con métodos que produzcan secuencias flanqueadas por adaptadores. La secuenciación clonal se realiza por PCR en emulsión y los amplicones producidos se capturan en microesferas magnéticas. Después de romper la emulsión se recuperan las microesferas con los productos amplificados y se inmovilizan en un sustrato sólido y plano para generar una densa micromatriz desordenada. La principal

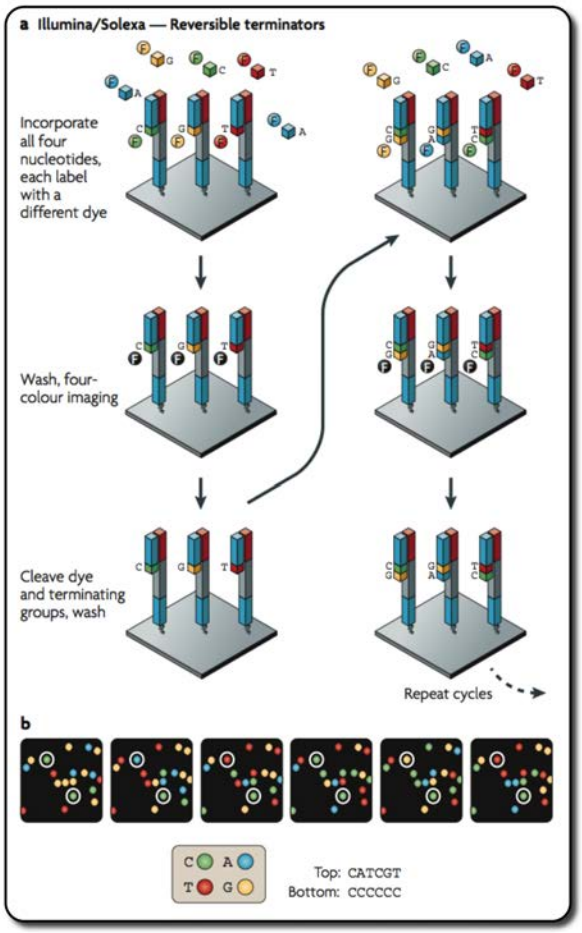


Figura 3.6: *Método de terminación reversible de cuatro colores de Solexa. (a) En cada ciclo de secuenciación se producen varios pasos, y comienza por la incorporación de nucleótidos modificados para bloquear la elongación de la cadena, cada nucleótido marcado con un fluoróforo diferente. Seguidamente se procede al lavado y la captura de imágenes. Finalmente se elimina el fluoróforo y la molécula que evita la elongación, y se expone a un agente reductor para regenerar el grupo 3'-OH necesario para continuar la cadena y se realiza otro lavado. (b) Ejemplo de las imágenes tomadas durante la secuenciación, que servirán para reconstruir la secuencia de cada molécula de DNA. Figura tomada de [154]*

particularidad de este método es que no se secuencia por polimerización sino por ligación. En cada ciclo de secuenciación se produce la ligación de una población de octámeros degenerados marcados con fluorescencia [201]. La máquina SOLiD 5500 produce entre 10-300 Gb de secuencias cortas (máximo de 75 pb) en cada ejecución, con un coste de 4$ y una precisión del 99 % (tabla 3.1).

La aplicación de secuencias pareadas está también disponible tanto para Solexa como para SOLiD, aunque con Roche 454 y SOLiD las secuencias pareadas producidas están orientadas en el mismo sentido y en el caso de Illumina la orientación es sentido-antisentido [213].

Se han desarrollado varias tecnologías, que están en constante evolución, cuyo objetivo es obtener secuencias más largas y de mayor calidad para reducir la dificultad de los ensamblajes. Entre estas tecnologías es posible encontrar IonTorrent (Life Technologies, Carlsbad, USA), SMRT (Pacific Biosciences, Menlo Park, USA), y Helicos (Helicos Biosciences, Cambridge, USA) [213]. Además, en el caso de SMRT de Pacific Biosciences se requiere menos cantidad de fungibles que en otras tecnologías de NGS, por lo que se espera que en un futuro se reduzcan más los costes de secuenciación.

3.5. Estrategias de secuenciación

En la actualidad se utilizan principalmente dos estrategias para abordar la secuenciación de genomas, BAC a BAC (cromosomas artificiales de bacterias), o mediante la secuenciación de todo el genoma (WGS, del inglés *whole genome sequencing*). La elección de la estrategia más apropiada dependerá del tamaño y de la complejidad del genoma que se quiere secuenciar, así como de las tecnologías utilizadas y del presupuesto disponible. Otra opción más barata y rápida consiste en la secuenciación del transcriptoma para el estudio de los genes que se expresan (EST, etiquetas de secuencias expresadas, del inglés *expressed sequence tags*) en determinadas condiciones experimentales, útil para obtener información en especies no modelo con genomas complejos.

BAC a BAC: El genoma o un cromosoma que se quiere secuenciar se trocea en fragmentos solapantes de menor tamaño (100 a 200 kpb aproximadamente) que se clonan en vectores BAC (figura 3.7-A.1). Después se seleccionan los BAC y se ordenan en un mapa físico (figura 3.7-A.2). Al final, cada BAC se secuencia por separado tras una digestión parcial con una enzima de restricción que producirá fragmentos solapantes al azar, en lo que se conoce por el término inglés *shotgun* (figura 3.7-A.3). Al secuenciar los BAC de uno en uno se consigue reducir la complejidad del ensamblaje, de modo que esta estrategia puede resultar útil para genomas con gran cantidad de repeticiones. Sin embargo, crear la genoteca de BAC y mapear cada clon sobre el genoma supone una enorme cantidad de esfuerzo, además de conllevar mucho tiempo y un gran coste económico. Por eso, esta técnica se está utilizando cada vez menos (figura 3.8).

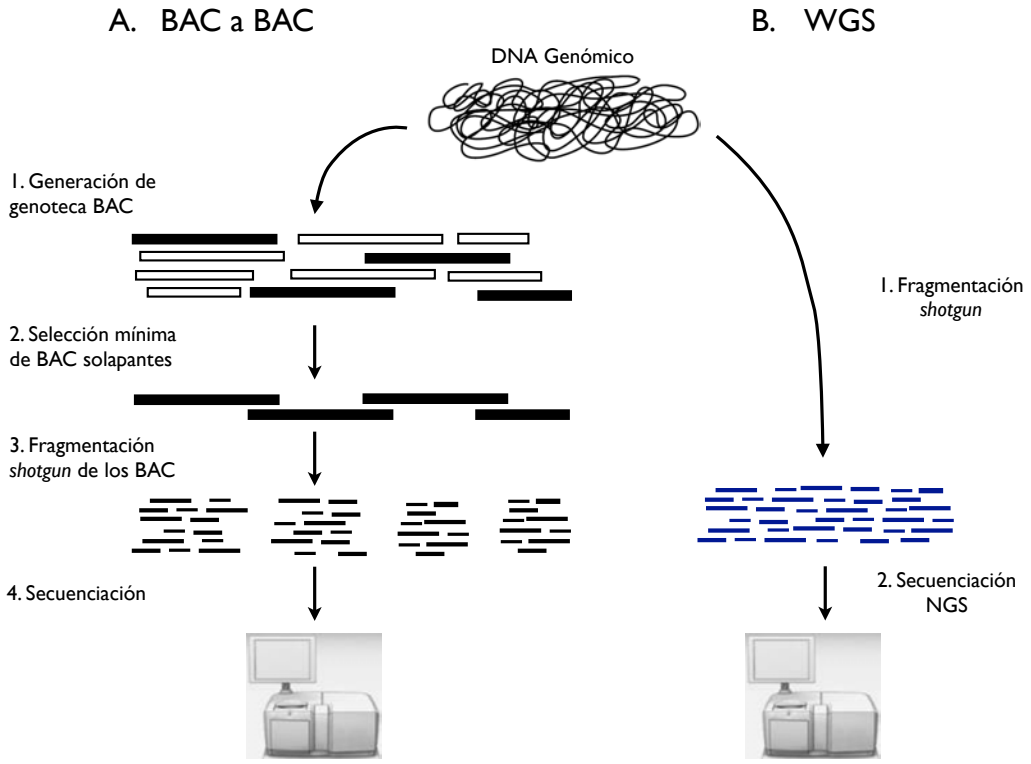


Figura 3.7: *Comparación de las estrategias de secuenciación (A) BAC a BAC, y (B) WGS.*

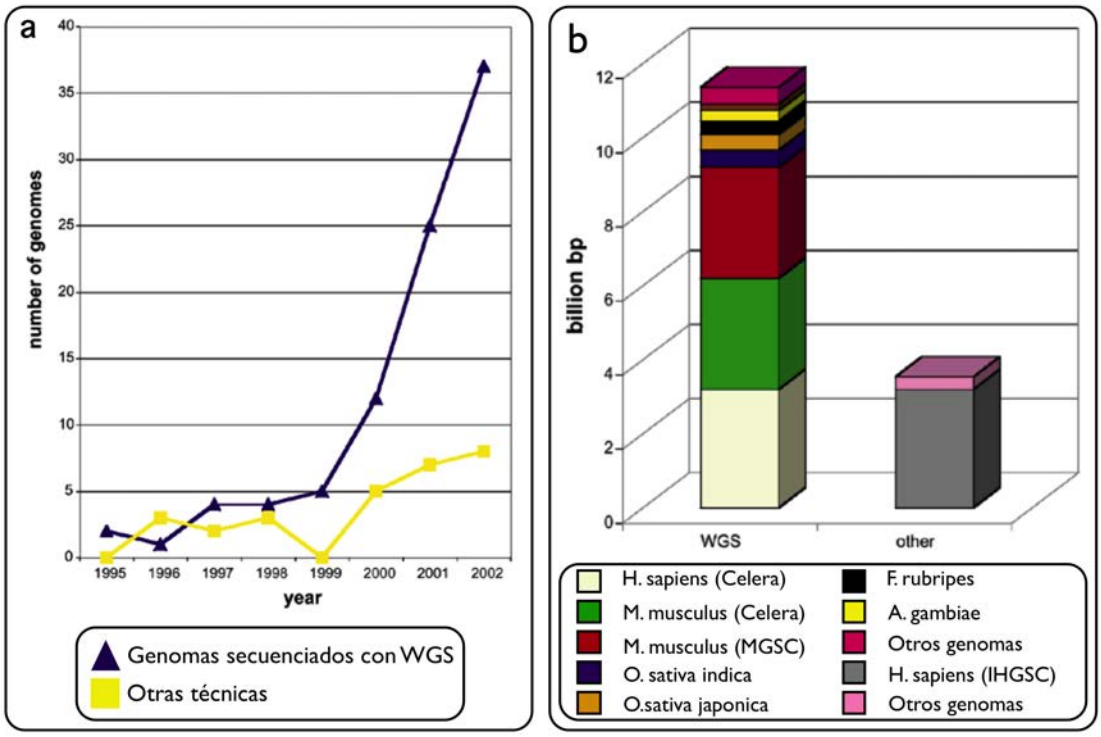


Figura 3.8: *Número de genomas secuenciados cada año desde 1995. (a) Número de genomas secuenciados utilizando WGS y con otras estrategias. (b) Cada columna indica el tamaño acumulado (en miles de millones de pares de bases) de los genomas secuenciados por WGS (columna de la izquierda) y por otros métodos (columna de la derecha). Figura tomada de [225]*

WGS: En esta estrategia se evita la generación de genotecas BAC fragmentando directamente el genoma al azar en elementos solapantes de menor tamaño (figura 3.7-B.1). Los fragmentos se secuencian directamente con tecnologías de NGS (figura 3.7-B.2), o se clonan primero y se secuencian después en el caso de las tecnologías de secuenciación basadas en capilares. Debido a su mayor sencillez, esta es la estrategia de secuenciación de genomas más utilizada hoy en día (figura 3.8). A diferencia del caso anterior, aquí toda la complejidad de la secuencia del genoma tendrá que resolverse con aplicaciones bioinformáticas, ya que no se dispone de ninguna información orientadora.

Secuenciación de transcriptomas: La secuenciación de *pools* de ADNc a menudo se utiliza para caracterizar el transcriptoma de un organismo de un modo rápido y barato. El transcriptoma de un organismo engloba al conjunto de genes que se expresan (EST) en una célula o conjunto de células, que incluyen tanto a los ARN que codifican proteínas como los no codificantes (ARNnc). Este tipo de secuenciación proporciona información sobre los genes de un organismo a bajo coste en comparación con la secuenciación de genomas, ya que solo se investigan las regiones que se transcriben en lugar del genoma completo. La generación de EST a partir de ARNm se considera la estrategia más frecuente y más útil para descubrir genes [113].

En los casos en los que el análisis transcriptómico está enfocado a estudiar los genes que codifican proteínas es importante utilizar secuencias de ARN enriquecidas en poly-$(A)^+$, ya que de este modo se reduce en gran parte los ARN no deseados, como los ARN pequeños y los abundantes ARN ribosómicos (ARNr). También es deseable realizar la normalización de las muestras de ARN, ya que así se reduce la aparición de los transcritos más abundantes y se aumenta la de los poco abundantes [43].

3.6. De lecturas a genes

3.6.1. Preprocesamiento

Después de obtener las lecturas de una secuenciación es indispensable preprocesarlas para eliminar los elementos que provienen de la manipulación del ADN o del ARN en el laboratorio. Así se obtendrá únicamente la parte útil y se descartan los fragmentos de baja calidad, contaminantes, vectores, adaptadores [47], y otros artefactos. En las nuevas tecnologías, además, es posible encontrar otros elementos nuevos que se deben considerar en el preprocesamiento. Por ejemplo, el etiquetado con MID u otras etiquetas implica que hay que retirarlas de la secuencia final, y además hay que usarlas para agrupar las lecturas por experimentos. En resumen, la calidad y la fiabilidad del posterior ensamblaje dependerá de un buen preprocesamiento [70] ya que, de otra manera, los consensos obtenidos podrían contener numerosos errores.

Ya existen programas para preprocesar secuencias de tipo Sanger, como SeqClean (`http://compbio.dfci.harvard.edu/tgi/software/`), Lucy [137], ESTPrep [192] y SeqTrim [70]. Algunas herramientas bioinformáticas como TagCleaner [197] y NovoBarCode (Novocraft) se han diseñado para identificar los MID y otras etiquetas, y clasificar las secuencias procedentes de los diferentes experimentos.

En cuanto a los contaminantes, los más comunes en el laboratorio suelen ser hongos y bacterias, como por ejemplo *Delftia* [67]. También se consideran contaminantes los restos de tejidos humanos procedentes de los investigadores y microorganismos utilizados con frecuencia en el laboratorio, como *E.coli* [63] y *Agrobacterium tumefaciens* [110], este último sobre todo cuando se trabaja con plantas. Si no se eliminan las secuencias procedentes de organismos contaminantes se corre el riesgo de considerarlas parte del organismo que se está estudiando [174]. Para la detección y eliminación de secuencias contaminantes existen programas como DeconSeq [196].

Ninguno de los programas anteriores se basta por sí solo para un buen preprocesamiento. Por eso, para la NGS se están desarrollando nuevas herramientas bioinformáticas especializadas como SeqTrimNext (ver apartado 10.1.2) o el paquete Galaxy [84] (`https://main.g2.bx.psu.edu/`).

3.6.2. Ensamblaje de un transcriptoma

El principal objetivo de este paso es la reconstrucción *in silico* de las secuencias de ADN o ARN original, fragmentadas durante la preparación de las muestras y la secuenciación. En el caso de los ensamblajes de transcriptomas, el objetivo es agrupar todas las lecturas del mismo gen en una secuencia consenso, con la unión o separación de las isoformas, si se desea. Las secuencias consenso formadas por la unión de varias lecturas se denominan *contigs* y las lecturas que no solapan con otras se conocen como singulones. El conjunto de contigs y singulones es conocido como unigenes.

Con el ensamblaje se evita la redundancia y se procura que el número de unigenes se aproxime al de los genes reales. En un caso ideal, los millones

de lecturas quedarán representadas por unas pocas decenas de miles de unigenes. Sin embargo, en la práctica algunos genes están representados por varios unigenes porque pueden ser diferentes fragmentos del gen que no solapan lo suficiente entre sí o porque pueden ser diferentes parálogos del gen o incluso diferentes alelos. El problema es que no existe criterio claro para distinguir un alelo de un parálogo reciente de forma general, por lo que los ensambladores se equivocarán [213]. También está el problema de que muchas lecturas no tienen una secuencia real, sino que son el producto de una interpretación errónea, con lo que provocarán también errores de ensamblaje.

A la hora de elegir el ensamblador más adecuado al experimento hay que tener en cuenta varios factores. El primero de ellos es la cantidad de secuencias y los recursos de computación disponibles, porque para el ensamblaje de transcriptomas complejos o para grandes conjuntos de secuencias es necesario contar con ordenadores con gran capacidad de cálculo [213] (por ejemplo, los disponibles en el Centro de Supercomputación y Bioinformática [SCBI] de la Universidad de Málaga). Otro factor a tener en cuenta es la plataforma de secuenciación, ya que algunos algoritmos —como los contenidos en Velvet [237], SOAPdenovo [136] y ABySS [204]— están preparados para trabajar con secuencias cortas, como las obtenidas con SOLiD o Solexa, mientras que otros lo están para secuencias largas, como las obtenidas con el 454 de Roche (Euler-SR [175], MIRA3 [44], CABOG [155] y Newbler [147]). Además, en las estrategias transcriptómicas, suelen incorporarse todavía EST de tipo Sanger junto a las secuencias de NGS [93, 215], por lo que es necesario utilizar un ensamblador que permita realizar ensamblajes mixtos, como MIRA3 o CABOG. También es importante elegir programas con buena documentación, mantenimiento y con ficheros de salida fáciles de utilizar en posteriores pasos [213].

Para especies no modelo y casos en los que no se dispone de una referencia adecuada se realizan ensamblajes *de novo*. Sin embargo, en estos casos es difícil ensamblar correctamente las secuencias producidas por los ayustes alternativos procedentes de un mismo gen [213]. La alternativa, cuando se dispone de un genoma o transcriptoma de referencia adecuado, es realizar un ensamblaje mediante mapeo, en el que se alinean las lecturas sobre la secuencias de referencia.

Los dos principales tipos de algoritmos de ensamblaje son los basados en grafos de De Bruijn y los basados en solapamiento de secuencias, los *overlap-layout-consensus* (OLC).

Ensamblaje por solapamiento: Esta estrategia se basa en tres pasos.

1. Búsqueda de regiones solapantes entre las secuencias de dos en dos, de manera que cada una se compara con todas las demás. Hay que definir la región mínima de solapamiento y el porcentaje mínimo de identidad deseados para considerar correcto el ensamblaje. Este cálculo puede llegar a saturar a los ordenadores, sobre todo con el volumen de lecturas que son capaces de proporcionar los modelos más recientes de las plataformas de NGS.

2. Construcción de un esquema (grafo) en el que cada solapamiento es una relación que conecta a cada una de las demás secuencias (nodos) con las que solapan.

3. Resolución del grafo para obtener las secuencias ensambladas mediante la búsqueda de una ruta hamiltoniana. Este es el único paso que se podría paralelizar para acelerar la computación, ya que en los anteriores se requiere manejar todas las secuencias conjuntamente.

Esta estrategia es la clásica, y ya se empleaba con las lecturas de tipo Sanger (por ejemplo, en CAP3 [105] y Phrap [56]). Sin embargo para ensamblar secuencias de NGS se han tenido que recodificar los algoritmos para adaptarlos a las necesidades específicas de esta tecnología. Los casos más claros son CABOG [155], una evolución para lecturas de 454 del ensamblador de Celera, MIRA3, la evolución para lecturas de 454 de miraEST [44], y Newbler [147], desarrollado por Roche específicamente para 454.

Grafos de De Bruijn: Es la estrategia más utilizada para ensamblar secuencias cortas, que se basa en las siguientes etapas:

1. Fragmentación de las lecturas secuenciadas en una colección de oligomeros (k-meros), donde todas las lecturas tienen la misma longitud k. Los cebadores redundantes se comprimen en uno solo para reducir la complejidad del análisis, aunque se conserva la información del número de veces que se repiten. Los valores de k suelen asignarse entre 19 y la longitud de las lecturas. De esta forma se evita el paso limitante de tener que comparar todas las secuencias entre sí.

2. Construcción de un grafo de De Bruijn con el conjunto de k-meros, de modo que cada nodo contiene un k-mero de longitud $k-1$ que se solapa exactamente en $k-2$ nucleótidos con otros nodos.

3. Búsqueda de los caminos eulerianos que reconstruyen la secuencia original de la que proceden todas las lecturas relacionadas

Esta estrategia se ve muy afectada por las repeticiones y por los errores de secuenciación que aparecen en los ensamblajes por solapamiento [156]. Además, como el ADN es de doble cadena, a la hora de reconstruir los caminos eulerianos puede darse el caso que algún k-mero aparezca en el sentido de la transcripción y otro en antisentido, provocando secuencias consenso artefactuales. Algunos ejemplos de los ensambladores más populares que utilizan esta estrategia son Euler-SR [175], Velvet [237], SOAPdenovo [136] y ABySS (el único que es parcialmente paralelizable por el momento [204]).

3.6.3. Anotación

Se anota por similitud

Una vez se han ensamblado los unigenes de un transcriptoma o se han reconstruido las regiones genómicas se procede a anotar las secuencias, ya que el interés de tener las secuencias de un transcriptoma o un genoma reside en la obtención de información biológica que permita seguir profundizando en el funcionamiento de los organismos y su relación con el entorno. Este incremento de conocimiento científico podrá posteriormente traducirse en aplicaciones prácticas que conlleven beneficios económicos y mejoras en la calidad de vida.

La anotación de genes se lleva a cabo habitualmente de un modo automático e informatizado. Existen muchas herramientas bioinformáticas libres, como BLAST, que resulta de gran utilidad para encontrar parecidos significativos con genes conocidos que están almacenados en las bases de datos. La anotación por similitud de secuencia tiene algunas limitaciones, especialmente en las plantas (y sobre todo en los organismos no modelo), donde el número de genes con anotaciones es menor, sobre todo porque se basan principalmente en la información que se conoce de *Arabidopsis thaliana* [213]. La anotación basada en similitud será tan buena como la anotación de las secuencias almacenadas en las bases de datos con las que se compara. Para comparar las secuencias de nucleótidos de los unigenes obtenidos durante el ensamblaje de un transcriptoma suele emplearse con frecuencia BLASTx con bases de datos que contienen proteínas conocidas, como *refseq_protein* de GenBank [26] o SwissProt y TrEMBL, ambas de UniProt [15]. Estas dos bases de datos de secuencias de proteínas, tienen diferentes niveles de anotación, por ejemplo, SwissProt está revisada manualmente, y TrEMBL, utiliza anotaciones automáticas procedentes de referencias cruzadas [15]. Suele ser frecuente, cuando se trabaja con especies no modelo, encontrarse una gran cantidad de ortólogos de función desconocida. Además hay que tener en cuenta que si no se toman las precauciones adecuadas, a veces se incorporan a las bases de datos secuencias con errores de anotación o secuencias que realmente son contaminantes o artefactos [174]. Si se considera como buena una secuencia mal anotada, este error también se mantendrá en nuestras secuencias anotadas por similitud. También existe la posibilidad de anotar con BLASTn para confirmar que los transcritos reconstruidos con el ensamblaje fueron secuenciados también en otros experimentos de EST.

Anotaciones bioinformáticamente útiles

Gracias a la anotación por similitud se puede enriquecer la información de los transcritos reconstruidos con una definición, y con otra información, como la ontología de genes (GO, del inglés *Gene Ontology*) [17], las rutas metabólicas en las que intervienen los transcritos, recogidas en los mapas KEGG [116], los dominios proteicos registrados en InterPro [107], y en caso de tener actividad enzimática, el código de la EC (del inglés, *Enzyme Commission*).

Gene Ontology: El objetivo del consorcio de la GO es producir un vocabulario controlado y dinámico, aplicable a todos los seres vivos, incluso aunque el conocimiento acerca de la función de los genes y de las proteínas en las células esté en continuo cambio [17]. Con este fin se desarrollaron tres ontologías independientes: procesos biológicos, funciones moleculares y componentes celulares. La GO, como todas las ontologías, mantiene un lenguaje controlado que es útil para las personas y para los ordenadores, ya que cada una de sus ontologías tiene un código numérico (por ejemplo, GO:1234567) práctico para los análisis bioinformáticos, y una descripción para cada elemento que sea informativa para los usuarios. Además, la GO mantiene una relación jerárquica entre sus elementos, que permite inferir relaciones entre los genes que contienen estas anotaciones. Estas ontologías están disponibles en `http://www.geneontology.org/`

Mapas KEGG: Los mapas de rutas metabólicas de KEGG son diagramas gráficos que representan interacciones moleculares y redes metabólicas, procesos con información genética, ambiental, procesos celulares, sistemas orgánicos y enfermedades humanas [116]. Se pueden consutar en

http://www.genome.jp/kegg/, y son de gran utilidad para relacionar entre sí los genes que participan en una misma ruta, algo muy útil cuando se realizan análisis funcionales.

Código EC: La Comisión Internacional para Enzimas se creó en 1956 en la Unión Internacional de Bioquímica y Biología Molecular para evitar que la misma enzima recibiera diferentes descripciones o nombres. Su función fue aportar un nombre descriptivo sobre la reacción que cataliza la enzima, y un código único para cada función enzimática. Las enzimas se nombran con cuatro números separados por puntos, EC 1.2.3.4, donde el primer número da la característica más genérica y los siguientes son cada vez más específicos. Así, el primer número puede tomar 6 valores; 1 para oxidoreductasas, 2 para las transferasas, 3 para las hidrolasas, 4 para las liasas, 5 para las isomerasas y 6 para las ligasas. Todo lo referente con estos códigos se puede encontrar en http://www.chem.qmul.ac.uk/iubmb/enzyme/

InterPro: La base de datos InterPro integra modelos predictivos de varios repositorios (Pfam, PRINTS, PROSITE, SMART, ProDom, PIRSF, SUPERFAMILY, PANTHER, CATHGene3D, TIGRFAMs y HAMAP). Cada uno se centra en diferentes aspectos biológicos o utiliza una metodología distinta para encontrar el denominador común de las secuencias. El propósito de InterPro es combinar los puntos fuertes de cada uno de los repositorios para poner a disposición de la comunidad científica una única fuente con información estructurada sobre familias de proteínas, dominios y regiones funcionales [107]. InterPro está disponible en http://www.ebi.ac.uk/interpro/. Existe una herramienta que se dedica expresamente a encontrar los dominios InterPro para una colección de proteínas concretas: InterProScan (http://www.ebi.ac.uk/Tools/pfa/iprscan/)

Programas para anotar

Se han desarrollado muchas herramientas bioinformáticas para obtener información biológica acerca de los productos de los genes que contienen nuestras secuencias. La mayor parte de ellas se basan en la búsqueda de anotaciones por similitud con otras secuencias ortólogas conocidas que están almacenadas en bases de datos. Por ejemplo, la herramienta más conocida y más utilizada para buscar información es BLAST+ [34], y quedarse con las anotaciones de la primera secuencia de la lista.

Otra herramienta muy popular y cómoda es Blast2GO [49], que tiene una interfaz fácil de utilizar y ejecutable en cualquier sistema operativo en el que esté instalado Java. Blast2GO ejecuta BLAST de modo remoto para para determinar a que se parecen las secuencias que se desean analizar, y de ahí extrae los términos de la *Gene Ontology*, la nomenclatura de la *Enzimme Commission*, las rutas KEGG y los códigos InterPro.

En 2005 se describió una herramienta de anotación de secuencias transcriptómicas conocida como AutoFact [126]. Este programa realiza búsquedas por similitud utilizando BLAST con varias bases de datos (UniRef90, UniRef100, NCBI's nr, COG, KEGG, Pfam, Smart, est_others, LSU [Large SubUnit ribosomal RNA], SSU [Small SubUnit ribosomal RNA]), y entre todas ellas elige la descripción que considera más adecuada, clasificando las anotaciones obtenidas de un modo estructurado. AutoFact también anota las enzimas con la nomenclatura de la *Enzimme Commission* y las rutas KEGG.

Conviene mencionar que la anotación de secuencias genómicas tiene que seguir un protocolo totalmente diferente, y no vale ninguna de las herramientas anteriores. Una herramienta destacable es Maker [37], creada para la anotación de genomas eucariotas, capaz de identificar repeticiones y alinear EST y secuencias de proteínas a la secuencia genómica de interés para predecir los genes que contiene. También utiliza tres predictores de exones y de uniones entre intrones y exones para poder dar precisión de nucleótido a las predicciones que realiza. Recientemente ha aparecido MAKER2 [101], una nueva versión de este programa adaptada a proyectos de NGS, en el que el análisis de los datos se realiza con varias CPU simultáneamente y no le supone ningún problema el análisis de un gran volumen de secuencias.

3.6.4. Mapeo

El mapeo (del inglés *mapping*) consiste en alinear secuencias cortas, como las obtenidas con NGS, frente a otras secuencias de referencia de mayor tamaño, como puede ser un genoma. Esta técnica se utiliza a menudo para resecuenciar genomas [188], para análisis cuantitativos de expresión de ARN [160], y para la detección de SNP [232]. Todos los años están saliendo nuevas herramientas que aceleran el proceso y aumentan la fiabilidad y la versatilidad.

Las secuencias se mapean sobre un transcriptoma habitualmente con el objeto de cuantificar la expresión [112] o incluso detectar SNP [232]. Pero en las especies no modelo, al carecer de un genoma de referencia, el alineamiento de las lecturas puede servir para comprobar que el transcriptoma ensamblado proporciona secuencias que se parecen a las lecturas

originales, como se propone en un estudio de la formación de la madera con angiospermas [232], en el que utilizan Bowtie [129] con este propósito. Cuando el objetivo del mapeo es la detección de SNP, es necesario permitir algunos apareamientos erróneos (*mismatch*) para que las lecturas se alineen a la referencia a pesar de mostrar algún cambio. Sin embargo, si el objetivo del mapeo es realizar una estimación de cuantas de las lecturas originales se detectan en las secuencias obtenidas tras el ensamblaje, es deseable que el alineamiento sea exacto, sin errores.

Capítulo 4

Bases de datos

Debido a la gran cantidad de datos, incluidas las secuencias, sus anotaciones y los datos de expresión de micromatrices (capítulo 2), se ha vuelto imprescindible la aplicación de la tecnología informática de bases de datos para mantener y guardar de forma ordenada toda la información biológica que hoy en día se está generando. Y no solo para ofrecerla a la comunidad científica, sino para poder gestionarla dentro de los laboratorios que las están generando. Estas bases de datos deben tener interfaces fáciles de manejar y herramientas para consultar los datos disponibles, de modo que resulten útiles a usuarios sin una gran experiencia en informática. Por este motivo conviene saber que pueden ser de diferentes tipos en función de como se organicen sus datos: jerárquicas, de red, relacionales, orientadas a objetos y documentales. Las bases de datos relacionales son las más utilizadas y las que mejor se adaptan a los proyectos de genómica y transcriptómica.

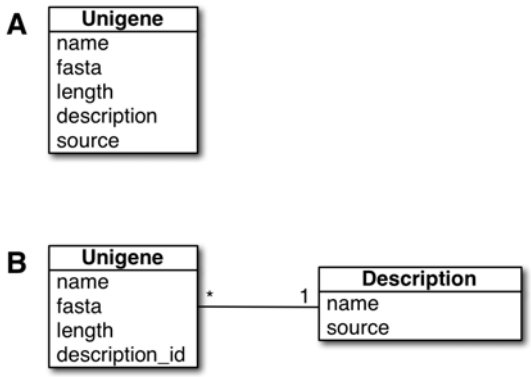


Figura 4.1: *Ejemplo de tablas para una base de datos de tipo relacional sin normalizar (A) y normalizada (B)*

4.1. Bases de datos relacionales

Una base de datos de tipo relacional almacena la información en tablas, que a su vez se organizan en columnas, que se relacionan entre sí. En un caso ideal, el diseño del esquema de la base de datos debe contener una estructura de tablas completamente normalizada, es decir, la información no se almacena de un modo redundante, ya que diferentes tablas de la misma base de datos no almacenan la misma información, y las columnas no almacenan valores repetidos.

Esto queda más claro si se observa el modelo más simple sin normalizar (figura 4.1-A), en la que un unigén tiene una sola tabla `Unigene` que contiene su nombre (`name`), secuencia (`fasta`), longitud, la descripción (`description`), y la fuente (`source`) de la que obtuvo la información. El contenido de la descripción puede estar vacío en muchos casos, y también puede que muchos unigenes compartan la misma descripción. Esto implica que se estará ocupando en disco un espacio en el que no hay información, o que se estará escribiendo muchas veces el mismo contenido, una vez por cada unigén que lo tiene. En resumen, el contenido de la base de datos ocupará mucho espacio vacío o con los mismos datos para guardar la información.

En cambio, en una base de datos relacional normalizada (figura 4.1-B), la información sobre la descripción y su fuente se colocan en una tabla nueva (`Description`) a la que hará referencia un identificador (`description_id`). La principal ventaja de la división en tablas es que los unigenes sin descripción no ocuparían registros vacíos en la tabla `Description`, lo que ahorra mucho espacio al no rellenar miles de registros sin información en cada tabla. Además, para aquellos que compartan la misma descripción, solo hará falta guardarla una vez. Supongamos un ejemplo en el que 1000 unigenes obtuvieron la descripción *Unknown Protein* con el programa BLASTx. En este caso, con el esquema de la figura 4.1-A se almacenarían 1000 entradas en la columna `description` con valor *Unknown Protein* y 1000 entradas con *blastx* para la columna `source`. En cambio, con el esquema normalizado de la figura 4.1-B, lo que se hace es almacenar la pareja *Unknown Protein* y *blastx* una única vez en la

tabla `Description`, y en la tabla `Unigene` solo aparecerá el identificador `description_id` que apunta a dicha información. Por eso, ahora cada uno de los 1000 unigenes tendrá valores propios para `name`, `fasta`, y `length`, pero a todos se le asignará el mismo `description_id`. Se podría resumir que una base de datos normalizada permite ocupar solo el espacio necesario y no malgastar disco en sectores sin información, con la consiguiente aceleración de las búsquedas.

También es esencial en toda base de datos de transcriptómica o genómica, con miles de genes y anotaciones, un modo de consultar la información fácil y rápido. Para ello se suelen incluir motores de búsqueda de texto que localizan palabras clave dentro de la información de la base de datos, por lo general en las anotaciones de los genes, y muestran al usuario de modo ordenado la información de la base de datos que está relacionada con la búsqueda realizada. Otro modo de buscar información en una base de datos de secuencias consiste en realizar una comparación con BLAST para encontrar si la secuencia que se quiere estudiar se parece a alguna de las que hay en la base de datos, y así conocer la información disponible para ella.

En este trabajo, las bases de datos se desarrollaron y gestionaron con Ruby On Rails, comentado en el apartado *Ruby On Rails y Django* del apartado 5 (pág. 35). Con este entorno de trabajo, el desarrollo de la base de datos se divide en tres partes, modelo, vista y controlador:

- El modelo indica las relaciones entre las tablas de la base de datos relacional
- En la vista se desarrolla la página web que mostrará la información de la base de datos
- El controlador es el intermediario que consulta la base de datos relacional y envía los datos necesarios a la vista que se está consultando

El lenguaje más utilizado para consultar y gestionar bases de datos es SQL (*Structured Query Language*), aunque con *Ruby On Rails* se utiliza Ruby y comandos propios de *Ruby On Rails* para asociar la web con la base de datos y realizar consultas (*Ruby On Rails* se encarga de traducir estos comandos a SQL para interactuar con la bases de datos, lo que permitiría crear las bases de datos sin conocimientos de SQL). Además, otra ventaja de *Ruby On Rails* es que el mismo código es válido para interaccionar con bases de datos creadas en SQLite, MySQL y Oracle.

4.2. Bases de datos internacionales

4.2.1. Generalistas

Las nuevas técnicas de secuenciación están provocando que el número de genomas de orgánulos, cloroplastos y mitocondrias, este en continuo aumento [221], al igual que el número de secuencias almacenadas en las tres principales bases de datos internacionales (GenBank [26], EMBL [117], DDBJ [217]). Para una mayor comodidad, en la tabla 4.1 están disponibles las direcciones web de las bases de datos mencionadas en este apartado. Estas bases de datos almacenan y ponen a disposición de la comunidad científica muchos tipos de información para un gran número de organismos, aunque aquí nos centraremos esencialmente en las relacionadas con secuencias nucleotídicas. GenBank (mantenida por el *National Center for Biotechnology Information*, NCBI, EE.UU.), EMBL (mantenida por el *European Bioinformatics Institute*, EBI, Unión Europea) y DDBJ (*DNA Data Bank of Japan*, en Japón) tienen una colaboración para compartir repositorios [118], de modo que todas las secuencias están disponibles para los usuarios en las tres bases de datos, independientemente de en cual se añadió originalmente. Sin embargo, el continuo crecimiento al que se ven sometidas cada año, junto con la explosión en el crecimiento producida por las NGS [123], está provocando incluso que se anuncie la posibilidad de cerrar algunos repositorios [213], como el de lecturas de NGS, *The Sequence Read Archive* (SRA) [133] de NCBI. Una alternativa firme para continuar dando soporte a estas lecturas es *The European Nucleotide Archive* (ENA) [132].

Debido al aumento de datos se han desarrollado bases da datos con menor redundancia para acelerar las consultas y comparaciones. Un ejemplo de ello son las bases de datos de proteínas RefSeq del NCBI o UniRef50, UniRef90, UniRef100 de UniProt. Estas tres últimas bases de datos de UniProt solo contienen la secuencia más larga para los genes que tienen una similitud entre sí del 100 %, del 90 %, o del 50 %, es decir si en UniRef90 10 genes tienen una secuencia idéntica al 90 %, en la base de datos solo se almacenaría el más largo de de ellos, y lo mismo ocurre en UniRef100 y Uniref50, siendo los porcentajes de identidad 100 % y 50 % respectivamente.

4.2.2. Específicas

Otro modo muy interesante y útil de abordar el problema de la gran cantidad de datos que se están generando en biología es el desarrollo de nuevas

Tabla 4.1: *Resumen de bases de datos más representativas y sus direcciones web*

Database	URL
Bases de datos de nucleótidos primarias	
GenBank	http://www.ncbi.nlm.nih.gov/genbank/
EMBL	http://www.ebi.ac.uk/embl/
DDBJ	http://www.ddbj.nig.ac.jp/
Bases de datos de lecturas	
SRA	http://www.ncbi.nlm.nih.gov/sra
ENA	http://www.ebi.ac.uk/ena/
Bases de datos de organismos secuenciados	
Ensembl	http://www.ensembl.org/
Phytozome	http://www.phytozome.net/
PlantGDB	http://www.plantgdb.org/
Bases de datos de un organismo modelo	
TAIR	http://www.arabidopsis.org/
FlyBase	http://flybase.org/
MGD	http://www.informatics.jax.org/
ZFIN	http://zfin.org/
Bases de datos de coníferas	
CGN	http://pinegenome.org/
TreeGenes	http://dendrome.ucdavis.edu/treegenes/
ConiferGDB	http://www.conifergdb.org/cgdb3p1/tiki-index.php
ConiferEST	http://www.conifergdb.org/coniferest/index.php
DFCI	http://compbio.dfci.harvard.edu/tgi/plant.html

bases de datos especializadas en sólo un organismo o varios organismos relacionados, o en secuencias que comparten algo en común. Por ejemplo, está Ensembl [106], que contiene secuencias de genomas completos de vertebrados y otros eucariotas principalmente, o Phytozome [86] para estudios de genómica comparativa en las plantas verdes (*Viridiplantae*), con genomas de plantas de este grupo, anotaciones de PFAM, KOG, KEGG y PANTHER. También es posible encontrar PlantGDB [64], otra base de datos de plantas verdes, con EST, ensamblajes, genomas y herramientas bioinformáticas.

En relación con las bases de datos de especies modelo es posible encontrar TAIR [104] para *Arabidopsis thaliana*, FlyBase [61] para *Drosophila melanogaster*, The Mouse Genome Database (MGD) [32] para ratón (*Mus musculus*), o The Zebrafish Model Organism Database (ZFIN) [210] para el pez zebra (*Danio rerio*), con los genomas de estos organismos y herramientas bioinformáticas para consultar su mapa físico y genético, y conocer qué proteínas se codifican en sus genes.

En cuanto a bases de datos de coníferas, existe el portal *The Conifer Genomics Network (CGN)* con noticias e información de los proyectos y grupos de investigación implicados, y enlaces a TreeGenes [229] y ConiferGDB. TreeGenes contiene datos genómicos de especies forestales, que incluyen EST, SNP, mapas genéticos, marcadores moleculares, fenotipos y QTL. ConiferGDB incluye un buscador de contigs y BAC de *Pinus taeda* y la base de datos de EST de coníferas, ConiferEST [139] —que parece fuera de servicio—, con EST preprocesados y anotados, y un sistema para ver las lecturas originales coloreadas para resaltar el preprocesamiento y visualizar los valores de calidad. Por otro lado está The Gene Index Project [180] con EST de píceas, (DFCI Spruce Gene Index), y pinos (DFCI Pine Gene Index). Las secuencias consenso construidas para las especies del Gene Index Project se construyen mediante agrupamiento (*clustering*), ensamblaje y anotación de secuencias de EST procedentes de GenBank [180], basándose únicamente en secuencias de tipo Sanger, sin llegar a incorporar secuencias de NGS. En la última versión del DFCI Pine Gene Index (9.0 del 26 de marzo de 2011) hay ensamblados 77 326 unigenes, clasificados en 44 858 posibles consensos, que serían el equivalente a los contigs, y 32 337 singulones de EST. Desde 2011 también está disponible EuroPineDB, una base de datos de transcriptómica principalmente basada en *Pinus pinaster*, y que se describirá en este trabajo.

La mayoría de las EST de pino almacenadas en las bases de datos proceden de proyectos que buscan conocer mejor la formación de la madera y los mecanismos que regulan su crecimiento [11, 38, 12, 170],

aunque también se pueden encontrar proyectos destinados a conocer mejor otros procesos biológicos como la embriogénesis somática [33], la respuesta a insectos patógenos [182], la respuesta a estrés hídrico [141] o a la disponibilidad de una fuente de nitrógeno [36].

Capítulo 5

Lenguajes de programación

A continuación se incluye un artículo publicado en el número 134 de verano de 2011 de *Encuentros en la Biología*, revista de divulgación científica editada por la Facultad de Ciencias de la Universidad de Málaga. En él se describen brevemente los lenguajes de programación más corrientes en bioinformática, sus ventajas e inconvenientes, ejemplos para usuarios inexpertos y una breve explicación de las diferencias entre los lenguajes compilados y los lenguajes interpretados o de *scripting*.

Lenguajes de programación para la bioinformática

Noé Fernández Pozo
Contratado predoctoral de la Universidad de Málaga, Plataforma Andaluza de Bioinformática
noefp@uma.es

Saber programar te puede facilitar la vida

Hoy en día, con el avance de la informática, las redes de comunicación y las nuevas técnicas de laboratorio, podemos disponer de gran cantidad de información en formato digital. En el caso de la biología, hay multitud de bases de datos que almacenan información procedente de experimentos biológicos y se están realizando experimentos con aparatos que nos pueden devolver una gran cantidad de información en ficheros informáticos, como por ejemplo los secuenciadores de nueva generación y los experimentos con micromatrices.

Las bases de datos de secuencias muestran un crecimiento exponencial, de modo que el volumen de información que se acumula no puede ser procesado manualmente. Así que, si queremos sacar conocimientos útiles de las grandes cantidades de datos, no nos queda más remedio que recurrir a la informática. Si conocemos algún lenguaje de programación, podremos acceder a muchos programas y módulos ya creados para procesar datos biológicos e incluso desarrollar nuestros propios *scripts* (pequeños programas realizados en lenguajes interpretados), de modo que podamos realizar un análisis a medida de nuestros datos.

En principio, un usuario básico puede pensar que le costará mucho que los lenguajes de programación sirvan para sus intereses. Para eliminar este prejuicio conviene empezar con cosas sencillas que según nuestras necesidades nos ahorren bastante trabajo. Por ejemplo, ese usuario debería aprender a utilizar interpretes de comandos (lo que se conoce como el *shell*), por ejemplo el *bash*, que vienen instalados necesariamente en los sistemas operativos Linux y Mac OS X. Vamos a ver un par de ejemplos muy sencillos, en los que nos podemos ahorrar mucho tiempo, tan solo escribiendo una línea de código. Si tecleas

```
cat *fasta > todas_las_seqs.fasta
```

podrás unir en un sólo fichero todos los ficheros de secuencias con extensión fasta que tengas en el directorio en el que te encuentres. El comando cat imprime en pantalla el fichero o ficheros que le indiquemos. Al poner *fasta le estamos indicando a cat que imprima todos los ficheros con cualquier nombre (*) que acabe en la cadena de texto fasta. Después de esto, al poner el '>' seguido de un nombre de fichero de salida, estamos indicando que queremos que el resultado, en lugar de salir por la pantalla, se escriba en el fichero de salida indicado. Con:

```
grep -c '^>' todas_las_seqs.fasta
```

podremos saber cuantas secuencias hay dentro del fichero todas_las_seqs.fasta porque el comando grep sirve para imprimir únicamente las líneas que contiene un texto. Para indicar este texto se puede utilizar lo que se conoce como expresiones regulares. Al indicar a grep que imprima las líneas que comiencen (^) por '>', extraerá las líneas que contienen el nombre de la secuencias, pero si utilizamos el parámetro '-c' (*count)*, en lugar de esto, obtenemos el número de líneas que comienzan por '>' en nuestro fichero fasta, o lo que es lo mismo, cuantas secuencias contiene.

Lenguajes interpretados frente a lenguajes compilados

Los lenguajes interpretados o de *scripting*, utilizan un programa que interprete el código para que se pueda ejecutar. El interprete verifica que no haya errores sintácticos antes de ejecutar el programa. De este modo se pueden realizar muchos cambios sobre la marcha e ir viendo cómo han afectado. Sin embargo, los programas realizados en lenguajes compilados hay que convertirlos en binarios ejecutables cada vez que se quiere comprobar un cambio y, si no funcionan correctamente, se vuelven a editar, compilar y ejecutar. En cambio, cuando el programa ya esta acabado y compilado sin errores, son mucho más rápidos que los *scripts* realizados en lenguajes interpretados. Por estos motivos, con los lenguajes interpretados, como R, Perl, Ruby o Python, se aprende a programar con más facilidad y se desarrollan algoritmos y prototipos en muy poco tiempo. Sin embargo, los lenguajes compilados, como C o C++, proporcionan ejecutables mucho más rápidos, a los que se les habrá dedicado mucho más esfuerzo y tiempo.

Por ejemplo, en el caso del algoritmo BLAST que se utiliza en los servidores del NCBI, es impensable que estuviera escrito en un lenguaje interpretado ya que, a través de dicho servidor, se realizan miles o millones de peticiones diarias. La diferencia de tiempo, por mínima que sea, se volverá muy significativa si en lugar de estar escrito en C, lo estuviera en Perl. Por eso, en muchos módulos de BioPerl y de otros lenguajes interpretados, existen subrutinas escritas en C que se encapsulan (sin el que usuario lo perciba) para ejecutar la parte más dura del algoritmo; el resultado se devuelve en el lenguaje interpretado sin que el usuario perciba el trasiego de lenguajes. De este se aceleran algunas partes básicas de los programas escritos en lenguajes interpretados.

Otro motivo por el cual los lenguajes compilados son más rápidos es porque no verifican el código del programa durante su ejecución, lo que pone obliga al inexperto a realizar el tedioso trabajo de arreglar los errores en el código. Sin embargo, en los lenguajes interpretados, la verificación del código es constante, lo que permite ir arreglando los errores que vayan surgiendo en nuestros *scripts* en el momento.

Para hacer pequeños programas o *scripts* que cambien el formato de nuestros datos, obtengan la información útil de un fichero o analicen nuestros datos de forma rápida y automática, es interesante para todo biólogo aprender, al menos a un nivel básico, un leguaje interpretado. Entre los más utilizados en biología están Perl, Python, Java y Ruby.

Parte II

Objetivos

Con este trabajo se pretende profundizar en el conocimiento del transcriptoma de *Pinus pinaster*, una especie de gran interés económico y forestal. Para ello se desarrollarán herramientas bioinformáticas que permitan extraer información biológica útil de los datos de expresión génica y secuenciación realizados en el grupo de investigación Biología Molecular y Biotecnología de Plantas (BIO-114). Se pretende que estas herramientas, aunque en principio estén orientadas a resolver problemas de las coníferas, también se puedan aplicar al estudio del transcriptoma de cualquier especie eucariota no modelo.

Tanto los datos originales como los procesados se integrarán en una base de datos para ofrecer a la comunidad científica un transcriptoma depurado y anotado que por lo pronto ya ha permitido desarrollar nuevas micromatrices que permitan profundizar en el conocimiento del transcriptoma, localizar ortólogos y parálogos de genes y vías metabólicas de interés, y empezar a abordar el genoma de *Pinus pinaster*. En la figura 5.1 se resumen las interacciones entre las distintas áreas del transcriptoma de pino que se abordarán en este trabajo.

En concreto, los objetivos plantados en este trabajo son:

1. Desarrollo de un análisis bioinformático automático para los resultados de expresión génica basados en micromatrices de dos colores.

2. Desarrollo de un flujo de trabajo que incluya tanto herramientas propias como otras disponibles en la bibliografía para proporcionar un análisis de secuencias semiautomático que abarque desde el preprocesamiento a la anotación, y cuyo resultado requiera la menor intervención manual posible.

3. Desarrollo de una herramienta bioinformática que permita obtener rápidamente una primera visión de los diferentes transcriptomas de *Pinus pinaster* que se pueden generar con los mismos datos para poder elegir cuál es el mejor.

4. Desarrollo de una base de datos en la que se almacene la información de modo estructurado y extensible, adaptada a las necesidades del laboratorio y al tipo de datos que se manejan habitualmente. Esta base de datos permitirá analizar el estado actual del transcriptoma de pino.

5. Completar la micromatriz de pino marítimo con nuevos genes procedentes del transcriptoma deducido.

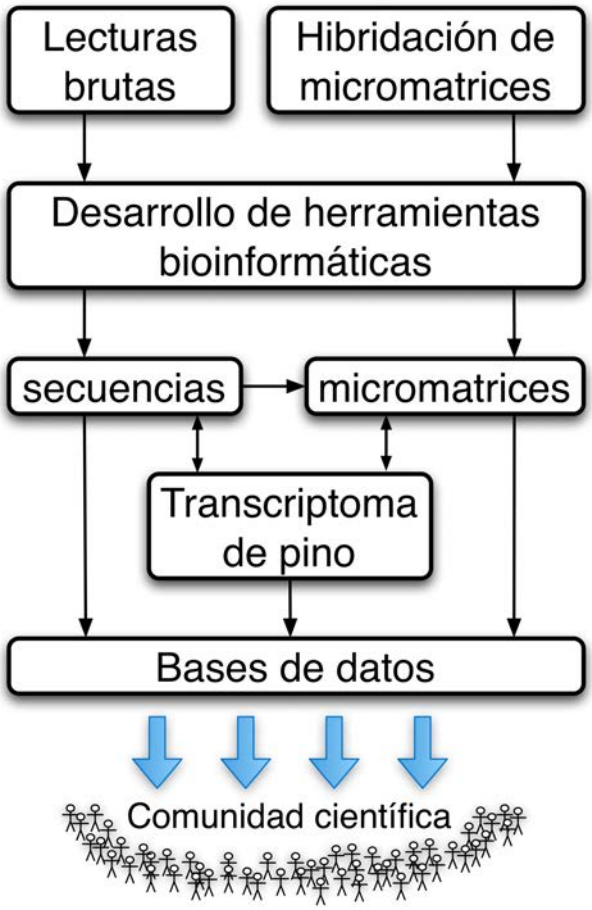


Figura 5.1: *Esquema de los objetivos. Desarrollo de herramientas para el análisis de secuencias y micromatrices. Los resultados obtenidos al analizar las secuencias se aplican en mejorar las micromatrices y en obtener el transcriptoma de pino, y posteriormente la información se integra en bases de datos disponibles para la comunidad científica.*

Parte III

Materiales y métodos

Capítulo 6

Equipos y lenguajes

6.1. Equipos informáticos

Durante el desarrollo de esta tesis se han utilizado varios ordenadores personales y los supercomputadores Picasso SuperDome y Picasso-Cluster, todos ellos con sistema UNIX. Los supercomputadores forman parte de la infraestructura del Centro de Supercomputación y Bioinformática que la Universidad de Málaga tiene en el edificio de Bioinnovación del Parque Tecnológico de Andalucía.

- **Picasso:** se trata de un HP SuperDome con 128 núcleos Intel Itanium2 a 1,6 GHz y con 400 GB de memoria compartida. Ordenador y memoria se encuentran en dos *racks* conectados entre sí. Un tercer *rack* contiene el sistema operativo y el *scratch* (espacio del disco duro para el almacenamiento de datos temporales). El gestor de colas es PBS Pro y el sistema operativo es el Suse Linux Entrerprise Server v. 10.2.5.

- **Picasso-cluster:** Se trata de un *cluster* de 80 núcleos Intel Xeon E5450 a 3,00 GHz (con arquitectura x86) y con 160 GB de RAM, divididos en 10 *blades* con 8 núcleos y 2 GB de RAM para cada uno. Los *blades* están interconectados por red InfiniBand. El gestor de colas es PBS Pro y el sistema operativo es el Suse Linux Entrerprise Server v. 10.2.5.

A continuación se recoge la información de los ordenadores personales utilizados durante es trabajo de tesis

- **iMac G5:** Apple PowerPC a 1,8 GHz, con 512 Mb de RAM. Sistema operativo OS X 10.5

- **MacBook:** ordenador portátil Apple Intel Core 2 Duo a 2 GHz, con 2 GB DDR3 de RAM. Sistemas operativos OS X 10.5 y 10.6

- **iMac:** Core 2 Duo a 3,06 GHz, con 4 GB de RAM. Sistema operativo OS X 10.6

6.2. Lenguajes de programación

Todos los *scripts* y programas desarrollados en este trabajo están escritos en Ruby, Perl o R. También se utilizaron otros lenguajes como UNIX, HTML, Ruby-on-Rails (ROR) y LaTeX. Las ventajas e inconvenientes de algunos de estos lenguajes se describieron en la introducción, apartado 5 (pág. 35).

R: es un lenguaje con entorno de trabajo específicamente desarrollado para realizar análisis estadísticos y su representación gráfica. Se utilizó principalmente para el análisis de micromatrices de dos colores (apartado 9.2.8) y para crear el programa de análisis de micromatrices de dos colores MADE4-2C (apartado 9.2). La página de referencia de este lenguaje es `http://www.r-project.org/`. Una de las grandes ventajas de utilizar R es la gran cantidad de paquetes disponibles para el análisis de datos biológicos. Los paquetes de R son un modo de compartir funciones creadas por otros investigadores. Son de fácil instalación y proporcionan una gran cantidad de nuevas posibilidades, útiles para crear figuras, gráficas, realizar pruebas estadísticas o realizar análisis de datos.

Perl: es un leguaje de *scripting* orientado a objetos ampliamente utilizado y de los más rápidos dentro de los lenguajes interpretados. Se Utilizó para realizar pequeños *scripts*, para el diseño del fichero GAL de Pinarray1 (apartado 9.1) y para preparar y analizar datos relacionados con las pruebas realizadas sobre SeqTrim (apartado 10.1.1). La última versión utilizada al término de este trabajo fue la v5.10.0. La página de referencia de este lenguaje es `http://www.perl.org/`

Ruby: es un leguaje de *scripting* interpretado orientado a objetos, capaz de crear aplicaciones complejas de un modo rápido, manteniendo el código legible y corto [4]. Se ha utilizado para el

desarrollo de la mayoría de los programas que se describen en este manuscrito: SeqTrimNext (apartado 10.1.2), el *script* de descarga de posibles secuencias contaminantes a utilizar con SeqTrimNext (apartado 10.1.2) y el *script* para realizar el informe de la limpieza realizada por SeqTrimNext (apartado 10.1.2), Genote (apartado 10.3.3), FullLengtherNext (apartado 10.2) y el diseño de Pinarray2 (apartado 11.2.3). También se utilizó para manipular secuencias, anotaciones y otros ficheros con grandes cantidades de datos. La última versión utilizada al término de este trabajo fue la 1.9.2. La página de referencia de este lenguaje es `http://www.ruby-lang.org/es/`

Ruby On Rails (ROR): es un entorno de desarrollo de páginas web con base de datos. Se utilizó para el desarrollo de las bases de datos EuroPineDB (apartado 11.1), SustainPineDB (apartado 11.3) y SoleaDB (apartado 11.3.6), así como para realizar los *scripts* de importación y exportación de datos a estas bases de datos. La última versión utilizada al término de este trabajo fue la 2.3.8. La página de referencia de este lenguaje es `http://www.rubyonrails.org.es/`

UNIX OSX 10.6: es un sistema operativo. Sus comandos se utilizaron para la ejecución de programas y para la visualización y manipulación de datos en general.

UNIX SLES 10.2.5: Suse Linux Entrerprise Server se utilizó para la ejecución de programas en el sistema de colas de los supercomputadores Picasso SuperDome y Picasso-cluster, y para la consulta y manipulación de datos en general.

HTML: es el lenguaje utilizado para el desarrollo de páginas web. Se utilizó en las bases de datos EuroPineDB (apartado 11.1), SustainPineDB (apartado 11.3) y SoleaDB (apartado 11.3.6).

LaTeX: es un lenguaje para la composición de textos científicos. Se ha utilizado para crear los informes para SeqTrimNext (apartado 10.1.2) y MADE4-2C (apartado 9.2), y para la realización de este manuscrito. Se utilizó la versión pdfTeX distribuida con TeX Live 2009). La página de referencia de este lenguaje es `http://www.latex-project.org/`

Capítulo 7

Programas informáticos

Durante la realización de este trabajo se ha utilizado una gran variedad de herramientas bioinformáticas. En los siguientes apartados se describe los distintos programas utilizados en los diferentes capítulos de este manuscrito.

7.1. De uso general

7.1.1. *Array-Jobs*

No se trata realmente de un programa, sino de cómo aprovecharse de la ejecución del sistema de colas para ejecutar en paralelo (nunca de forma distribuida) algunos programas que no están pensados en paralelo, siempre que e conjunto de datos de entrada se pueda dividir en conjuntos más pequeños que se pueden analizar de forma independiente. Supongamos que la entrada es un fichero en formato fasta y que el programa analiza cada secuencia una a una de modo independiente (por ejemplo, sería el caso de Blast o de los programas de preprocesamiento y anotación, pero no sería aplicable a programas de ensamblaje). Con el uso de los *array-jobs* se dividirá el fichero de entrada en varios «subficheros» con el mismo formato y un número menor de secuencias, y cada uno de estos «subficheros» se ejecutará independientemente en paralelo. Para dividir el fasta original se puede utilizar el *script* `split_fasta.rb` desarrollado en este trabajo (apéndice A, pág. 197), al que solo hay que pasar dicho fichero original que contiene todas las secuencias y el número de secuencias que va a tener cada uno de los «subficheros» que se crearán, tal como se indica en la siguiente línea de comandos:

```
ruby split_fasta.rb my_file.fasta 500
```

con el que se se crearán «subficheros» de 500 secuencias cada uno.

Para ejecutar el programa en paralelo con cada uno de los «subficheros» se crea un fichero de `bash`, que llamaremos `array_job.sh`, con la siguiente estructura:

```
# numero de cpus:
#PBS -l ncpus=4
# memoria:
#PBS -l mem=8000mb
# tiempo limite:
#PBS -l walltime=100:00:00

#PBS -J 1-20
# para que vaya al directorio actual
cd ${PBS_O_WORKDIR}

# creamos una carpeta nueva por paquete
mkdir pack_$PBS_ARRAY_INDEX

# movemos el fasta a la carpeta nueva
mv my_file_part$PBS_ARRAY_INDEX.fasta
pack_$PBS_ARRAY_INDEX

# nos situamos dentro de la carpeta nueva
cd pack_$PBS_ARRAY_INDEX

# ejecutamos el programa
my_program -input [options]
```

En el fichero del ejemplo se indica que cada uno de los paquetes de secuencias se ejecuta en paralelo utilizando 4 núcleos y 2 Gb de RAM por núcleo (8 000 Mb en total). En caso de que el programa no pueda ejecutarse en paralelo debe utilizarse una CPU únicamente. El número de CPU es conveniente que sea múltiplo del número total de CPU que tiene el supercomputador en el que se ejecuta. Por ejemplo, en Picasso-cluster hay 8 CPU por cada *blade*, lo que limita la ejecución a un máximo de 8 CPU por paquete, así que utilizando 4 CPU podrán ejecutarse 2 paquetes en cada *blade*. Sin embargo, si se eligieran 5 CPU el sistema de colas no tendría más remedio que ejecutar cada paquete en un nodo diferente, dejando 3 CPU libres en cada uno, lo que sería mucho menos eficiente.

En las siguientes líneas del fichero se indica el número de paquetes que contiene el *array-job* (#PBS

-J 1-20), que en nuestro caso será 20, de modo que la variable $PBS_ARRAY_INDEX ira cambiando su valor desde 1 hasta 20. Después se crea una carpeta para cada uno de los paquetes, y se ejecuta el programa dentro de esa carpeta. Es importante ejecutar cada paquete en una carpeta diferente, porque algunos programas crean archivos temporales que al coincidir en un solo directorio, pueden provocar que los más recientes se sobrescriban sobre los más antiguos, perdiendo información imprescindible y provocando errores graves. Una vez configurado a las necesidades, el fichero .sh se envía al sistema de colas con la orden:

```
qsub array_job.sh
```

con lo que todos los paquetes de secuencias se envían al sistema de colas (PBS Pro en el caso de Picasso-cluster) a la vez en un *array-job*, para que se ejecuten simultáneamente, tantos a la vez como permita el sistema de colas.

7.1.2. Textmate

editor de texto muy versátil que se ha utilizado para la realización de todos los programas que se desarrollaron en este marco de este trabajo, incluida la escritura de este manuscrito en formato LaTeX. Se puede descargar de `http://macromates.com/`.

7.1.3. Gemas de Ruby

se trata de paquetes autónomos para el lenguaje de programación Ruby con un formato estándar que proporcionan una funcionalidad estándar (serían el equivalente de los *plugins* en otros programas). Se puede encontrar información detallada de estas gemas en `http://rubygems.org/` y en el caso de las que fueron desarrolladas en el Centro de Supercomputación y Bioinformática de la UMA (SCBI) y la PAB también en `http://www.scbi.uma.es/downloads`. Todas se instalan con el comando:

```
gem install nombre_de_la_gema
```

Las gemas de Ruby y ROR utilizadas en algún momento en este trabajo son:

- **bio (v 1.4.2):** gema necesaria para utilizar Bioruby [87], fue de utilidad para interactuar con las variables disponibles en el interfaz de programación de aplicaciones (Application Programming Interface, API) de la web de KEGG pathway. Este método se utilizó para resaltar las enzimas contenidas en las rutas metabólicas de las secuencias de las bases de datos SustainPineDB y Solea DB. Enviando la información a través del API de KEGG se indica que enzimas de la ruta deben tener el fondo de color amarillo (H. Benzekri, comunicación personal).
- **JSON (v 1.4.6):** sirve para generar y leer ficheros en formato JSON y pasar los objetos a memoria de un modo ordenado.
- **mysql (v 2.7):** para utilizar mySQL en las bases de datos de ROR.
- **rails (v 2.3.5):** para instalar ROR.
- **sqlite3-ruby (v 1.2.4):** para utilizar SQLite3 en las bases de datos de ROR.
- **will_paginate (v 2.3.15):** para paginar en las vistas de ROR.
- **xml-simple (v 1.0.12):** para procesar ficheros en formato xml y pasarlos a objetos en memoria de un modo ordenado.

Las gemas utilizadas en este trabajo se diseñaron específicamente para alguno de los programas que aparecen en este manuscrito, en todos los casos gracias a la colaboración de otros miembros del grupo de investigación, cuyo nombre se especifica en cada una:

- **full_lengther_next (v 0.0.2):** para instalar Full-LengtherNext (D. Guerrero-Fernández y N. Fernández-Pozo).
- **scbi_ace (v 0.0.4):** sirve para indexar los nombres de las secuencias de un fichero en formato ACE y para acceder a todos los datos de un fichero ace en objetos de Ruby (D. Guerrero-Fernández).
- **scbi_blast (v 0.0.28):** se utiliza para la ejecución de Blast+ y el procesamiento de sus resultados sin necesidad de archivos temporales (D. Guerrero-Fernández).
- **scbi_fasta (v 0.1.3):** para leer ficheros fasta, con o sin calidades y pasar nombre y secuencia a objetos de ruby (A. Bocinos).
- **scbi_mapreduce (v 0.0.37):** para ejecutar procesos de modo paralelo o distribuido (D. Guerrero-Fernández).
- **seqtrimnext (v 2.0.39):** para instalar SeqTrimNext (D. Guerrero-Fernández y A. Bocinos).
- **seqtrimnext_report (v 0.0.3):** para instalar la realización de informes en SeqTrimNext (D. Guerrero-Fernández y N. Fernández-Pozo).

- **scbi_plot (v 0.0.6):** para crear gráficos con gnuplot que se utilizan en la gema anterior (D. Guerrero-Fernández).

7.2. Para analizar micromatrices

7.2.1. Bioconductor

Se trata de paquete de R que reúne un conjunto de librerías útiles para el análisis y la comprensión de datos genómicos, de análisis de micromatrices, y de anotación de secuencias. También contiene herramientas para el análisis de lecturas de NGS, pero no se ha utilizado en este sentido. En este trabajo se utilizaron principalmente algunos paquetes de funciones avanzadas diseñadas para análisis de micromatrices. La descripción detallada se encuentra en `http://bioconductor.org/biocLite.R` y en su artículo [82].

Para instalar el paquete básico de Bioconductor, basta ejecutar los siguientes comandos en la consola de R:

```
> source("http://bioconductor.org/biocLite.R")
> biocLite()
```

Para la instalación o actualización posteriores de las librerías de Bioconductor que se han utilizado en este trabajo se desarrolló el *script* textttinstall_update Bioconductor.R

Los paquetes de Bioconductor utilizados en este trabajo son los siguientes:

- **Biobase:** Contiene funciones básicas de Bioconductor necesarias para otros paquetes de R.
- **arrayQuality:** ayuda a evaluar la calidad de las micromatrices. Ofrece varias gráficas e imágenes y medidas estadísticas que sirven para determinar si las hibridaciones son de buena calidad.
- **arrayQualityMetrics:** realiza mediciones de calidad en datos de expresión de micromatrices de cualquier plataforma, de uno o dos colores. Se utiliza para evaluar la calidad del conjunto de micromatrices.
- **convert:** necesario para poder cambiar de formato los diferentes tipos de datos que se utilizan durante el análisis de las micromatrices. Por ejemplo, permite intercambiar los mismos datos entre los diferentes objetos MAList (de `limma`), marrayRaw (de `marray`), marrayNorm (de `marray`) y exprSet (de `Biobase`).
- **graph:** necesario para la creación de algunas gráficas.
- **limma:** su nombre es el acrónimo de *Linear Models for Microarray Data* y contiene un conjunto de funciones para el análisis de micromatrices, como cargar diferentes formatos de datos obtenidos al escanear las micromatrices, corregir el fondo, normalizar los datos y buscar los genes expresados diferencialmente.
- **marray:** paquete para el análisis de datos de micromatrices de dos colores que contiene funciones para la manipulación de los datos, para crear gráficas de diagnóstico, normalizar los datos y verificar la calidad.
- **maSigPro:** es un método basado en la regresión lineal para encontrar los genes que comparten un perfil de expresión génica, que los distingue de otros genes a lo largo de una serie temporal [50].
- **multtest:** método no paramétrico para calcular las tasas de error y los valores estadísticos que indican la fiabilidad de los resultados obtenidos, utilizado para asignar los valores de expresión diferencial de los genes de las micromatrices. Determina el valor ajustado de P de los genes de la micromatriz mediante el método *familywise error rate* (FWER) de Bonferroni, el método de Benjamini y Hochberg para calcular la tasa de positivos falsos (FDR).
- **RankProd:** contiene funciones para el análisis de la expresión diferencial de los genes de micromatrices de expresión [30]. Es un método no paramétrico para identificar los genes expresados diferencialmente (sobreexpresados o subexpresados) basándose en el porcentaje estimado de positivos falsos.
- **Rgraphviz:** paquete para la representar gráficas complejas.
- **vsn:** Otro método de normalización. Realiza la normalización estabilizando la varianza de los datos de las micromatrices.

7.2.2. Librerías generales de R

Otras librerías de R utilizados en este trabajo no forman parte de Bioconductor, y están disponibles en el repositorio gernal cRan `http://cran.es.r-project.org/bin/macosx/universal/contrib/2.6/EMV_1.3.1.tgz`). Son las siguientes:

- **gdata:** contiene varias herramientas de programación para la manipulación de datos.

- **gplots:** contiene herramientas para generar gráficas.
- **gridBase:** paquete para la integración de varias gráficas juntas dispuestas a modo de gradilla.
- **gtools:** contiene varias herramientas de programación.
- **statmod:** paquete para la aplicación de funciones estadísticas.

7.2.3. MADE4-2C

MADE4-2C (***M**icroarray **A**nalysis of **D**ifferential **E**xpression **for Two C**olor Hybridisations*) es un flujo de trabajo para el análisis de micromatrices de dos colores que se ha desarrollado en este trabajo utilizando el lenguaje de programación R y basado en las librerías para el de análisis de micromatrices del paquete *Bioconductor*. A continuación se explicará cómo se ejecuta y configura la ejecución de este *script*, ya que el contenido y los fundamentos se tratarán en los resultados, en el apartado 9.2.

MADE4-2C requiere una serie de ficheros de entrada que aporten la información necesaria para analizar los datos de modo automático. Unos ficheros son obligatorios y otros optativos, que sirven para aportar información extra, aunque no imprescindible. Los ficheros que deben prepararse son:

- `MADE-4-2C_conf.R`: fichero de configuración (disponible en el apéndice D), de uso obligatorio necesario para aportar todas las opciones y parámetros disponibles. En él se especifican, por ejemplo, las veces de cambio (FC) mínimas para considerarlo expresión diferencial, el valor de corte de P, y el método estadístico para ajustar los valores del P, entre Benjamini y Hochberg [23], Bonferroni [27], o Holm-Bonferroni [100]. También se especifica si se desea generar el informe en PDF o se prefiere un análisis rápido en pantalla. Otra información que ha de incluirse es las ruta donde está instalado el programa, y las rutas y nombres de los ficheros donde están los datos y otra información adicional. Dichos ficheros son los que se describen a continuación y suele ser cómodo tenerlos en el mismo nivel que el de configuración.
- `fichero_targets.txt`: fichero donde se indica el diseño experimental, necesario para saber con qué fluoróforo se marcó cada condición experimental en cada micromatriz, cuáles son réplicas técnicas y cuáles las biológicas. En él se especifican también todos los ficheros que contienen los datos de expresión de cada micromatriz en formato GenePix (de extensión `.gpr`) o Spot (de extensión `.spot`).
- `fichero.gal`: contiene toda la información conocida sobre cada sonda impresa en la micromatriz, como mínimo las coordenadas de la sonda, su nombre y las anotaciones.
- `SpotTypes.txt`: fichero opcional para realizar el seguimiento de varias sondas en las imágenes generadas durante el análisis de la micromatriz.
- `BadSpots.txt`: fichero opcional para añadir una lista de sondas de la micromatriz que no se desea que se utilicen en el análisis. Pueden ser desde sondas que se sabe que se imprimieron mal, a controles o genes que no conviene analizar porque no corresponden a lo que se creía inicialmente.
- `ControlSpots.txt`: fichero opcional para añadir una lista de sondas utilizadas como control, o cuya pista se quiere seguir de manera específica. Estas sondas se resaltan en las figuras que muestran los genes expresado diferencialmente.

Para ejecutar el programa, únicamente es necesario llamar al fichero de configuración `MADE-4-2C_conf.R` desde la consola de R con el comando *source*, tal y como se muestra a continuación, pero indicando la ruta en la que se encuentra el fichero:

```
>source("/mi_ruta/MADE-4-2C_conf.R")
```

En la consola de R para MAC OSX, por ejemplo, la ejecución es tan sencilla como hacer doble click en el fichero de configuración `MADE-4-2C_conf.R` para abrirlo y utilizar el atajo de teclado *comando*+E, lo que directamente ejecutará el comando *source* mencionado anteriormente sin necesidad de escribir la ruta en la que se encuentra. Como el fichero de configuración `MADE-4-2C_conf.R` indica donde encontrar los ficheros del programa (MADE4-2C) y los datos del proyecto, es posible tener varios ficheros de configuración con diferentes parámetros para un mismo proyecto. Para una mejor organización, este fichero puede situarse junto con los datos del proyecto o en cualquier otra ruta que se desee.

Una vez ejecutado MADE4-2C, los resultados se organizan las carpetas y ficheros (cuando se ha elegido que el resultado sea en PDF) de la siguiente manera:

- `FinalReport.pdf`: informe donde se explica detalladamente cada etapa, por qué se ha realizado y qué cabría esperar. Es útil para evaluar

la calidad de la hibridación, así como la sustracción del ruido de fondo, la normalización y la detección de genes expresados diferencialmente. Se puede consultar un ejemplo de este fichero en el apéndice B, pág. 199.

- `images`: carpeta con todas las imágenes generadas por el programa que se han incluido, o a las que se hace referencia, en el informe final.
- `Latex`: carpeta con todos los ficheros LaTeX necesarios para generar el informe. Solo resulta útil cuando la compilación del LaTeX produce un error y no se obtiene el PDF.
- `Results`: carpeta con todos los ficheros de resultados en modo texto que se mencionan en el informe:
  - `annotatedPinarray2fatiscan.txt`: en el caso de utilizar el Pinarray1 se devuelve este fichero con anotaciones de todos sus sondas en el formato requerido para utilizar posteriormente el programa FatiScan (apartado 7.2, a continuación).
  - `limma_t_ord.txt`: contiene el valor estadístico de la t de Student ordenado para todas las sondas de la micromatriz, en el formato requerido para utilizar posteriormente FatiScan.
  - Hay una colección de ficheros en formato de `texto tabulado` con los datos de expresión normalizados por los diferentes métodos utilizados (Loess, Loess-Quantile, Loess-Scaled, VSN y VSN-Loess) y con la corrección del fondo. Estos datos pueden resultar útiles para utilizarlos posteriormente con otras librerías de R, como `maSigPro`. Los datos normalizados también se guardan en un formato marrayNorm para ser importado en R para así ahorrarse los pasos de normalización y corrección del ruido de fondo en posteriores análisis.
  - Otra serie de ficheros contienen las listas de los GED en cada uno de los métodos de normalización óptimos que pasó los filtros de calidad de MADE4-2C. Además, llevan la extensión `limma` cuando se ha utilizado este método para obtener los GED, o `rankprod` cuando se utiliza el método Rank Products. También hay otra lista con la intersección de los mismos, como resultado de los GED más fiables.
  - La información referente a los genes expresados diferencialmente se puede encontrar en una serie de ficheros en formato html y texto tabulado con los valores de intensidad media de la expresión, veces de cambio, valor de P y otros valores estadísticos como el valor de P ajustado o la t de *Student*, además de incluir anotaciones de los genes si están disponibles en el fichero GAL (véase el apéndice B).

7.2.4. FatiScan

Se trata de un programa con interfaz web para el análisis funcional de una serie de datos clasificados según un valor estadístico [6]. Resulta útil para analizar las sondas de una micromatriz en función del valor de t obtenido en el análisis, sin descartar ninguna. Con estos datos, FatiScan resalta las funciones (no los genes) que varían en el experimento. FatiScan se puede encontrar en Babelomics 3 `http://babelomics3.bioinfo.cipf.es`, pulsando en las pestañas *Tools*, *Gene Set Enrichment* y *Fatiscan*. Como todos los programas de análisis funcional (apartado 2.5.6), FatiScan dispone de anotaciones para los organismos modelo. Pero a diferencia de otros, cuenta con la posibilidad de realizar el análisis empleando anotaciones propias gracias al método *Your Annotations*. Para eso se requieren dos ficheros, uno con los valores de t de Student ordenados de mayor (positivo) a menor (negativo) para cada una de las sondas de la micromatriz, y otro fichero con anotaciones de cada una de las sondas, preferiblemente términos de la *Gene Ontology*. De esta forma se pueden conocer las funciones moleculares, componentes celulares y los procesos biológicos que destacan en ambas condiciones del experimento en cualquier especie, como el pino.

7.3. Para analizar secuencias

7.3.1. AlignMiner v1.0

En la anotación de las secuencias de EuroPineDB (véase apartado 11.1) se utilizó una versión diseñada para la detección de SNP. El portal donde se puede ejecutar AlignMiner es `http://www.scbi.uma.es/alignminer/` [91], y a continuación se muestra la ejecución del programa por línea de comandos:

```
# inicializamos ruby y gnuplot
. ~ruby/init\_env
. ~gnuplot/init\_env

# ejecutamos el programa
~bioperl/cominer/cominer.rb
~bioperl/cominer/params.txt
my_file.ace
```

7.3.2. AutoFact

programa de anotación de secuencias que puede encontrarse en `http://www.bch.umontreal.ca/Software/AutoFACT.htm`.

AutoFact [126] consulta numerosas bases de datos [UniRef90, UniRef100, NCBI's nr, COG, KEGG, Pfam, Smart, est_others, LSU (Large SubUnit ribosomal RNA), SSU (Small SubUnit ribosomal RNA)] para obtener la mejor descripción del producto del gen, clasificándolas de un modo estructurado, según el resultado encontrado, en:

- Ribosomal RNA
- Functionally annotated protein
- Unassigned protein
- [domain-name] containing protein
- Unknown EST
- Unclassified

En las ejecuciones realizadas en este trabajo se indicó en el fichero de configuración de AutoFact (`AutoFACT.conf`), que el orden de las bases de datos para obtener la información debía ser UniRef90, nr de NCBI, COG, Pfam, KEGG, est_others, es decir, las secuencias solo se anotan con información de EST, si previamente no se ha encontrado ninguna información fiable en el resto de bases de datos. Las bases de datos LSU y SSU no se utilizaron porque ya fueron incluidas en el preprocesamiento realizado con SeqTrimNext.

Si procede, en caso de encontrar similitud con un gen de función conocida (*Functionally annotated protein*) se añaden las enzimas clasificadas según la *Enzyme Commission* y rutas metabólicas de KEGG [116]. Aunque actualmente este programa parece fuera de mantenimiento y es extremadamente lento para utilizarlo con grandes cantidades de secuencias, resulta interesante porque obtiene la descripción de los productos de los genes basándose en una gran cantidad de bases de datos, y devuelve los resultados de un modo estructurado.

Para reducir el tiempo de cálculo de AutoFact se modificó la versión original del programa y se actualizaron las bases de datos (Guerrero et al., resultados no publicados). Para ello se actualizó la versión de BLAST que utilizaba el programa a la versión 2.2.20 de BLASTALL, que permite la paralelización del análisis. Además se modificó la linea de ejecución de BLAST añadiendo un filtro de $E = 10^{-3}$ y poniendo un número máximo de encuentros de 12. Anteriormente, AutoFact ejecutaba BLAST sin limitación, lo que producía hasta 500 descripciones con sus 500 alineamientos por cada secuencia introducida, mientras que con la versión modificada se limitan únicamente a las 12 mejores, en el caso de que mejoren la probabilidad mínima indicada por el valor de E. Así se reduce el tiempo que requiere BLAST para encontrar las secuencias y el tiempo necesario para que AutoFact analice los resultados de BLAST, sin reducir la fiabilidad de éstos.

Para ejecutar los trabajos de AutoFact se dividen las secuencias de entrada en paquetes de 500 secuencias. Cada uno de estos paquetes se ejecuta con BLASTALL en paralelo, utilizando 4 núcleos y 2 Gb de RAM por núcleo, y además se envían al sistema de colas mediante un *array-job* (véase el apartado 7.1.1, pág. 47).

La ejecución de esta versión acelerada de AutoFact se realiza en dos pasos; primero se realizan todas las ejecuciones de BLAST con varios núcleos (en los análisis realizados en este trabajo se utilizaron 4 CPU porque cada nodo tiene 8, y esta cantidad facilita la posibilidad de entrar al sistema de colas, consiguiendo un buen rendimiento de BLAST), y posteriormente AutoFact analiza los resultados de BLAST con un solo núcleo por paquete de secuencias. De este modo, en este segundo paso se ocupan muchos menos núcleos, por lo que el sistema de colas permitirá que se analicen más paquetes a la vez y quedarán más CPU libres para otros usuarios del sistema. A continuación se muestra la línea de ejecución utilizada para este programa:

```
qsub_dep afact_blast.sh afact_parse.sh
```

Al contenido de estos dos ficheros de bash, `afact_blast.sh` y `afact_parse.sh`, se deben añadir delante las líneas de código comentadas anteriormente en los *array-job* (apartado 7.1.1). De este modo, el fichero `afact_blast.sh` continúa tras las líneas indicadas en los *array-job* con:

```
# inicializamos autofact
. ~autofact/init_env_par

# ejecutamos el programa
AutoFACT.pl -f
        my_file_part$PBS_ARRAY_INDEX.fasta
        -a ../AutoFACT.conf -b -c 4
```

de manera que cada paquete de secuencias utiliza 4 núcleos y hay 6 paquetes de secuencias ejecutándose simultáneamente, o sea, que se ocupan 24 núcleos a la vez. Sin embargo, si se utiliza la versión original de AutoFact, sólo se podría utilizar un núcleo.

Al segundo fichero, `afact_blast.sh`, también hay que añadirle las líneas de código indicadas en los *array-job* y seguidamente continuar con:

```
# inicializamos autofact
. ~autofact/init_env_par

# ejecutamos el programa
AutoFACT.pl -f
        my_file_part$PBS_ARRAY_INDEX.fasta
        -a ../AutoFACT.conf -p -c 1
```

En la salida de AutoFact, los ficheros con extensión `.out` contienen las anotaciones recogidas por cada base de datos y la seleccionada como la mejor por el algoritmo de AutoFact. Estos ficheros se utilizan posteriormente para importar las anotaciones de los unigenes a las bases de datos, como se indica en el apartado 11.3.5.

7.3.3. BLAST

Busca regiones similares entre secuencias. Compara secuencias de nucleótidos o proteínas con bases de datos de secuencias de nucleótidos o proteínas y devuelve las secuencias más parecidas en las bases de datos, ordenadas por el valor estadístico *E*, que indica la probabilidad de que el parecido entre las secuencias no se debe al azar. Las secuencias encontradas por BLAST se consideran genes ortólogos a los que contienen nuestras secuencias de entrada. A lo largo de este trabajo se han utilizando muchas versiones de BLAST [13], siendo ncbi-blast-2.2.25+ [34] la última versión utilizada. BLAST se puede descargar desde `ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/`.

Según el tipo de secuencias de entrada y la base de datos que haya que utilizar se ejecutará un BLAST diferente. Durante la realización de este trabajo se utilizaron habitualmente:

- BLASTN: para comparar secuencias nucleotídicas con bases de datos de secuencias de nucleótidos. Ej: utilizado en SeqTrim, SeqTrimNext, Genote, AutoFact, EuroPineDB y SustainPineDB.
- BLASTN: para comparar secuencias de proteínas con otras secuencias de proteínas. Como en este trabajo se trabaja principalmente con secuencias de nucleótidos de partida, este es el tipo de BLAST fue menos utilizado, pero fue bastante útil para realizar pruebas con las secuencias de proteínas de salida de FULL-LENGTHERNEXT.
- BLASTX: para comparar secuencias nucleotídicas con una base de datos de secuencias de proteínas. Ej: utilizado en FULL-LENGTHERNEXT, Genote, Blast2GO, AutoFact, EuroPineDB y SustainPineDB.
- TBLASTN: para comparar secuencias de proteínas con una base de datos de secuencias de nucleótidos (esta base de datos se traduce a proteínas para llevar a cabo la comparación). Ej: es posible utilizarlo en la página web de EuroPineDB y SustainPineDB.

7.3.4. Blast2GO

programa de anotación con interfaz gráfica, fácil de utilizar e instalar. Útil para la asignación de términos de la *Gene Ontology* [17], códigos de familias de proteínas de InterPro [107], el código de la *Enzyme Commission* en caso de que la proteína sea una enzima, y ruta de *KEGG pathways* [116] en caso de que intervenga en alguna de las rutas recogidas en este repositorio. Blast2GO resulta lento en caso de analizar una gran cantidad de secuencias, ya que la mayor parte de sus análisis se realizan de forma remota para evitar la instalación de las bases de datos, y solo es capaz de emplear un núcleo. A lo largo de este trabajo se han utilizado muchas versiones de Blast2GO [49], la última fue la v2.5.0. Blast2GO está disponible en: `http://www.blast2go.com/b2glaunch`.

La realización de anotaciones con Blast2GO se divide en varias etapas muy sencillas que en la interfaz gráfica van marcando nuestras secuencias con diferentes colores según se superan estas etapas (tabla 7.1). Al empezar, las secuencias aparecen en fondo blanco y se ejecuta el primer paso de BLAST con valor de $E = 10^{-6}$ frente a una base de datos de proteínas. Las secuencias que encuentran similitud aparecen en fondo naranja y las que no en fondo rojo (tabla 7.1). El siguiente paso es el mapeo, en el que el programa se encarga de consultar en las bases de datos qué códigos de la GO están asociados a nuestras secuencias y cambia el fondo a verde. Después se ejecuta la función de anotación, en la que se filtran los términos GO encontrados según su procedencia, marcando las secuencias con fondo azul. Para este paso se utilizaron los valores mostrados en la tabla 7.2, que son algo más restrictivos que los valores por omisión y aseguran que sólo pasen el filtro los términos GO más fiables (se valoran mucho más los anotados manualmente con respecto a los anotados automáticamente). Por último se lanza la anotación con códigos InterPro (no tienen color asociado).

Con el fin de obtener las anotaciones para su importación en las bases de datos, una vez se han ejecutado todas las etapas se exportan los datos en formato de texto tabulado (realizando los siguientes los pasos en la barra del menú de Blast2GO: File>Export>Export Sequence Table). Sin embargo, si se realiza la exportación de los datos de to-

Tabla 7.2: *valores utilizados en Blast2GO para filtrar los códigos de la* Gene Ontology *según su procedencia.*

Name	Definition	Value	Category
EXP	Inferred from Experiment	1.0	
IDA	Inferred from Direct Assay	1.0	
IPI	Inferred from Physical Interaction	1.0	Experimental Evidence Codes
IMP	Inferred from Mutant Phenotype	1.0	
IGI	Inferred from Genetic Interaction	1.0	
IEP	Inferred from Expression Pattern	1.0	
ISS	Inferred from Sequence or Structural Similarity	0.5	
ISO	Inferred from Sequence Orthology	0.5	
ISA	Inferred from Sequence Alignment	0.5	Computational Evidence Codes
ISM	Inferred from Sequence Model	0.5	
IGC	Inferred from Genetic Context	0.3	
RCA	Inferred from Reviewed Computational Analysis	0.5	
TAS	Traceable Author Statement	0.5	Author Statement
NAS	Non-Traceable Author Statement	0.4	
IC	Inferred by Curator	0.5	Curator Statement
ND	No biological Data available	0.0	
IEA	Inferred from Electronic Annotation	0.0	Automatically-Assigned
NR	Not Recorded	0.0	Obsolete Evidence Codes

Tabla 7.1: *Colores de las etapas de Blast2GO realizadas.*

Etapa	Status	Color
Sin analizar	Sin analizar	Blanco
BLAST	Similitud con BLAST	Naranja
BLAST	Sin simil. en BLAST	Rojo
Mapping	Con términos GO	Verde
Annotation	Con GOs validados	Azul

das las secuencias conjuntamente, los términos GO filtrados por Blast2GO (con fondo azul), considerados fiables se exportan en el mismo fichero que los GO que no han sido filtrados (con fondo verde), sin poder diferenciar que unigenes tienen GO fiables y cuales no. Para solucionar esto, la exportación se realiza en dos pasos. (1) Se ordenan las secuencias por color, se marcan únicamente las azules (con GO fiables) y se exportan en un fichero al que llamaremos `my_project_b2go_reliable.txt` (realizando los siguientes los pasos en la barra del menú de Blast2GO: Select>Select by Color>blue (b2g-annotated)). (2) Se marcan el resto de secuencias (realizando los siguientes los pasos en la barra del menú de Blast2GO: Select>Invert Selection) y se exportan en otro fichero de texto tabulado, `my_project_b2go_sin_gos.txt`, que sabemos que contiene términos GO que no son fiables. Estos ficheros se utilizan posteriormente para importar las anotaciones de los unigenes a las bases de datos, como se indica en el apartado 11.3.5.

7.3.5. Bowtie 2

Se trata de una herramienta para el alineamiento de lecturas de 200-600 pb sobre secuencias largas de referencia, que pueden provenir del genoma o de los unigenes del transcriptoma de un organismo [129]. Se eligió este programa porque dispone de una buena documentación (`http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml`), tiene una gran cantidad de parámetros ajustables para su ejecución, es capaz de trabajar en paralelo, y sobre todo porque es muy rápido. A continuación se describe brevemente cómo se ha utilizado Bowtie2 en este trabajo, teniendo en cuenta que se ejecuta en dos pasos. En en primer paso, a partir de las secuencias de referencia guardadas en el fichero multifasta `my_reference.fasta`, crea los índices de la referencia en el fichero `my_indexes` del siguiente modo:

```
bowtie2-build -f my_reference.fasta
              my_indexes
```

En la siguiente etapa se emplea el fichero `my_indexes` para alinear las lecturas contenidas en el fichero `my_reads_1.fastq` a la referencia:

```
bowtie2 my_indexes -q -U my_reads_1.fastq
        -p 4 -S my_align.sam
```

donde `-q` indica que el fichero de lecturas está en formato FASTQ [46], `-U` indica que lo que viene a continuación es el nombre del fichero o ficheros (separados por comas) con las lecturas, `-p` indica

el número de CPU a utilizar, y `-S` el nombre del fichero de salida con los alineamientos en formato SAM [134]. Puesto que el análisis con Bowtie2 se realizó únicamente para estimar de forma rápida la distribución de las lecturas en los unigenes del transcriptoma se utilizaron sus parámetros por defecto. Estos parámetros incluyen el modo de alineamiento *end-to-end*, es decir que las lecturas alineen desde un extremo al otro con la referencia y no se ignore si los extremos de la lectura no alinean, el modo *sensitive*, que es el tercero más sensible de cuatro modos y el segundo menos rápido (los modos disponibles son *very fast*, *fast*, *sensitive* y *very sensitive*), no se permiten errores de alineamiento (*mismatches*) y cada gap penaliza -5 además de -3 por cada nucleótido que abarque la longitud del gap.

7.3.6. CAP3

Programa de ensamblaje de secuencias de tipo OLC, basado en el solapamiento entre los extremos de las secuencias [105]. En los estudios de transcriptómica se considera como uno de los ensambladores más fiables para secuencias de tipo Sanger [140] y también para re-ensamblar varios ensamblajes procedentes de un mismo conjunto de secuencias para conseguir un resultado que se ajuste más a la realidad [128, 238]. Se puede descargar en `http://seq.cs.iastate.edu/cap3.html`, o ejecutar on-line en la web del SCBI-PAB, `http://www.scbi.uma.es/cap3`. Este programa permite ajustar numerosos parámetros que en la versión web se encuentran rellenos con los valores por omisión para ensamblar EST; si se fueran a ensamblar secuencias genómicas o reensamblar unigenes de transcriptoma habría que incrementar el valor de identidad al 90 %. Por línea de comandos sólo hay que indicarle el fichero fasta y los parámetros que varíen entre los valores por omisión:

```
cap3 my_file.fasta -p identity_value
```

CAP3 proporciona varios ficheros de salida: el fichero con extensión .ace que contiene la información del ensamblaje, un fichero con extensión .contig, donde se encuentra la secuencia consenso de los contigs formados en formato fasta, y otro fichero con extensión .singlets, donde se incluyen las secuencias de los singulones en formato fasta. En caso de redirigir la salida en pantalla a un fichero, se obtendrían los alineamientos del ensamblaje en un fichero de texto.

7.3.7. CD-HIT

Programa para el agrupamiento de secuencias, tanto de proteínas como de nucleótidos [138], disponible en `http://cd-hit.org`. Es de gran utilidad para reducir la redundancia de un grupo de secuencias, ya que agrupa las secuencias según los criterios indicados y devuelve una secuencia consenso por cada conjunto de secuencias que se parecen entre sí.

Ejemplo de ejecución para agrupar secuencias de nucleótidos:

```
cd-hit-est -i input.fasta -o output.fasta
          -c 0.9 -G 0 -n 10 -aS 0.1 -A 40
          -T 4 -M 6000
```

donde `-i` y `-o` indican los ficheros de entrada y salida respectivamente, `-c` el porcentaje mínimo de similitud entre las secuencias que se agrupan, `-G 0` indica que para los cálculos de similitud solo se utilicen las partes que alinean, `-n` es la ventana de nucleótidos o aminoácidos que utiliza el programa en la comparación (en inglés *word size*), `-aS` indica la cobertura mínima que supone el solapamiento para la secuencia más pequeña, `-A` es el porcentaje mínimo de alineamiento, `-T` indica el número de CPU utilizadas, y `-M` la memoria RAM utilizada.

Para más información acerca de como ejecutar el programa y de sus parámetro es recomendable consultar su manual (`http://www.bioinformatics.org/cd-hit/cd-hit-user-guide.pdf`).

7.3.8. Euler-SR

Sirve para ensamblar secuencias basándose en los grafos de De Bruijn y los caminos eulerianos [175]. Para las secuencias transcriptómicas de pino era el ensamblador de tipo De Bruijn que mejores resultados obtenía (R. Bautista, comunicación personal). Es fácil de ejecutar, basta con indicar el fichero de entrada y el valor de k-mero que se quiere utilizar para obtener el resultado en el fichero `my_results_file.txt`:

```
Assemble.pl my_file.fasta kmer_value >
           my_results_file.txt
```

7.3.9. Full-Lengther

Sirve para determinar si una EST se obtuvo a partir de un clon de ADNc que contiene todo el gen [130]. Solo estaba disponible vía web: `http://www.scbi.uma.es/cgi-bin/full-lengther/full-lengther_login.cgi`, donde tan solo requería un fichero con las secuencias (aceptaba varios formatos: fasta, phd, cromatogramas y ficheros comprimidos zip con cromatogramas), el valor de E mínimo para considerar como fiable un ortólogo, y el número máximo de aminoácidos que pueden faltar al principio del ortólogo. Aunque la idea original seguía siendo válida

para las lecturas obtenidas por los secuenciadores de nueva generación, el algoritmo para deducir si se había identificado el gen completo no lo era. Por eso se desarrolló FULL-LENGTHERNEXT, un programa nuevo adaptado a NGS y con más aplicaciones (apartado 10.2, pág. 105).

7.3.10. MIRA3

Se trata de un programa de ensamblaje de tipo OLC, es decir, basado en el solapamiento de los extremos de las secuencias, como CAP3. Es la evolución de miraEST [44] para adaptarse a secuencias de Roche-454 y otras secuencias de NGS, según se indica en su página web oficial (`http://sourceforge.net/apps/mediawiki/mira-assembler/index.php`). Se puede descargar de `http://sourceforge.net/projects/mira-assembler/files/` y son muchos los parámetros que permite ajustar para cada ejecución; por eso dispone de una buena documentación (`http://mira-assembler.sourceforge.net/docs/DefinitiveGuideToMIRA.pdf`). Además, hay disponible una versión web del programa en el SCBI-PAB (`http://www.scbi.uma.es/mira`), y también se puede ejecutar por línea de comandos utilizando el sistema de colas de Picasso (véase el apartado 6.1). A continuación se muestra un ejemplo del fichero de bash necesario para la ejecución de MIRA3 en el sistema de colas. En este ejemplo se combinan secuencias de tipo Sanger y de 454 para ser ensambladas juntas:

```
# numero de cpus:
#PBS -l select=ncpus=8
# memoria:
#PBS -l select=mem=16000mb
# tiempo limite (h:min:seg):
#PBS -l walltime=160:00:00

# para que indicar el directorio de trabajo
cd $PBS_O_WORKDIR

# se inicializa mira en picasso
. ~mira/init_env_dev

# se crean variables
NAME=my_project_name
LOG_DIR=/drives/scratch1/users/noefp/$$/

# se crea el directorio de salida
echo LOG_DIR
mkdir -p ${LOG_DIR}

# se ejecuta el programa
mira -fasta -project='my_project'
     --job=denovo,est,normal,sanger,454
     -CL:ascdc
     454_SETTINGS -CO:fnicpst=yes
     -notraceinfo
     SANGER_SETTINGS -LR:wqf=no
     -AS:epoq=no -AS:bdq=22
     -CO:fnicpst=yes
     COMMON_SETTINGS -GE:not=4
     -DI:lrt=${LOG_DIR}
     > my_project_output.txt
```

MIRA3 solo fue posible ejecutarlo en Picasso porque es el único ordenador con memoria compartida y, por lo tanto, el único capaz de satisfacer los requisitos de RAM para su ejecución; además, Picasso tampoco tenía el límite de tiempo de ejecución restringido a 3 días, por lo que era posible mantenerlo en funcionamiento durante los días necesarios. Los ficheros de entrada deben nombrarse como `my_project_in.454.fasta` o `my_project_in.sanger.fasta`. A continuación se describen los parámetros utilizados para realizar el ensamblaje mixto con secuencias de tipo Sanger y de 454 del ejemplo anterior:

- `-fasta` indica el formato del fichero de entrada. Otros formatos posibles son fastaq, phd y caf.
- `-project` indica el nombre del proyecto.
- `-job` indica el tipo de ensamblaje. Las posibles opciones son denovo|mapping, genome|est, draft|accurate, sanger|454|iontor|solexa.
- `-CL:ascdc` parámetro para evitar quimeras.
- `454_SETTINGS` indica que a continuación se van a especificar los parámetros para las secuencias de 454 (`-CO:fnicpst=yes`, `-notraceinfo`).
- `-CO:fnicpst=yes` para evitar la aparición de nucleótidos degenerados, que pueden dar problemas en pasos posteriores si el programa se ejecuta en un flujo de trabajo automatizado (*pipeline*). A pesar de ello, siempre aparecen cuando se realiza un ensamblaje mixto con secuencias de 454 y de tipo Sanger, en cuyo caso los contigs mixtos podrán contener nucleótidos degenerados.
- `-notraceinfo` para no incluir el fichero XML que indica cómo se recortan las secuencias de 454.
- `SANGER_SETTINGS` indica que a continuación se van a especificar los parámetros para las secuencias de tipo Sanger (`-LR:wqf=no`, `-AS:epoq=no`, `-AS:bdq=22`, `-CO:fnicpst=yes`).

- `-LR:wqf=no` para no añadir fichero de calidades al ensamblaje.
- `-AS:epoq=no` para no forzar que se utilicen valores de calidad, es decir, evitar que MIRA3 detenga su ejecución si alguna secuencia carece de valores de calidad.
- `-AS:bdq=22` para poner todos los valores de calidad (QV) en 22
- `-CO:fnicpst=yes` para que no se devuelvan consensos con nucleótidos degenerados.
- `COMMON_SETTINGS` indica que a continuación se van a especificar los parámetros generales (`-GE:not=4`).
- `-GE:not=4` indica que el número de núcleos que utilizará será de 4.
- `-DI:lrt=`LOG_DIR `>my_project_result.txt` indica el directorio de salida del log.
- `>` `my_project_output.txt` para redirigir la salida a pantalla hacia un fichero. En algunos casos, los errores pueden recogerse en este fichero, en lugar de en el fichero de errores del sistema de colas.

Este programa devuelve como resultado varios directorios con ficheros de salida, de los que los más importantes para este trabajo han sido:

- `my_project_d_info`: Directorio que contiene los ficheros:
  - `my_project_info_assembly.txt`: contiene un resumen de lo acontecido en el ensamblaje.
  - `my_project_debrislist.txt`: recoge la lista de las lecturas descartadas.
- `my_project_d_results`: Directorio que contiene los ficheros:
  - `my_project_unpadded.fasta`: contiene los unigenes en formato fasta, desprovistos de los huecos (*gaps*) que pueden formarse al obtener la secuencia consenso de varias lecturas que al alinearse contienen indels.
  - `my_project_padded.fasta`: contiene los unigenes en formato fasta marcando en su secuencia con asteriscos la presencia de huecos.
  - `my_project_out.ace`: el ensamblaje en formato ACE.

Para más información consúltese el manual del programa, mencionado al comienzo de este apartado.

7.3.11. MREPS

Se trata de un pprograma para la detección de repeticiones de secuencias simples (SSR) a partir de un fichero en formato fasta [124]. Disponible en: `http://bioinfo.lifl.fr/mreps//`, la versión que se a utilizado es la 2.5. Se ejecuta con la orden

```
mreps.linux.bin -minsize 12 -minperiod 2
                -exp 3.0
                -fasta my_file.fasta
```

donde `-minsize` indica la longitud mínima de la repetición encontrada, `-minperiod` el número mínimo de nucleótidos que se repiten (2:GAGA 3:CATCAT 4:GATAGATA), y `-exp` el número mínimo de veces que se repite (2:TATA 3:TATATA 4:TATATATA).

7.3.12. SeqTrim

Se trata de una herramienta desarrollada en este trabajo para el preprocesamiento de secuencias, principalmente las de tipo Sanger empleando un solo núcleo de procesador (figura 7.1-A). A lo largo de este trabajo han sido muchas las versiones utilizadas de SeqTrim, la última versión utilizada al término de este trabajo fue la v0.111. SeqTrim se puede descargar de `http://www.scbi.uma.es/bio/soft/seqtrim/downloads/seqtrim.zip`. Para más información, véase el artículo incluido en este manuscrito, en el apartado 10.1.1, página 83. Se puede ejecutar en modo web en `http://www.scbi.uma.es/cgi-bin/seqtrim/seqtrim_login.cgi`, donde hay que indicar los parámetros necesarios y el orden en el que se ejecutan las diferentes etapas de preprocesamiento, que por omisión son:

1. Eliminación de vector, poli-A/T, adaptadores.
2. Eliminación de indeterminaciones (Ns).
3. Eliminación de secuencias de baja calidad.
4. Eliminación de secuencias contaminantes.

A continuación se muestra un ejemplo de la ejecución de SeqTrim por línea de comandos:

```
~bioperl/seqtrim/seqtrim.pl -Cv
     --adaptorSeqLeft="6"
     --adaptorSeqRight="7"
     --seqMinLength=100
```

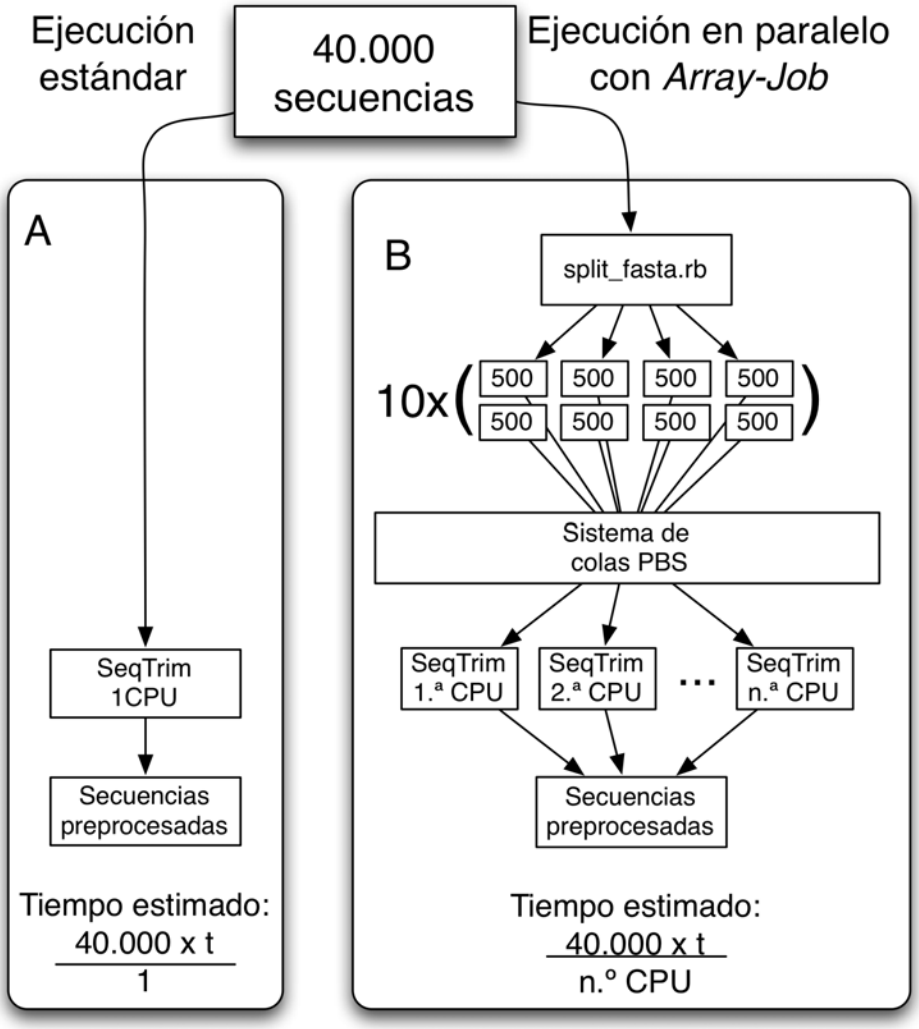


Figura 7.1: *Ejecución de SeqTrim de manera estándar (A) y utilizando un* array-job *en el sistema de colas (B). Para un tiempo t, estimado para el procesamiento de una secuencia, la ejecución con* array-job *tardará tantas veces menos como CPU se utilicen en la ejecución.*

```
--upstreamRestrictSite="-"
--downstreamRestrictSite="-"
-f pin_originalf.fasta
-q pin_originalq.qual
-o pin_cleaned.fasta > screen.txt
```

donde `-Cv` muestra los resultados en pantalla en color (`-C`) y de un modo verboso (`-v`), es decir, mostrando todos los mensajes. Es posible indicar los adaptadores utilizados en el vector de clonación y el sitio de restricción esperado, para ello basta con elegir el número que los representa en la lista mostrada en la ayuda de SeqTrim (`-h`). Con `-f` y `-q` se indica el fichero fasta de entrada y las calidades correspondientes. Con `-o` se indica el nombre del fichero de salida que contiene las secuencias del inserto desprovistas de elementos que no contienen información biológica. El símbolo > redirige la salida impresa en pantalla a un fichero. No se ha declarado en qué orden se quieren ejecutar los módulos de SeqTrim, lo que indica que se empleará el orden por defecto. Para más información acerca de las opciones que permite utilizar SeqTrim ejecútese la ayuda del programa con `-h`.

Cuando se requiere analizar decenas de miles de secuencias, como en el caso de cualquier librería de 454 (véase, por ejemplo, el apartado 11.1, pág. 139 de este trabajo), la ejecución de SeqTrim se puede realizar en paralelo, utilizando un *array-job* (apartado 7.1.1, pág. 47) para acelerar el proceso, como se muestra en la figura 7.1-B. Para ello se fragmenta el fichero de entrada en varios ficheros con 500 secuencias cada uno. Como SeqTrim no admite su ejecución con más de una CPU, en el *array-job* se selecciona una CPU únicamente, pero en el sistema de colas se analizan simultáneamente varios de

los ficheros de 500 secuencias a la vez, tantos como permita el sistema de colas. De esta manera, si se considera que se tarda 1 s en preprocesar cada secuencia ($t = 1$) y que se distribuye el trabajo en 20 «subficheros» (con lo que se podrán utilizar simultáneamente hasta 20 núcleos), la ejecución utilizando un *array-job* podría llegar a ser 20 veces más rápida que la ejecución estándar (figura 7.1-A) puesto que tardaría 40 000 segundos en 7.1-A y 2000 segundos en 7.1-B ($40\,000 \times 1/20 = 2000$).

7.3.13. SeqTrimNext

Se trata una herramienta para el preprocesamiento de secuencias de nueva generación desarrollada para este trabajo. Se puede utilizar por línea de comandos, como servicio web basado en REST y como herramienta web. Para más información véase su portal `http://www.scbi.uma.es/seqtrimnext` y el artículo incluido en este manuscrito (apartado 10.1.2, página 97).

A continuación se muestra un ejemplo de la ejecución de SeqTrimNext a través del sistema de colas de Picasso-Cluster (véase el apartado 6.1):

```
# se inicializa SeqTrimNext en el Cluster
. ~seqtrimnext/init_env

# se recogen las CPU asignadas por el PBS
cat ${PBS_NODEFILE} > workers

seqtrimnext -t my_template.txt -w workers
            -f my_seqs.fasta -q my_seqs.qual
```

en la que `-t` indica que el fichero `my_template.txt` es una plantilla en la que se detallan los parámetros que se desean utilizar —en la versión web del programa se ofrecen plantillas con valores por omisión para varias situaciones (genómica, transcriptómica, amplicones, secuencias de plantas, etc.)—. El parámetro `-w` indica que el programa se ejecutará en los núcleos con las ID que se recogen en el fichero `workers`. Con `-f` y `-q` se indican los nombres de los ficheros con las secuencias y las calidades (conviene consultar la ayuda del programa para ver más opciones y formatos de entrada).

7.3.14. Tablet

Se trata de una herramienta gráfica interactiva para la visualización de ensamblajes. Admite multitud de formatos diferentes: ACE, AFG, MAQ, SOAP2, SAM, BAM, FASTA, FASTQ y GFF3. Puede descargarse en `http://bioinf.scri.ac.uk/tablet/` y es muy sencillo de utilizar, tan solo hay que abrir el programa y elegir el fichero de ensamblaje que se quiere visualizar.

Capítulo 8

Datos biológicos

Los datos biológicos sobre los que se aplicaron los distintos programas y con los que se construyeron las bases de datos proceden en su mayoría del grupo de investigación Biología Molecular y Biotecnología de Plantas (BMBP, BIO-114), pero también se utilizaron datos de los repositorios internaciones. Se indicarán con detalle en cada uno de los apartados.

8.1. Micromatrices de pino

Se han utilizado tres micromatrices de pino: Pinarray1, Pinarray2 y SSH-Ma. Pinarray1 es una micromatriz de ADNc preparada en nuestro grupo de investigación por D. Pacheco-Villalobos, S. Díaz-Moreno y F.R. Cantón con secuencias impresas de tres especies de pino: *Pinus pinaster*, *Pinus sylvestris* *Pinus pinea* [35, 169]. Estas secuencias proceden principalmente de una selección de clones de tres genotecas de ADNc Gemini [38], CK16 [12] y Pin (para más información, véase el apartado 11.1, pág. 139 de este trabajo). Pinarray1 contiene 3456 puntos; 2800 con secuencias de Gemini, 201 de Pin, 345 de CK16, 8 con genes del metabolismo del nitrógeno procedentes de *pinaster*, *sylvestris* y *taeda*, 62 con DMSO al 50 % (control negativo de hibridación) y 40 con secuencias de control, entre las que se incluyen secuencias cloroplastídicas, desmina, nebulina, varios cebadores corrientes, los plásmidos pGEMT y pBSK, ADN genómico de pino y bacteriano, y los *spikes*. Los 3456 puntos se imprimen dos veces en cada micromatriz, de modo que al hibridar, en cada micromatriz se dispone de una réplica técnica. Posteriormente, cuando se analizan estas micromatrices se separan en dos set de datos mediante un *script* de Perl (M.G. Claros, datos no mostrados), a los que se les asigna el sufijo `-A` y `-Z` respectivamente, y se indican como réplicas técnicas en el diseño. El Pinarray1 se utilizó para la comparación de diferentes grupos de genes de pino expresados en diferentes condiciones, que se especifican en el siguiente apartado.

La micromatriz Pinarray2 se desarrolló en este trabajo añadiendo a Pinarray1 los clones nuevos que se identificaron en las genotecas que se describen en el artículo de EuroPineDB (apartado 11.1, pág. 139). Para más información, véase el apartado 11.2.3, pág. 154, de este trabajo.

La micromatriz SSH-Ma fue desarrollada por D. Pacheco, S. Díaz Moreno y F.R. Cantón [58, 226] a partir de unos 4000 clones de secuencia desconocida procedentes de genotecas de xilema en formación [58]. Estas genotecas fueron creadas mediante el método de hibridación sustractiva por supresión (SSH). Las muestras de RNA empleadas como *tester* y *driver* en cada substracción se obtuvieron siempre de un mismo individuo de *P. pinaster* procedente de la estación forestal de INRA-Pierroton. Las poblaciones de RNA substraídas correspondieron a muestras de xilema en formación juvenil frente a maduro y viceversa, además de xilema en formación de compresión frente a opuesto y viceversa. El uso de la micromatriz SSH-Ma puede encontrarse en los apartados 9.2.8 y 9.3. Esta micromatriz también tenía impresos dos juegos de sondas, por lo que todas las hibridaciones resultants también se dividieron informáticamente en dos réplicas técnicas con el mismo *script* que se utilizó para el Pinarray1.

8.2. Para micromatrices

Durante el desarrollo de MADE4-2C (apartado 9.2) se utilizaron diferentes muestras de datos de micromatrices de dos colores procedentes de nuestro grupo de investigación:

- **Madera madura frente a madera juvenil:** datos de madera juvenil y madura de *Pinus pinaster*, cedidos por S. Díaz Moreno y F. R. Cantón.
- **Madera de compresión frente a madera lateral:** datos de *Pinus pinaster* formando madera de compresión y madera lateral, cedidos por D. Pacheco y F. R. Cantón.

- **Estrés por exceso y ausencia de amonio:** Se analizaron datos de micromatrices procedentes del ápice y la base de plántulas de pino sometidas a diferentes concentraciones de amonio [35].
- **Estudio de dos líneas transgénicas:** Se analizó la expresión en acículas procedentes de dos líneas transgénicas 1 y 18, de *Pinus pinaster*, que sobreexpresan la glutamina sintetasa a (GS1a), frente a acículas procedentes de un conjunto de individuos control. Datos cedidos por J. Canales y F. M. Cánovas.
- **Análisis de varias combinaciones con presencia y ausencia de nitrógeno y carbono:** datos de pinos sometidos a diferentes combinaciones de carencia de carbono y nitrógeno, cedidos por J. Canales, M Rueda, C. Ávila y F. M. Cánovas.

Los datos de expresión utilizados que proceden de otros grupos de investigación o repositorios son:

- **Swirl zebrafish microarray experiment** que viene incluido en la librería `marray` de Bioconductor.
- Dos experimentos realizados con *Sinorhizobium meliloti* de Jose Antonio López Contreras y Manuel Fernández, de la Estación Experimental del Zaidín-CSIC (Granada).
- Experimentos realizados por María del Carmen Blanes del departamento de Biología Animal, Biología Vegetal y Ecología de la Universidad de Jaén, en el que se hibridaron secuencias de *Abies pinsapo* en el Pinarray1.
- Experimentos de *Pinus halepensis* sometidos a estrés hídrico durante 14, 28 y 35 días e hibridados frente a un grupo control en el Pinarray1, realizado en colaboración con el profesor Rafael Navarro del departamento de ingeniería forestal de la Universidad de Córdoba.

8.3. Para el transcriptoma de pino

Además de las genotecas que se encuentran detalladas en los materiales y métodos del artículo sobre EuroPineDB (apartado 11.1, pág. 139), también se ha utilizado estos otros datos:

- **EPDB2:** Unión de las secuencias sólo de *Pinus pinaster* de EuroPineDB (apartado 11.1). Incluye secuencias procedentes de secuenciadores automáticos de tipo Sanger y 454, formada por 944 742 lecturas, 913 786 lecturas brutas de 454, y 30 956 secuencias preprocesadas procedentes de clones de ADNc secuenciados con el método Sanger.
- **Biogeco_1:** librería de lecturas de 454 Titanium, formada por un *pool* de tejidos, con xilema en diferenciación, yemas y acículas procedentes de 6 genotipos de *Pinus pinaster*. Cedidas por C. Plomion (INRA Pierroton), contiene 1 571 741 lecturas.
- **Biogeco_2:** librería de lecturas de 454 Titanium, formada por EST procedentes de yemas en reposo de individuos de 2 años de *Pinus pinaster* procedentes de 2 grupos: sometidos a estrés por sequía y sin deficiencia de agua. Cedidas por C. Plomion (INRA Pierroton), contiene 768 224 lecturas.
- **UAGPF:** librería de embriones somáticos formada por lecturas de Roche 454 de *Pinus pinaster*. Cedidas por P. Label (INRA Orleans), contiene 990 405 lecturas.

Parte IV

Resultados y discusión

Capítulo 9

Análisis de la expresión génica con micromatrices

La existencia del Pinarray1 (véase el apartado 8.1 de Materiales y métodos), disponible para los investigadores del grupo de Biología Molecular y Biotecnología de plantas, ha hecho que se hayan planteado una serie de experimentos de expresión génica en *Pinus pinaster*, para los que convenía tener un sistema de análisis homogéneo que permitiera, además de localizar los GED, comparar los resultados obtenidos en los distintos experimentos. Por eso, lo primero fue anotar las secuencias que contenía el Pinarray1 y a continuación se preparó la herramienta de análisis conocida como MADE4-2C, cuya aplicación se explicará posteriormente sobre algunos ejemplos.

9.1. Anotación del fichero GAL de Pinarray1

En el fichero GAL del Pinarray1, fichero de texto tabulado generado por el *software* informático del robot de impresión, se encuentran las coordenadas en las que se han imprimido cada una de las secuencias. Los clones de ADNc que se imprimieron ya estaban secuenciados antes del comienzo de este trabajo, por lo que para preprocesar y anotar las secuencias se partió de los cromatogramas o de los números de acceso del GenBank. Unas y otras se preprocesaron con SeqTrim [70] y se anotaron con Blast2GO [49], con lo que se obtuvo la descripción del producto del clon, los términos de la *Gene Ontology*, los códigos de la *Enzimme Commission* y las rutas metabólicas de *KEGG pathways* asociadas a estas secuencias. A estas anotaciones se les añadieron las coordenadas de las placas de 96 en las que se almacenaban los clones, y las de 384 en las que se reordenaron para imprimirse en el Pinarray1. También se incorporaron los números de acceso de las secuencias en GenBank y, por último, para cada clon se añadió el número de acceso del *Tentative Consensus* (TC) más parecido en el Pine Gene Index (PGI) [180]. Para obtener los TC se descargó la última versión del PGI, la 8.0 en aquel momento, y con sus secuencias se creó una base de datos de nucleótidos de Blast. Posteriormente se compararon las secuencias impresas en el Pinarray1 con esta base de datos mediante BlastN con un valor de $E = 10^{-3}$, tomando el número de acceso del TC con mayor similitud. Todas estas anotaciones se añadieron al fichero GAL, obteniéndose un nuevo fichero GAL anotado que contenía nuevas columnas para las anotaciones. Así pues, las columnas de este fichero GAL para Pinarray1 contiene las columnas Block, Row, Column, Name, ID, placa384, Clone, EMBL, Longitud, Blast2GO, GoTerms, ECs, ECNames, KeggMaps, y PGI. A continuación se explica el contenido de este nuevo fichero GAL, según sus tres categorías de datos (coordenadas, nombres y anotaciones):

- **Coordenadas:** `Block`, `Row` y `Column` indican las coordenadas de las sondas de ADNc en la micromatriz. `Placa384` indica las coordenadas del clon del que se obtuvo la sonda en la placa de 384 pocillos utilizada para crear la micromatriz. `Clone` indica las coordenadas de estos clones en las placas de 96 pocillos en las que se almacenaron originalmente.

- **Nombres:** En el fichero GAL anotado hay diferentes nombres e identificadores para reconocer las secuencias de ADNc en diferentes situaciones. `Name` indica el nombre que el programa informático del robot de impresión asigna a la secuencia de ese punto, `ID` indica el número de acceso en las bases de datos internacionales GenBank, EMBL y DDBJ. `Clone` muestra el nombre asignado al clon en el laboratorio. `PGI`, número de acceso del TC más parecido en el Pine Gene Index.

- **Anotaciones:** agrupa diversa información para identificar el producto del gen; `EMBL`, contiene la descripción asignada a la secuencia en

la base de datos EMBL. `Blast2GO`, la descripción obtenida con este programa. `GoTerms`, los términos de la *Gene Ontology*, `ECs` el código de las enzimas según la *Enzimme Commission*, `ECNames` el nombre de las enzimas, y `KeggMaps` las rutas metabólicas en las que participa el gen.

El nuevo preprocesamiento con SeqTrim, puso de manifiesto que algunos de los clones impresos en el Pinarray1 se encontraban repetidos y que otros no contenían información útil. Los nombres de los clones sin información se almacenaron en el fichero `BadSpots.txt` que se utilizará en MADE4-2C para descartar la utilización de estos puntos en los análisis de micromatrices realizados con el Pinarray1.

Al ver que los resultados obtenidos en el análisis anterior con los clones de Gemini (genoteca que más aporta al Pinarray1), utilizando StackPACK [83] en el INRA de Burdeos (J.M. Frigerio, comunicación personal), eran mejorables, se decidió realizar de nuevo el mismo análisis aplicado para anotar el fichero GAL del Pinarray1 sobre los clones de la genoteca Gemini. En este análisis se descubrieron 559 secuencias descartadas en el análisis anterior que resultaron ser válidas. Todas estas secuencias recuperadas se incluyeron en las bases de datos internacionales con los números de acceso FM945441 a FM945999.

Los conocimientos obtenidos durante la construcción del fichero GAL para el Pinarray1, fueron útiles para la anotación y el análisis de un chip de Affymetrix, con secuencias de cerdo (*Sus scrofa*), realizado en colaboración con el grupo de investigación del profesor Juan José Garrido (Genómica y Mejora Animal, del Departamento de Genética de la Facultad de Veterinaria de la Universidad de Córdoba). Este trabajo queda reflejado en el artículo incluido en el apéndice C, pág. 245.

9.2. MADE4-2C: automatización del análisis de micromatrices de dos colores

MADE4-2C (*Microarray Analysis of Differential Expression for two color hybridisations*) es un flujo de trabajo para el análisis de micromatrices de dos colores. Se escribió utilizando R y las librerías de *Bioconductor* disponibles para el de análisis de micromatrices, principalmente `limma`, `marray` y `rankprod` (apartado 7.2, pág. 49). Esta herramienta genera un informe en formato PDF (apéndice B, pág. 199) con toda la información necesaria para que el investigador recuerde cómo analizó el experimento, evalúe la calidad de los datos y obtenga el resultado de las hibridaciones; pero lo más importante es que con ese informe se pretende que el investigador entienda paso a paso cómo se han preprocesado, normalizado y analizado los datos, y cómo se ha obtenido la lista de los GED.

9.2.1. Definición del experimento

Para la ejecución de MADE4-2C es necesario rellenar un fichero de configuración con los parámetros del programa y los valores que se quieren aplicar al análisis (véase el apéndice D). Además de ser imprescindible también, la definición de un fichero `targets` con el diseño del experimento, como el que se muestra en la tabla 1.1 del apéndice B. En este fichero se indica la ruta relativa de la que tomar los datos originales de expresión que se van a analizar (en *FileName*), los nombres de las micromatrices que se utilizarán en las imágenes que genera el programa (en *Label*), y el diseño del experimento, indicando que condición experimental se marcó con cada fluoróforo en cada micromatriz (en *Cy3* y *Cy5*), y que micromatrices son réplicas técnicas o biológicas (en *repBiol*). En el ejemplo del apéndice hay 4 réplicas biológicas con 2 réplicas técnicas cada una. Los datos del diseño experimental son imprescindible para el análisis, ya que son requeridos por algunas funciones de `limma`.

Antes de pasar a la siguiente etapa, MADE4-2C verifica que los ficheros necesarios se han cargado y que los valores que hay que añadir se encuentran en los márgenes aceptables. En caso de encontrar algún error, avisará al usuario y detendrá la ejecución.

9.2.2. Evaluación de la calidad

Lo primero que hace MADE4-2C es generar una serie de imágenes que aportan información de la homogeneidad de la señal y del ruido de fondo. La primera es una gráfica donde se observa la variabilidad de la intensidad del ruido de fondo en cada canal de cada micromatriz (figura 2.1, apéndice B). En esta figura, los valores de intensidad del fondo se acotan entre 5,5 y 10, valores arbitrarios asignados basándose en nuestra experiencia con distintos conjuntos de datos; para que sirvan de referencia aparecen como líneas de color turquesa. Si el ruido de fondo es superior a 10, un valor demasiado alto para el ruido de fondo incluso si la micromatriz se escaneó con un valor de fotomultiplicador alto, MADE4-2C incluirá en el informe un mensaje en el que advertirá de este problema.

También se proporcionan imágenes artificiales de intensidad por rangos para cada micromatriz hibridada, tanto de la señal de fondo (figuras 2.2 y 2.3,

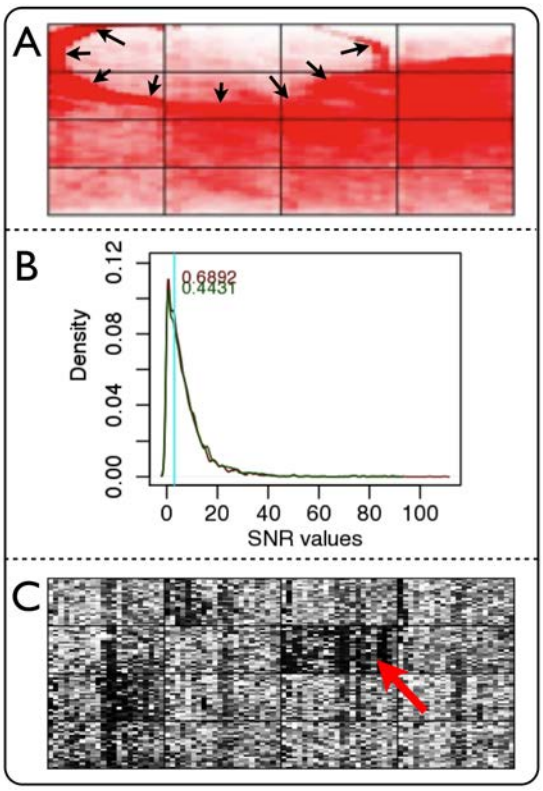


Figura 9.1: *Ejemplo de micromatrices que ilustran problemas con la señal y con el fondo. (A) Micromatriz con patrón circular en la señal del ruido de fondo, señalado con flechas negras. (B) Intensidad global de las sondas de la micromatriz con un valor inferior al triple del ruido de fondo. (C) Micromatriz con un bloque cuyos valores de expresión son menores que en el resto, señalado con flecha roja.*

apéndice B), como de la señal propiamente dicha (figura 2.4, apéndice B). La agrupación de los valores por rangos sirve para que los valores continuos pasen a ser discretos y se acentúan las diferencias. En el informe se explica que estas imágenes deben ser homogéneas y no presentar sesgos ni irregularidades. Este no es el caso de la figura 9.1-A, donde se puede ver un patrón de ruido de fondo claramente artefactual, en el que se ve con claridad una irregularidad circular en la señal del ruido de fondo, posiblemente debido a la formación de una burbuja en algún momento de la hibridación o los lavados. La aparición de este artefacto indica que conviene repetir la hibridación para evitar que los resultados se vean afectados por estos artefactos técnicos.

A continuación se evalúa la relación entre la señal y el ruido de fondo de los dos canales de la micromatriz, lo que puede ser un buen indicador de problemas en el marcaje: cuando ambos valores están muy próximos, las señales no son fiables. Se consideran aceptables las micromatrices cuya intensidad de la señal es al menos el triple de la intensidad del fondo, como se muestra en la figura 2.5 del apéndice B. En ella, la línea celeste divide la gráfica entre valores cuya señal es inferior a 3 (izquierda) y superior a 3 (derecha). Lo ideal es que la mayoría del área de la curva esté del lado derecho, lo que se puede evaluar de manera sencilla comprobando que el pico está a la derecha de la recta, como es el caso todas las hibridaciones de la figura comentada, pero no de la figura 9.1-B. Tras esta figura, MADE4-2C muestra en el informe los valores de la relación observada entre la intensidad y el fondo, señalando los que presenten una relación entre el fondo y la señal inferior a 3, en esos casos convendría repetir las hibridaciones de esas micromatrices. En el ejemplo del apéndice B, todas las micromatrices mostraron valores fiables.

Otra valoración de la calidad consiste en comprobar que los valores de M y de A son homogéneos después de corregir el ruido de fondo, puesto que la distribución de los genes impresos en la micromatriz es al azar. Por eso en la figura 2.6 del apéndice B se comprueba la homogeneidad del marcaje con los dos fluoróforos, mostrándose en la figura puntos verdes o rojos según si la expresión es mayor en un canal o en otro. En la figura 2.7 del apéndice B lo que se pone a prueba es si la distribución de la intensidad de señal es homogénea, mostrándose los puntos vacíos (sin sonda) en negro y los puntos con mayor intensidad en blanco. Los puntos con intensidades intermedias se representan según su intensidad con colores de la escala de grises. En caso de que la distribución no sea homogénea en cualquiera de estas figuras, habrá que esperar a ver si los datos normalizados corrigen las imperfecciones mostradas. De no corregirse, se deberían repetir las hibridaciones. En cada bloque de la micromatriz las sondas se imprimen con la misma aguja, por lo que un bloque de color negro puede estar indicando el fallo de una de las agujas. Este sería el caso del resultado mostrado en la figura 9.1-C, en el que se observa que una de las agujas del robot de impresión introduce diferencias en la expresión antes de la normalización. Es posible que el bloque señalado contenga menos sondas porque la aguja esté rota, mal calibrada o sea defectuosa. La diferencia de intensidad de señal mostrada en la figura 9.1-C se debe a problemas técnicos y no a la expresión diferencial de los genes, por lo que se debe proceder a cambiar la aguja de impresión o a recalibrar el robot.

Es importante tener en cuenta que cuanto menos haya que modificar los datos de una micromatriz, más significado biológico se conservará. Por eso conviene evaluar el aspecto de los datos sin modificar. Para ello, MADE4-2C construye gráficas MA (figuras 2.8 y 2.9, apéndice B) donde observar si hay saturación en los datos por alta intensidad (figura

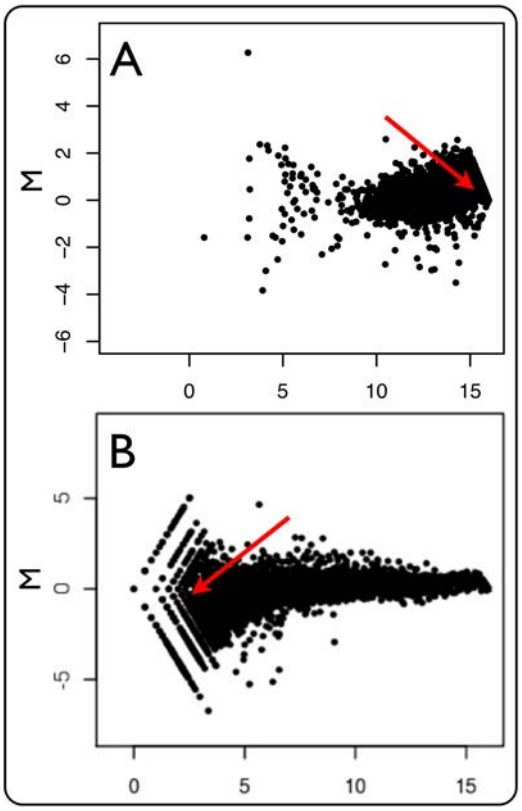


Figura 9.2: *Ejemplos de gráficas MA que muestran saturación por alta intensidad de la señal (A) y cuantización por baja intensidad (B), señaladas con flechas rojas. Ambos efectos provocan lo que se suele denominar efecto de «punta de flecha».*

9.2-A) o cuantización por baja intensidad de la señal (figura 9.2-B). La presencia del efecto en forma de flecha patente en los gráficas MA con saturación o cuantización (figura 9.2) indica que los valores de expresión de esas sondas son demasiado altos o bajos, respectivamente. El problema de estas sondas es que han sobrepasado los límites de medición superior o inferior del escáner, por lo que muchas sondas reciben un mismo valor, a pesar de que este debería de ser distinto para cada una de ellas. Además, la presencia de saturación implica que se escaneó la micromatriz con un valor de fotomultiplicador demasiado alto, y la cuantización indica que tenía que haberse utilizado un fotomultiplicador más alto.

Los datos en bruto deben presentar una distribución equilibrada en torno al valor $M = 0$, como los de la figura 2.8 de apéndice B. Si se observara que el centro de la nube de puntos está por encima o por debajo del valor $M = 0$ es porque hay un desequilibrio entre los canales R y G (que puede deberse a diferente marcaje experimental, o diferente sensibilidad del escáner). Este efecto se comprueba mejor en la figura 2.9 del apéndice B, en la que se representan las curvas de ajuste loess de los datos originales. Cuanto más solapantes sean las curvas de ajuste con la recta de $M = 0$, menos sesgo tendrán los datos. El sesgo mencionado se observa en la gráfica MA de la figura 9.3 (son los mismos datos que se muestran en la figura 9.1-C, del *Swirl zebrafish microarray experiment*). Se ve claramente que los ajustes loess de los datos brutos se separan claramente de la línea $M = 0$. En los casos de este tipo, aunque el sesgo luego desaparezca tras la normalización, debería considerarse repetir la hibridación porque la transformación matemática de los datos seguramente habrá hecho perder mucha información biológica.

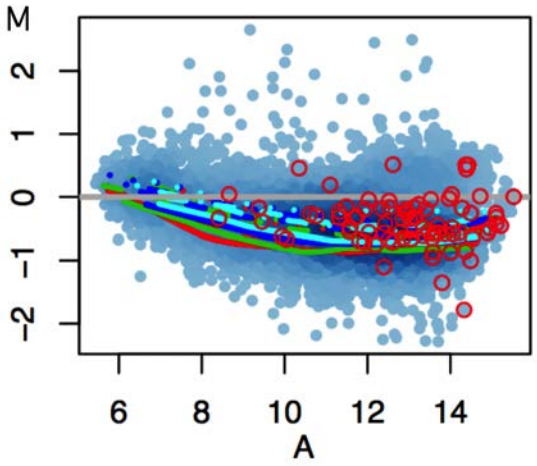


Figura 9.3: *Ejemplo de micromatriz con una distribución de datos desigual y con curvas loess muy heterogéneas.*

Conviene también señalar que, en la figura 9.3 de este apartado, y en las figuras 2.9 del apéndice B se indican en rojo aquellas sondas que por baja calidad no se considerarán en los análisis posteriores en ninguna de las micromatrices. Los valores que indican si un punto es de baja calidad vienen asignados por los programas de análisis de imágenes. Por ejemplo, GenePix, que es la plataforma utilizada en el Pinarray1 (apartado 8.1) produce dos índices para hacer referencia a la calidad de cada punto, por un lado utiliza las *flags* para indicar con valores negativos aquellos puntos con diferentes tipos de problemas. Por otro lado, en el índice *area* indica la calidad de la forma y el área que ocupa cada punto, considerando de mala calidad tanto a valores demasiado altos como demasiado bajos en referencia a 100 como valor ideal. Así pues, MADE4-2C descartará del análisis las sondas con un peso específico 0, que se asignará a cada sonda de cada micromatriz cuando el valor de *flags* es menor de -50 [206] y a puntos cuya área no esté entre 170 y 20 píxeles. La mayoría de las sondas descartadas suelen mostrar valores bajos de intensidad, que el escáner no ha

sido capaz de medir con fiabilidad. Suelen corresponder a puntos vacíos, puntos donde se despegó sonda después de imprimirla, o donde la sonda no ha mostrado una hibridación suficientemente buena con las dianas del experimento.

En conclusión MADE4-2C es capaz de detectar errores en la intensidad de la señal, en el lavado, la hibridación, el marcaje con el fluoróforo, las agujas de impresión y la calidad de las sondas impresas. Esto ayuda a evitar que los resultados se basen en las variaciones técnicas en lugar de en las variaciones biológicas. Además, ofrece toda la información en un informe denso pero comprensible para el investigador, lo que permite una buena evaluación del experimento sin tener unos conocimientos avanzados sobre micromatrices.

9.2.3. Descarte de sondas fallidas

Una vez que se proporciona información al usuario sobre la calidad de los datos originales que quiere analizar, MADE4-2C procede a la corrección del ruido de fondo utilizando `normexp` ([184]) y genera las gráficas MA que muestran cómo quedan los datos tras corregir el fondo (figuras 2.10 y 2.11, apéndice B).

A continuación se muestran las sondas que se utilizarán en el experimento y las que se descartarán. Una sonda se descartará siempre cuando su punto está vacío según la información del fichero GAL, o cuando la sonda contiene una secuencia artefactual o mal caracterizada (información que se incorporó desde el fichero `BadSpots.txt`). Existen dos motivos de rechazo que solo afectan a algunas sondas en una micromatriz, pero no tiene por qué afectar a las demás réplicas:

- El punto correspondiente a la sonda no se imprimió o es de baja calidad, lo que viene indicado por su peso específico a partir de los campos *flags* y *area*.
- La corrección del ruido de fondo con `normexp` ha marcado la sonda como descartable.

La tolerancia a estos fallos es controlable mediante un parámetro del fichero de configuración (véase el apéndice D) que indica el número de réplicas fallidas permitidas para cada sonda en el experimento que se analiza. Lo recomendable es que se retire la sonda en todas las micromatrices en cuanto falle una de las réplicas por cualquiera de los motivos anteriores, aunque teóricamente el análisis se puede realizar con tal que una sonda tenga dos o más réplicas valores de intensidad válidos. En el caso de los experimentos analizados sobre la expresión génica de pino se descartaron las sondas en cuanto fallaban en una réplica técnica o biológica.

Con toda esta información, MADE4-2C genera una figura en la que se marcan con un recuadro negro los puntos que serán descartados para el análisis (figura 2.12, apéndice B). Es de esperar que este filtro no retire más del 15 % de las sondas [184] como se muestra en la figura 2.12 del apéndice B. En cambio, es recomendable repetir el experimento si se acaban descartando más del 15 % de las sondas, como se muestra en la figura 9.4.

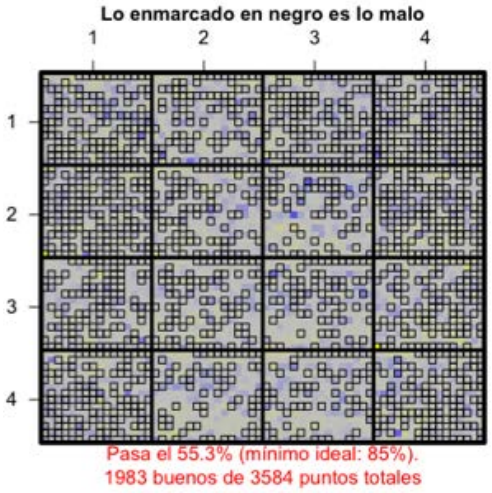


Figura 9.4: *Ejemplo de figura generada por* MADE4-*2C para indicar que se han descartado demasiadas sondas impresas para el análisis posterior.*

9.2.4. Normalización

La normalización de los datos tiene en cuenta las réplicas técnicas para confirmar que los valores de expresión no introducen más variabilidad de la que había antes de la normalización, y que ninguno de los marcajes con fluoróforos añade ningún tipo de sesgo a los datos. Aunque son muchos los métodos de normalización que se han propuesto, todavía no hay un consenso claro de que un método sea el mejor frente a las diferentes condiciones experimentales posibles [45], y puesto que el método de normalización utilizado es uno de los factores que más afectará posteriormente a la detección de GED [187, 98, 45], y es posible obtener mejores resultados combinando dos de ellos [187], MADE4-2C lleva a cabo la normalización de modo independiente con varios métodos: `Print-tip loess` [207], `Print-tip loess + scale`, `Print-tip loess + quantile` [28], con la función `normalizeBetweenArrays` de `limma`, y por último, `VSN` [62] y `VSN + Print-tip loess` [45].

En el análisis con MADE4-2C, los datos normalizados obtenidos se guardan en ficheros de texto tabulado y en formato marrayNorm para poder utilizarlos en otros programas o volverlos a analizar sin tener que empezar desde el principio.

Evaluación de la coherencia de las réplicas técnicas: Se supone que el objetivo de realizar las réplicas técnicas es comprobar que se comportan igual para confirmar que todo hay ido bien. La comparación se realiza calculando dos distancias diferentes (euclídeas y de tipo Manhattan) y dos correlaciones diferentes (de Pearson y de Spearman), para verificar si los datos de las réplicas son equivalentes o presentan algún tipo de sesgo. En un caso ideal se esperaría que las réplicas técnicas se agrupen juntas y a un nivel superior aparezcan las réplicas biológicas. En las figuras 2.15 a 2.21 del apéndice B puede observarse un ejemplo de los árboles de distancias y correlación que se generan para cada uno de los métodos de normalización, donde las réplicas técnicas A/Z siempre son las más cercanas. En cambio, en la figura 9.5 se muestra un ejemplo en el que los datos de la hibridación no son tan fiables, ya que la réplica técnica (102a) se parece más a las réplicas de otro individuo (101), que a la otra réplica técnica de su mismo individuo (102z). El usuario también puede descartar un método de normalización si ve que en él se producen comportamientos anómalos similares al descrito en la figura 9.5, mientras que en los demás no se observan.

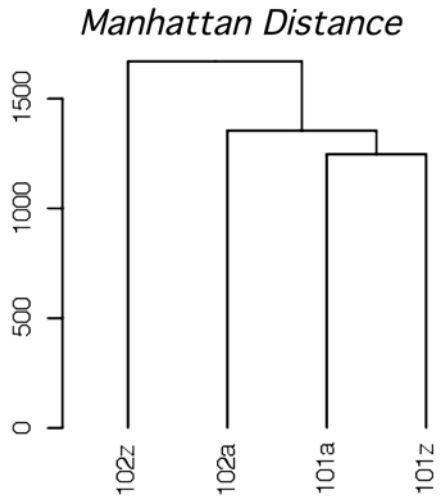


Figura 9.5: *Dendrograma basado en las distancias de Manhattan de dos individuos (101 y 102) de los que se han hecho dos réplicas técnicas (a y z). Se observa un comportamiento anómalo de la réplica técnica 102a, puesto que se parece más a las réplicas del individuo 101 que a la otra réplica técnica del individuo 102 (la 102z).*

A partir de este momento, MADE4-2C proporciona una serie de cálculos destinados a elegir las mejores normalizaciones basándose en los resultados de cuatro pruebas: los pesos específicos entre micromatrices, la dispersión de los datos, la dependencia de la variabilidad experimental frente a la intensidad y la correlación de los datos. Los datos de todos los métodos de normalización que superen estas pruebas se analizarán de forma independiente para detectar los genes que se expresan diferencialmente.

Evaluación de los pesos específicos de las micromatrices: Este análisis se ilustra en las figuras 2.13 y 2.14 del apéndice B, donde se comparan los pesos específicos de cada micromatriz y de cada bloque de la micromatriz, respectivamente, para cada método de normalización. Lo ideal es que ni una micromatriz valga más que otra, ni un bloque sea más azul o blanco que los otros.

En la figura 9.6 se muestra un ejemplo de un caso ideal y un caso de mala calidad que pone de manifiesto que alguna de las hibridaciones no ha salido bien. En el caso de la figura 9.6-A, todas las micromatrices tendrían el mismo peso específico en el análisis, y en el caso de la figura 9.6-B, las micromatrices 5, 6, 7 y 8 tendrían un peso mucho menor en relación a las micromatrices 1, 2, 3 y 4. MADE4-2C evalúa los resultados obtenidos en todos los métodos de normalización, y en los casos en los que la micromatriz de mayor peso específico tenga un valor 4 veces mayor que la micromatriz de menor peso (valor empírico obtenido de los datos que se han manejado), se descarta el método de normalización del que se obtuvo ese resultado. Independientemente del resultado obtenido, el programa continúa valorando por igual todas las micromatrices y es el usuario quien debe decidir si repetir las hibridaciones de las micromatrices con menor peso específico, y por tanto de menor calidad.

Evaluación de la dispersión de los datos: A continuación, MADE4-2C genera una serie de gráficas de cajas (figura 2.22 del apéndice B), en las que se evalúa la dispersión de los puntos en cada método de normalización. En esta prueba se compara la distribución de las sondas entre cada micromatriz del experimento, teniendo en cuenta la simetría de la distribución, los valores máximo y mínimo, y los cuartiles indicados en la figura 9.7. Es de esperar que tras la normalización los datos muestren una menor dispersión que en los datos originales. Por tanto, se espera que en la figura 2.22 del apéndice B los datos normalizados queden alineados por su mediana en el valor de $M = 0$ (línea turquesa) y que los tamaños de las cajas y los valores de

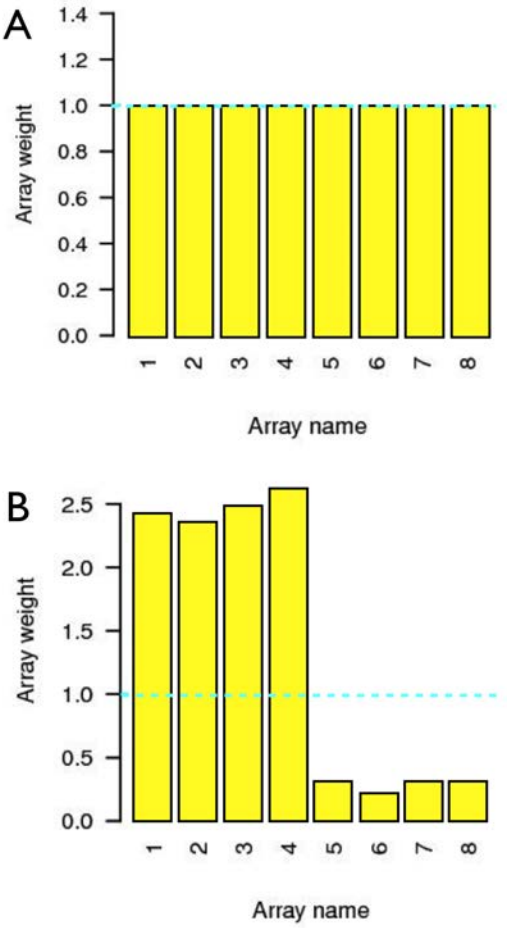


Figura 9.6: *Histograma de los pesos específicos relativos de las micromatrices en el experimento, en un caso ideal (A) en el que todas las micromatrices tienen el mismo peso específico, y un caso (B) en el que cada micromatriz muestra una calidad diferente.*

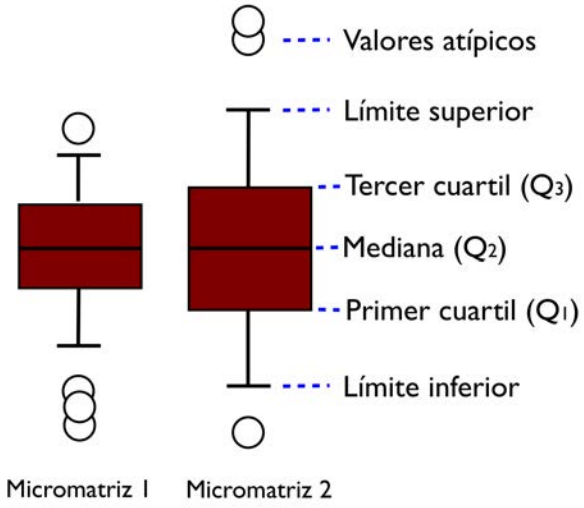


Figura 9.7: *Representación de los datos en gráficas de cajas divididas en cuartiles. Bajo el cuartil Q1 están representados el 25 % de los datos, el Q2 es la mediana, que divide el 50 % de los datos arriba y abajo, y bajo el Q3 están el 75 % de los datos. El límite inferior corresponde a Q1 − 1,5RIC, y el límite superior a Q3 + 1,5RIC, donde RIC (recorrido intercuartílico [IQR en inglés]) es Q3 − Q1. Por encima y debajo de los límites superior e inferior se encuentran los valores atípicos, donde aparecerán los GED.*

los límites inferiores y superiores sean equivalentes entre micromatrices. MADE4-2C compara los valores máximo y mínimo de cada micromatriz para los límites y los cuartiles permitiendo una diferencia máxima del 15 % para el valor de M en los límites, del 10 % para los cuartiles Q1 y Q3 y del 5 % para la mediana (Q2). Los métodos de normalización que muestran una dispersión mayor de la permitida son descartados y no se aplicarán en la detección de genes expresados diferencialmente. Por eso en la figura 2.22 del apéndice B, donde las cajas y límites parecen iguales, solamente la combinación de los métodos Loess y Scale permite obtener datos homogéneos y comparables.

Otro modo de evaluar la dispersión de los datos es a través de la visualización de curvas de densidad y gráficas QQ. En las figuras 2.23 y 2.24 del apéndice B respectivamente se muestran estas representaciones gráficas, aunque simplemente se muestran para ilustración por ser comunes en experimentos de micromatrices, pero no suponen la toma de ninguna decisión por parte del algoritmo de MADE4-2C. Se espera que las curvas de densidad sean similares en ambos canales R y G, y los máximos de densidad de cada micromatriz se encuentren en los mismos valores de intensidad. Es común encontrar picos pequeños antes o después de la curva principal de densidades, que corresponden a fenómenos de cuantización y saturación respectivamente. El método de normalización `quantile` debe mostrar curvas de densidades idénticas para todas las micromatrices en ambos canales, ya que su normalización se basa precisamente en equiparar las intensidades de las sondas a largo de cada micromatriz y cada canal [28]. Por otro lado, las gráficas QQ son de utilidad para ver si los datos se ajustan a una distribución normal. Los datos normalizados deben ajustarse en su mayoría a la línea de cuartiles, mientras que los puntos que se separan de ésta en los extremos corresponden a los valores atípicos, que posiblemente contengan a los genes expresados diferencialmente. Si la zona central de los datos no se ajusta a la recta se está teniendo un indicio de que los datos, a pesar de estar normalizados, no tienen suficiente calidad para asegurar que los GED obtenidos sean fiables.

Evaluación de la dependencia de la intensidad: En un experimento bien realizado, los valores de M no deben depender de los valores de intensidad A. Para evaluarlo, MADE4-2C genera las figuras 2.25 y 2.26 del apéndice B para cada método de normalización. En la figura 2.25 se muestra una gráfica MA con los valores de intensidad de A agrupados por rangos. Una línea azul da la referencia del valor de $M = 0$ y una línea roja muestra la dependencia de los valores de M con respecto a A. Es de esperar que, tras la normalización, la línea roja se ajuste los más posible a la línea azul con una pendiente cercana a cero. Los métodos que introduzcan más dependencia en los datos, es decir, los que muestren datos que se ajusten a una recta con mayor pendiente, que la que había en los datos originales o en los datos con el fondo corregido, son descartados.

En la figura 2.26 del apéndice B, lo que se compara es si la variabilidad experimental depende de la intensidad media de la expresión. En esta prueba se espera que los datos normalizados muestren una línea con una pendiente cercana a cero al menos el la zona central (las desviaciones a valores bajos de A son una prueba de que la normalización no corrigió la cuantización). No se utiliza para descartar métodos de normalización.

Evaluación de las normalizaciones que menos distorsionan los datos: Se consideran buenas normalizaciones aquellas que no introducen más dispersión en los datos que la que traen originalmente [233], y es de esperar que los niveles de expresión de un mismo gen sean los mismos entre las réplicas. Por tanto, la variabilidad de los valores de M para cada gen puede utilizarse para comparar los métodos de normalización entre sí [233]. Una manera de valorar la distorsión consiste en comparar la variabilidad (desviación estándar) de los valores de M para el conjunto de sondas de cada micromatriz y compararla con la variabilidad de los datos originales. Las desviaciones estándar de menor valor son indicativas de un proceso de normalización más eficaz [233]. Esta evaluación de la variabilidad de las réplicas técnicas para cada método de normalización se muestra en la figura 2.27 (apéndice B), y la variabilidad media entre todas las micromatrices, en la figura 2.28 y en la tabla 2.1 del apéndice B. Los métodos de normalización con las cajas más compactas muestran menos diferencias entre sus réplicas y son considerados como buenos. Los métodos de normalización en los que haya mayor variabilidad entre sus réplicas que en las réplicas de los datos originales, son descartados.

Basándose en otros estudios [233, 96] en los que se compara la variabilidad y la correlación de las réplicas para comparar que método de normalización es mejor, MADE4-2C pone a prueba los métodos de normalización mediante correlaciones de Pearson y de Spearman, y se calculan los coeficientes estadísticos Kolmogorov-Smirnov (KS), Pearson, y Spearman (figuras de 2.29 a 2.33 del apéndice B) para evaluar la correlación entre las réplicas técnicas y biológicas. Los métodos de normalización cuyas correlaciones entre réplicas sean peores que las de los datos originales, son descartados.

En la figura 2.29 del apéndice B se comparan los valores medios de KS de todas las micromatrices para los diferentes métodos de normalización (parte superior de la figura) y los KS de las 4 réplicas biológicas obtenidos a partir de las comparación de sus 2 réplicas técnicas (parte inferior de la figura). Un proceso de normalización eficaz deberá mostrar distribuciones muy parecidas, con un KS lo más cercano a cero posible [233]. Por tanto, en esta figura, los mejores métodos de normalización serán los que muestren barras de menor tamaño, y las réplicas biológicas con menores valores (de las 4 barras de un mismo color), serán aquellas cuyas réplicas técnicas son más parecidas.

En la figura 2.30 del apéndice B se realiza una prueba equivalente a la de la parte superior de la figura anterior, pero utilizando el coeficiente de correlación y el estadístico de Pearson para comparar la correlación que hay entre todas las réplicas del experimento (biológicas y técnicas), para cada método de normalización. En este caso, los mejores métodos de normalización mostrarán mayores valores de correlación y del estadístico (mejor cuanto más cercano a 1). La figura 2.31 es equivalente a la anterior, pero en ella se comparan las correlaciones de las réplicas técnicas de las 4 réplicas biológicas.

Las figuras 2.33 y 2.34 son equivalentes a las dos figuras anteriores, pero en este caso se utiliza el coeficiente de correlación de Spearman y su estadístico. La única diferencia a tener en cuenta, es que el estadístico de Spearman, a diferencia del de Pearson, indica mejor correlación cuando su valor es menor.

En el análisis de correlación con los coeficientes y estadísticos de KS, Spearman y Pearson, los métodos de normalización que muestren menor correlación entre sus réplicas, que los datos sin normalizar (con y sin corregir el fondo), son descartados, ya que están introduciendo más variaciones en los datos que las que había originalmente.

Selección de los mejores datos normalizados: Para la búsqueda de genes expresados diferencialmente se emplearán nada más que los métodos de normalización que hayan superado la evaluación realizada en las cuatro pruebas anteriores (pesos es-

pecíficos entre micromatrices, dispersión de los datos, dependencia de la intensidad y correlación de los datos). En caso de que ningún método de normalización supere esta evaluación se debe pensar que los datos originales no eran muy buenos y que la normalización modificó demasiado unos datos que ya debían presentar una dispersión anómala. Para que el usuario pueda de todas formas aprovechar estos datos se realizará un análisis con el método `loess` que, basándose en experiencias anteriores, suele ser el que menos modifica los datos, además de ser el método más extendido para este tipo de experimentos [207]. En el informe de MADE4-2C se indicará esta eventualidad, advirtiendo que sería aconsejable repetir la hibridación para tener unos datos más fiables.

Para cada método de normalización válido, MADE4-2C incluirá en el informe una serie de imágenes artificiales de los valores de A y M por rangos (figura 2.34 del apéndice B) para que el usuario compruebe visualmente que los datos normalizados han hecho desaparecer cualquier sesgo espacial anterior. También proporciona gráficas MA (figuras 2.35 y 2.36 del apéndice B) para comprobar que la nube de sondas está en torno a $M = 0$, que los ajustes loess coinciden con $M = 0$, y la posición de las sondas de control (si se pusieron) y las sondas que se han descartado para el análisis (en el ejemplo se observa que todas corresponden a sondas con un valor de A muy bajo).

9.2.5. Resolución de las réplicas

Hasta ahora se tiene una colección de micromatrices, réplicas unas de otras, que conviene dejar en un valor único para cada sonda, para que con él se puedan detectar los genes expresados diferencialmente. Para ello, se resuelven a la vez todas las réplicas técnicas y biológicas que MADE4-2C ha calificado como útiles. Además, con la resolución de las réplicas se consigue que se equilibren los casos en los que haya más réplicas técnicas para una condición que para la otra. Hay que tener en cuenta que es en este momento (no así en los datos originales) cuando se pueden comparar los valores entre micromatrices y se pueden promediar sin que este cálculo matemático introduzca una dispersión significativa. Si en el fichero de configuración (véase el apéndice D) se especificó un valor para el parámetro `uMinFC`, después de resolver las réplicas se descartan del análisis los puntos cuyas veces de cambio están por debajo de la mínima indicada por el usuario, lo que sirve para eliminar del análisis las sondas con valores de M más cercanos a cero. Esta eliminación se basa en la demostración de que hay una gran varianza en los valores de expresión cercanos a cero [62], y que este efecto penaliza la expresión diferencial [45]. Al final de este capítulo, en el apartado 9.3, se ilustra con un ejemplo cómo, al aplicar esta opción, aumenta el número de genes expresados diferencialmente.

Como se mencionó anteriormente (véase el apartado 9.2.4), no hay un método de normalización ideal para todos los experimentos, y la decisión de qué método elegir afectará profundamente a la posterior detección de genes expresados diferencialmente. Por ello, la estrategia de MADE4-2C de normalizar los datos con diferentes métodos y luego elegir mediante pruebas objetivas qué método o métodos son los que mejor normalizan los datos, suele devolver resultados más fiables. Además de que, como se verá más adelante, es posible obtener genes expresados diferencialmente más fiables si se han detectado independientemente utilizando diferentes métodos de normalización [187].

9.2.6. Detección de GED

Para la detección de los GED (Genes Expresados Diferencialmente), y dado que no existe acuerdo sobre si las micromatrices han de tratarse por métodos paramétricos o no paramétricos ([30, 205]), MADE4-2C utiliza dos aproximaciones. Por un lado utiliza las funciones de la librería `limma` de R/-Bioconductor como método paramétrico, en el que la detección de GED se realiza mediante ajustes lineales y bayesianos, y pruebas estadísticas de t moderadas (véase el apartado 2.5.5 de la introducción, pág. 14), y por otro, las funciones de la librería de R `rankprod` como método no paramétrico (descrito en el apartado 2.5.5 de la introducción de este manuscrito). A continuación, MADE4-2C ofrece hasta 12 listas de GED (véase el apartado 5.1 del apéndice B) para combinar los resultados obtenidos en cada condición experimental, como por ejemplo control y tratamiento, según el método de normalización y el método de detección de GED:

Hay 4 listas para los GED detectados con el método paramétrico, otras 4 con el método no paramétrico y 4 más para los GED detectados en ambos métodos simultáneamente:

1. GED en el control, detectados al menos con un método de normalización.

2. GED en el control, detectados en todos los métodos de normalización.

3. GED en el tratamiento, detectados al menos con un método de normalización.

4. GED en el tratamiento, detectados en todos los métodos de normalización.

Las listas con los GED detectados en al menos uno de los métodos de normalización contendrá un mayor número de GED (listas 1 y 3 mencionadas arriba), proporcionando a los investigadores una mayor cantidad de genes que se expresan diferencialmente, aunque pueden ser de menor fiabilidad si aparecen con unos métodos de normalización y otros no, por lo que en ese caso, deberían ser verificados con otras técnicas como RT-PCR. Las listas que contienen los GED que aparecen en todos los métodos de normalización (listas 2 y 4 mencionadas arriba) contendrán GED muy fiables, puesto que se ha descrito que la detección de GED utilizando diferentes métodos de normalización de modo independiente aumenta la fiabilidad de estos [187]. Por último, las listas 2 y 4 obtenidas con ambos métodos de detección de GED, aunque son más restrictivas, contienen los GED de máxima fiabilidad, puesto que han superado varias pruebas de normalización y métodos de detección de GED, uno paramétrico y otro no paramétrico.

Si el análisis de expresión diferencial ha funcionado correctamente se espera que aparezca un incremento claro en los valores de P cercanos a cero, como el observado en la figura 3.1 (apéndice B). En el caso del método no paramétrico, al seguir otros criterios, también es normal observar una acumulación de genes el extremo opuesto con valores de P cercanos a uno (figura 4.2 del apéndice B). En cambio, en la figura 9.8, se muestra un claro ejemplo de una mala distribución de los valores de P, ya que el valor acumulado en $P = 0$ es el menor, y el mayor es el de $P = 1$, señal de que o bien el experimento se hizo mal, o de que no hay genes expresados diferencialmente.

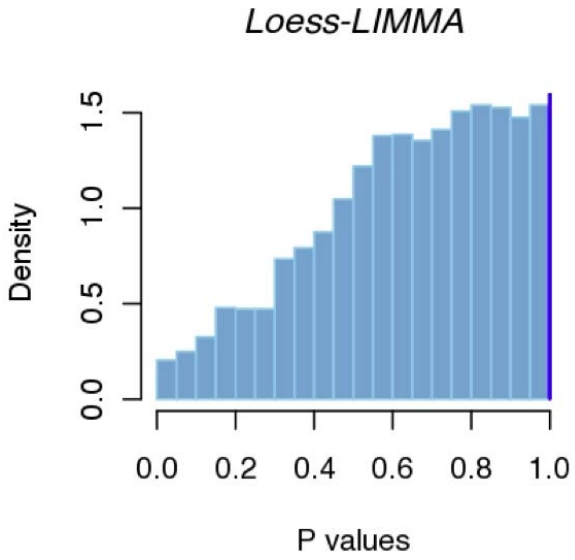


Figura 9.8: *Distribución incorrecta de los valores de P.*

Los GED se pueden inspeccionar mediante gráficas en volcán, gráficas MA y mapas térmicos (figuras 3.3, 4.3 y 5.2 del apéndice B). Si se han especificado las sondas de control en el fichero `ControlSpots.txt`, éstas aparecerán en dichas gráficas. En las figuras mencionadas del apéndice B aparecen marcadas unas sondas que se sabía que se comportaban como GED. Al aparecer todas ellas en las zonas de expresión diferencial se puede concluir que el experimento está bien realizado y los resultados serán útiles y coherentes.

En la figura 4.1 del apéndice B se muestra la representación gráfica que se obtiene al detectar los GED con el método no paramétrico, `Rank Products`, donde se pueden encontrar para cada condición por separado, el porcentaje de positivos falsos estimados (PFP) en el eje y frente a los genes identificados en el eje x. En los valores más bajos de positivos falsos estimados se encuentran los GED marcados en rojo. Como estas figuras no son muy intuitivas para interpretar los resultados, MADE4-2C une los datos obtenidos de ambas condiciones experimentales para generar un gráfico en volcán (figura 4.3 del apéndice B), que es una de las representaciones que se usan con más frecuencia para mostrar los GED. La salida de los datos del método no paramétrico devuelve dos listas de valores de P para cada gen, una para cada condición experimental. Así pues, los genes con el valor de $-log_{10}P$ más alto en una de las listas (arriba en rojo en la gráfica en volcán), son los que muestran los valores más bajos en la condición opuesta (abajo en azul en la gráfica en volcán), y viceversa. Por eso estas gráficas en volcán presentan dos colas en cada condición para los valores de P más altos y más bajos.

9.2.7. Ventajas de MADE4-2C frente a otras herramientas bioinformáticas

MADE4-2C es un flujo de trabajo que incorpora numerosas funciones de Bioconductor [82] para el análisis automático de micromatrices de dos colores. Uno de sus punto fuertes es la evaluación de la calidad del experimento, pudiendo detectar diferentes tipos de errores técnicos que sirven a los investigadores para saber qué ha podido fallar o qué se puede mejorar en su experimento. Entre estos se pueden destacar errores en la intensidad de la señal, en el lavado, la hibridación, las agujas de impresión, el marcaje de los fluoróforos y la generación de replicas de los datos. Esto es una gran ventaja con respecto a otras herramientas, en las que se obtienen los GED sin conocer la calidad del experimento, y en las que se podrían llegar a obtener unos resultados influenciados por las variaciones técnicas.

Además, el hecho de que MADE4-2C realice un análisis automático, facilita su utilización por usuarios inexpertos. Por lo general, otras herramientas para el análisis de micromatrices de dos colores, ofrecen varios métodos de corrección del fondo y normalización al usuario, que debe ser quién elija que método se ajusta mejor a su experimento, independientemente de sus conocimientos, y a pesar de que esta decisión afecta profundamente a los resultados obtenidos [187, 98, 45]. Con MADE4-2C se utiliza automáticamente `normexp` para la corrección del fondo, por ser el mejor valorado en la bibliografía, y se ponen a prueba varios métodos de normalización de modo independiente. Se seleccionan de forma objetiva mediante pruebas matemáticas los métodos que alteran menos los datos originales y eliminan la mayoría de las desviaciones técnicas.

En cuanto a la detección de GED, la mayoría de las herramientas se basan únicamente en un método paramétrico. Sin embargo, en MADE4-2C se aplican dos métodos absolutamente diferentes: uno paramétrico y otro no paramétrico (aunque el usuario puede quedarse con el resultado de solo uno de ellos si lo desea), que junto con la utilización de varios métodos de normalización, sirven para obtener GED más fiables, como se ha propuesto en [187].

Finalmente, otra gran aportación de MADE4-2C es la generación de un informe detallado (apéndice B), que muestra al usuario paso a paso, la evaluación de la calidad del experimento y el análisis realizado sobre sus datos. Hay que resaltar que MADE4-2C es el único programa de este tipo que devuelve un informe con el análisis del experimento. Lo que es de gran utilidad, ya que cuando los datos son de mala calidad, se pone de manifiesto de forma clara en muchas de las figuras del informe. Un ejemplo de esto se muestra en el análisis de hibridaciones de *Abies pinsapo* con el Pinarray1, incluido en el próximo apartado, en el que se ilustra con 4 figuras que el experimento contiene errores graves.

Por otro lado, centrándose en las limitaciones de MADE4-2C, la más importante posiblemente es, que no soporta contrastes múltiples, es decir, que solo está preparado para comparar dos muestras entre sí.

9.2.8. Usos de MADE4-2C

En la tabla 9.1 se resumen los genes expresados diferencialmente de los experimentos de micromatrices que fueron realizados en nuestro grupo de investigación y analizados con MADE4-2C. Los resultados de estos experimentos han ayudado a conocer un poco mejor el transcriptoma de pino en diversas condiciones. Los datos de las cuatro primeras filas muestran que ambos métodos, paramétrico y no paramétrico, `limma` y `Rank Products` respectivamente, son capaces de encontrar muchos genes expresados diferencialmente en común (columna `Ambos` de la tabla 9.1), a pesar de utilizar estrategias completamente diferentes. Esto demuestra que ambos métodos, que son ampliamente reconocidos como fiables por separado [30, 95, 102, 205], al aplicarlos conjuntamente nos aportan GED aún más fiables. Además, los conjuntos de genes candidatos a GED que se han podido detectar únicamente por una de las dos estrategias aportan GED de los que no se dispondría si, como ocurre en la mayoría del software para análisis de micromatrices, sólo se utilizase una estrategia para la detección de los GED. Los resultados de los experimentos de madera madura y juvenil, madera de compresión y opuesta, de las líneas transgénicas que sobreexpresan glutamina sintetasa, y el de aporte y déficit de carbono y nitrógeno, fueron puestos a disposición de los investigadores encargados de ellos, y se encuentran en proceso de estudio para su aplicación o publicación.

En el experimento de aporte y déficit de carbono y nitrógeno se buscaron además genes en común entre las condiciones mostradas en sus tres primeras filas. En los tratamientos con déficit de nitrógeno y carbono (–C–N) y déficit de carbono únicamente (–C+N) se detectaron 6 genes en común en las condiciones de control (C0), y 7 en el tratamiento (C1), perteneciendo dos de estos últimos a genes del metabolismo del nitrógeno impresos en el Pinarray1. Sin embargo, el caso de déficit de nitrógeno (+C–N), únicamente mostró un GED, de función desconocida, en común con respecto a (–C–N). Todo esto sugiere que en el Pinarray1 quizá no se encuentren suficientes genes de estas vías metabólicas, y que por eso se obtienen tan pocos candidatos.

En el experimento de base y ápice expuesto a diferentes concentraciones de amonio (NH_4) se realizó un estudio de los grupos de genes con patrones de expresión similares, utilizando maSigPro [50], en función de las diferentes concentraciones de NH_4 (0 mM, 3 mM y 10 mM), como una serie temporal (datos no mostrados). Estos datos, junto con los GED obtenidos en el análisis de base y ápice (tabla 9.1) analizados funcionalmente con FatiScan fueron de utilidad a los investigadores para demostrar que el ápice de los pinos marítimos (*Pinus pinaster*) es extremadamente sensible a las condiciones de exceso o deficiencia de amonio, y que esto podría servir para detectar los primeros síntomas por estrés de nitrógeno e incrementar la tasa de crecimiento de los pinos jóvenes [35].

Con MADE4-2C se analizaron también datos de otros grupos de investigación:

- Uso del Pinarray1 para analizar brotes, hojas

Tabla 9.1: *Genes expresados diferencialmente en los experimentos de nuestro grupo de investigación analizados con* MADE4-*2C. Todas las dianas procedían de* Pinus pinaster *y todas las hibridaciones se realizaron con Pinarray1, salvo en los casos marcados con un asterisco, en las que se utilizó una micromatriz diferente en la que se habían impreso solo clones obtenidos de un experimento de hibridación sustractiva.*

Experimento	Condicion 0			Condicion 1			FatiScan	maSigPro
	L	RP	Ambos	L	RP	Ambos		
Madera madura y juvenil								
madura (C0) - juvenil (C1)	74	108	74	163	148	148	si	si
madura (C0) - juvenil (C1)*	118	158	118	269	219	218	no	si
Madera de compresión y opuesta								
Comp. (C0) - Op. (C1)	283	263	260	132	186	132	si	si
Comp. (C0) - Op. (C1)*	502	347	347	286	288	281	no	no
Análisis de líneas transgénicas								
WT (C0) - línea 1 (C1)	11	-	-	4	-	-	no	no
WT (C0) - línea 18 (C1)	0	-	-	0	-	-	no	no
Aporte y déficit de carbono y Nitrógeno								
+C+N (C0) - –C–N (C1)	14	-	-	29	-	-	si	no
+C+N (C0) - –C+N (C1)	8	-	-	15	-	-	si	no
+C+N (C0) - +C–N (C1)	1	-	-	0	-	-	si	no
–C–N (C0) - –C+N (C1)	0	-	-	5	-	-	si	no
–C–N (C0) - +C–N (C1)	2	-	-	1	-	-	si	no
Base y ápice								
Base (C0) - ápice (C1)	26	-	-	44	-	-	si	no
Base y ápice con y sin aporte de amonio 0, 3 y 10mM								
Base: 0 mM (C0) - 3 mM (C1)	0	-	-	0	-	-	si	si
Base: 3 mM (C0) - 10 mM (C1)	5	-	-	3	-	-	si	si
Ápice: 0 mM (C0) - 3 mM (C1)	4	-	-	1	-	-	si	si
Ápice: 3 mM (C0) - 10 mM (C1)	1	-	-	10	-	-	si	si
Ápice y base: 0 (C0) - 3 (C1)	0	-	-	0	-	-	si	si
Ápice y base: 3 (C0) - 10 (C1)	0	-	-	1	-	-	si	si

L: Método paramétrico incluido en el paquete de Biocondictor `limma`. RP: método no paramétrico incluido en el paquete de R `Rank Products`; los experimentos sin resultados en RP se analizaron con una versión anterior de MADE4-2C que aún no incluía esta opción. En *ambos* se muestra el número de genes que se clasificaron como GED en ambos métodos a la vez.

y xilema procedentes de varias poblaciones de *Abies pinsapo* y tratados con y sin fosfato. En el caso de los análisis de brotes y hojas, MADE4-2C detectó (1) correlación negativa entre las réplicas (figura 9.9); (2) avisó de que la intensidad de la señal era muy parecida a la del fondo (9.1-B), probablemente debido a la distancia entre la secuencia de los genes de ambas especies; (3) encontró que las réplicas técnicas mostraban un orden erróneo en los arboles de distancias 9.5; y (4) mostró que la distribución del valor de P no seguía el perfil esperado (figura 9.8). Por eso no extrañó que no apareciera ningún GED. Pero gracias a los análisis de MADE4-2C se pudieron corregir estos errores repitiendo las hibridaciones (Datos no mostrados) y fueron de utilidad en la tesis de M. C. Blanes.

- Uso del Pinarra1 con individuos de *Pinus halepensis* sometidos a estrés hídrico durante 14, 28 y 35 días e hibridados frente a un grupo control. Los datos se pusieron a disposición de los investigadores que realizaron los experimentos y aún están pendientes de resolución.

- Dos experimentos en los que se comparaba la expresión de dos cepas de *Sinorhizobium meliloti* (apartado 8.2, pág. 61), y que fueron de utilidad para la tesis de J. A. López-Contreras.

9.3. Identificación de una muestra problemática

En un experimento realizado por S. Díaz-Moreno en nuestro grupo de investigación (datos sin publicar) se observó un comportamiento inesperado en uno de los individuos analizados. En el experimento se comparaba la madera madura y juvenil, y se utilizaron cuatro réplicas biológicas, una por individuo utilizado, y 2 réplicas técnicas de cada una de ellas. Las muestras utilizadas procedían de cuatro árboles adultos de *Pinus pinaster* de poblaciones

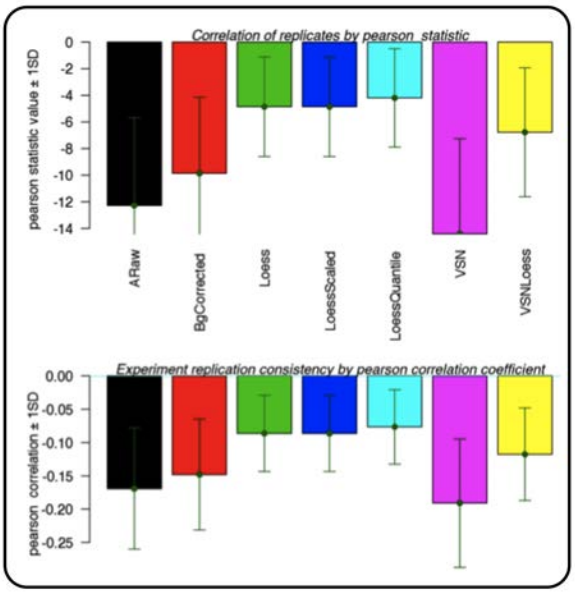


Figura 9.9: *Correlación negativa de las réplicas detectada en los experimentos de brotes y hojas de pinsapo.*

naturales de Sierra Bermeja (Málaga), que se hibridaron con el Pinarray1 y con una micromatriz con secuencias de pino obtenidas por hibridación sustractiva por supresión, llamada SSH-Ma (apartado 8.1). A continuación se presenta el diseño del experimento y los datos obtenidos al hibridrar con SSH-Ma por ser donde se observó este comportamiento originalmente. Las réplicas del experimento se organizan del siguiente modo:

- **Individuo 1-Sur**, hibridado en la micromatriz 10a marcando la muestra de madera madura con Cy3 y la de madera juvenil con Cy5. La micromatriz se dividió en dos réplicas técnicas 10a-A y 10a-Z.
- **Individuo 1-Norte**, hibridado en la micromatriz 22a marcando la muestra de madera madura con Cy3 y la de madera juvenil con Cy5. La micromatriz se dividió en dos réplicas técnicas 22a-A y 22a-Z.
- **Individuo 2-Norte**, hibridado en la micromatriz 23a, con intercambio de fluoróforos en relación a las hibridaciones anteriores, marcando la muestra de madera madura con Cy5 y la de madera juvenil con Cy3. La micromatriz se dividió en dos réplicas técnicas 23a-A y 23a-Z.
- **Individuo 3-Sur**, hibridado en la micromatriz 24a, con intercambio de fluoróforos en relación a las dos primeras micromatrices, marcando la muestra de madera madura con Cy5 y la de madera juvenil con Cy3. La micromatriz se dividió en dos réplicas técnicas 24a-A y 24a-Z.

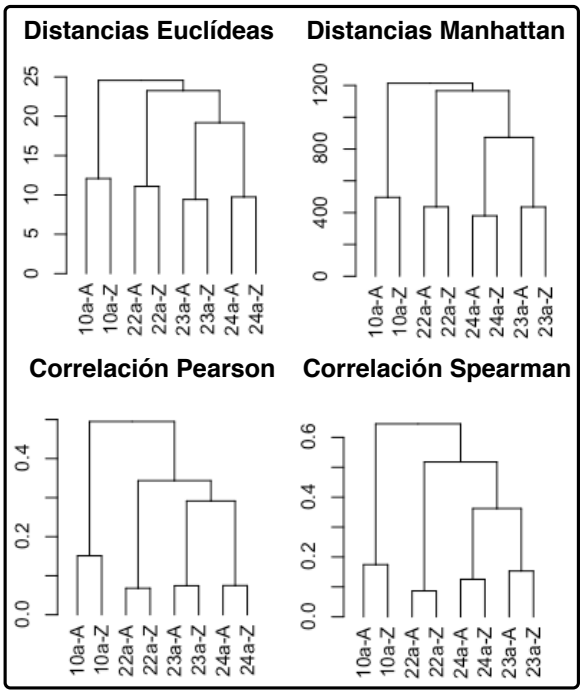


Figura 9.10: *Dendrogramas con las distintas distancias y correlaciones de las diferentes hibridaciones realizadas. Las micromatrices 10a-A, 10a-Z son réplicas técnicas del individuo 1-Sur (más detalles en el texto).*

En el análisis realizado con MADE4-2C (resultados no mostrados), al evaluar la correlación y distancias entre las muestras, se comprobó que las réplicas técnicas se agrupaban adecuadamente (figura 9.10), lo que sugería que el experimento estaban bien hecho. Pero en el caso de las réplicas biológicas se observó que el individuo 1-Sur (10a), en lugar de quedar emparejado con el individuo 1-Norte (22a) que llevaba el mismo marcaje, aparecía separado del resto de individuos en los cuatro dendrogramas (figura 9.10). Estos resultados nos hicieron plantearnos si cada gen candidato presentaba el mismo comportamiento en los distintos individuos. Como la búsqueda de GED que se realiza con MADE4-2C permite únicamente la comparación de dos situaciones se decidió realizar un análisis de varianza con la librería `maSigPro` [50], que agrupa genes con patrones de expresión similares en una serie temporal, aunque también puede utilizarse cambiando las mediciones de tiempo por otras condiciones (María José Nueda, comunicación personal), que en nues-

tro caso fueron madera juvenil y madera madura. Así pues, se realizó este análisis comparando si los agrupamientos de genes de `maSigPro` se comportaban igual en los 4 individuos del experimento comparando madera madura y madera juvenil como si fueran el tiempo 0 y el tiempo 1, respectivamente (figura 9.11).

Como entrada de `maSigPro` se emplearon los datos normalizados en formato maNorm que previamente se habían guardado con MADE4-2C. Los agrupamientos obtenidos con `maSigPro` confirmaron el análisis de correlación y distancia anteriores, ya que se veía claramente que sus genes no se comportaban como los del resto de individuos. Así pues, en el agrupamiento 1 (Cluster1, figura 9.11-A), correspondiente a los genes que se expresan más en madera juvenil que en madura, los valores medios de expresión de los genes en el individuo 1-Sur eran mayores que en el resto de individuos, si bien su pendiente era paralela a los del individuo 1-Norte, que lleva el mismo marcaje. En el agrupamiento 2 (Cluster2, figura 9.11-A), los valores de expresión de los genes del individuo 1-Sur eran claramente menores que para el resto de individuos, y su pendiente seguía siendo paralela al individuo 1-Norte. En el agrupamiento 3 (Cluster3, figura 9.11-A), correspondiente a los genes que se expresan más en juvenil que en madura, todos los individuos mostraban un patrón en el que había mayor expresión de esos genes en madera madura que en juvenil, aunque la pendiente para el individuo 1-Sur era diferente a la del resto.

Tabla 9.2: *Genes expresados diferencialmente entre madera juvenil y madura en dos micromatrices diferentes hibridadas con los cuatro árboles comentados en este apartado. Se presenta el número de GED analizando las hibridaciones con y sin el individuo 1-Sur.*

Micromatriz	GED	Madura	Juvenil
SSH-Ma+1S	220	74	146
SSH-Ma−1S	265	67	198
Pinarray+1S	206	70	136
Pinarray−1S	223	72	151

+1S: con el individuo 1-Sur; −1S: sin el individuo 1Sur; GED: genes expresados diferencialmente

Los mismos datos de expresión normalizados se volvieron a analizar descartando esta vez el individuo 1-Sur. En este caso, MADE4-2C reveló más genes expresados diferencialmente, tanto en conjunto como en cada una de las condiciones (tabla 9.2), gracias a que la ausencia del individuo 1-Sur implicaba la retirada de una muestra que introducía una variabilidad artefactual y que restaba potencia estadística al análisis. Al analizar también los mismos datos con `maSigPro` sin el individuo 1-Sur (figura 9.11-B), aparecieron grupos con un comportamiento más homogéneo entre los individuos que antes, aunque el número de genes que contenía cada grupo mostró bastantes diferencias. Estas diferencias se deben a que `maSigPro` simplemente agrupa genes que siguen un mismo patrón, y esto no quiere decir que esos genes se expresen diferencialmente. De hecho, tomando como ejemplo el *Cluster1* de la figura 9.11-A, los individuos representados por las líneas turquesa, azul y verde, apenas varían su expresión entre madera madura y juvenil. Sin embargo, en el *Cluster1* obtenido al descartar el individuo 1-Sur (figura 9.11-B), se observa un cambio mucho más pronunciado en la expresión entre madura y juvenil (obsérvese la pendiente de las líneas que representan los individuos y la escala del eje y). Por lo tanto, aunque en total hay menos genes candidatos, el resultado es más coherente entre todos los individuos.

El Pinarray1 también se hibridó con los mismos 4 individuos que se acaban de describir, y el análisis equivalente con la ayuda de `maSigPro`, confirmó también que el individuo 1-Sur no tenía el mismo comportamiento que los demás (resultados no mostrados). Por eso, también se obtuvieron más GED cuando no se usaba el individuo 1-Sur (tabla 9.2). Este comportamiento anómalo del individuo 1-Sur se confirmó finalmente en el laboratorio, con análisis de expresión por RT-PCR y microscopía (S. Díaz, comunicación personal).

Por otro lado, si se comparan los valores de este experimento sin el individuo 1-Sur (tabla 9.2) con los mostrados en la tabla 9.1 (columnas L dentro de *Condición* 0 y 1), que también se obtuvieron sin utilizar el individuo 1-Sur, se puede observar diferencias en el número de GED detectados. Estas diferencias se deben a la aplicación del parámetro `uMinFC` de MADE4-2C (véase el apéndice D), que descarta las sondas con valores de veces de cambio cercanos a $M = 0$ (comentado en el apartado 9.2.5). Este parámetro se basa en que se ha descrito que hay una gran varianza en los valores de expresión cercanos a cero [62], y que la eliminación de sondas altamente variables del experimento, además de suponer un enriquecimiento en sondas con expresión diferencial, aumenta la fuerza estadística del análisis y mejora la detección de GED [45]. Al aplicar el parámetro `uMinFC` en el experimento de madera madura frente a madera juvenil que fue hibridado con el Pinarray1 (tabla 9.1, fila *madura (C0) - juvenil (C1)*, columna L en condición 0 y 1) se detectaron 74 GED en madera madura y 163 en madera juvenil, y en el caso de la tabla 9.2 (experimento Pinarray−1S), en el que no se aplica el parámetro,

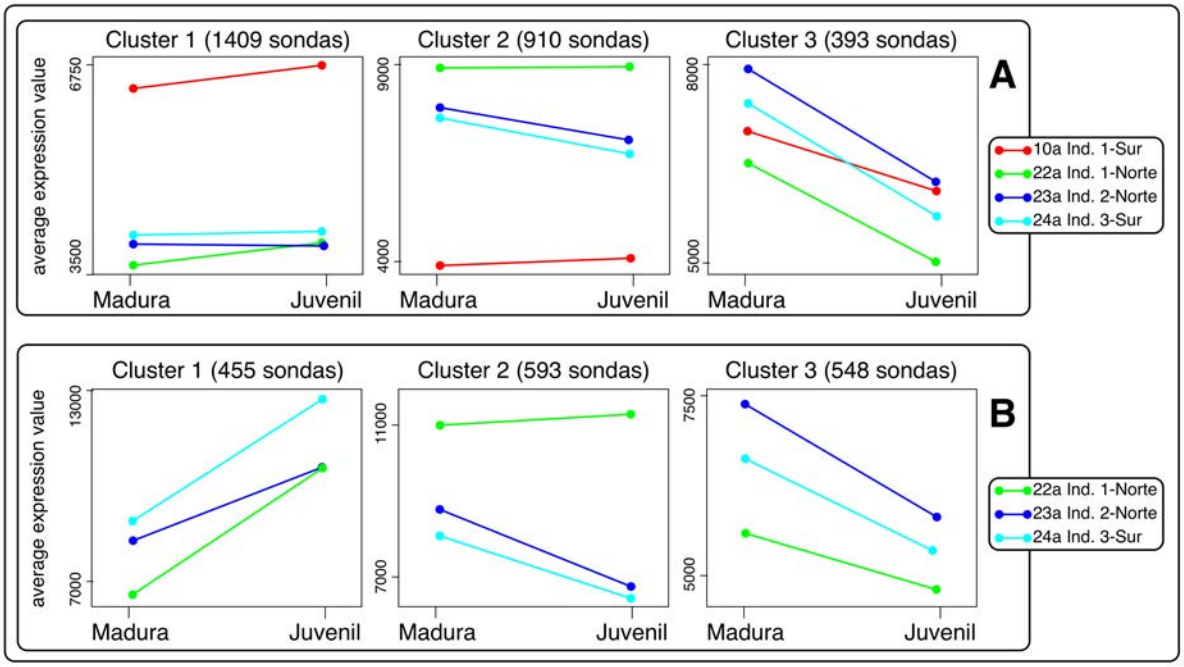


Figura 9.11: *Gráficas de los agrupamientos formados por* `maSigPro` *con el individuo 1-Sur (A) y sin él (B). Obsérvese que, al menos en los agrupamientos 1 y 2, los genes del individuo 1-Sur siguen claramente una distribución diferente a la de los demás individuos*

se detectaron 72 GED en madera madura, y 151 en juvenil. Por tanto, al aplicar el parámetro `uMinFC` se detectaron 2 GED más en madera madura y 12 más en juvenil. De igual modo, en la hibridación con SSH-Ma, en el análisis que se aplicó este parámetro (tabla 9.1, fila *madura (C0) - juvenil (C1)**, columna L en condición 0 y 1) se detectaron 51 GED más en madera madura y 71 en juvenil, en comparación con el resultado de SSH-Ma−1S mostrado en la tabla 9.2, en el que no se aplica el parámetro. La diferencia obtenida es mucho mayor en SSH-Ma que en el Pinarray1 porque al ser una micromatriz creada a partir de una genoteca SSH, la expresión diferencial es mas fuerte, y al quitar el individuo incoherente se detectan aún más candidatos.

En los resultados obtenidos con Pinarray1, gracias a tener el fichero GAL anotado, se pudo conocer al instante los posibles productos de los genes expresados diferencialmente, además de poder realizar un análisis funcional con FatiScan, incluido en el siguiente apartado.

9.4. Análisis funcional de la madera juvenil y madura

A pesar de que la mayor parte de los investigadores buscan GED en los análisis con micromatrices, la variabilidad biológica y metodológica hace que cada vez que se repiten experimentos con micromatrices se obtengan listas de GED diferentes. Sin embargo, varios estudios han demostrando que, a pesar de la baja concordancia de los genes, las funciones biológicas encontradas sí que se repiten [6, 60]. Esto tiene su lógica, ya que es bien conocido que los genes no actúan de modo independiente en las células, sino que forman parte de una compleja red de interacciones. Además, también es ampliamente aceptado que los genes que se coexpresan pueden tener funciones comunes en la célula y estar situados, a menudo, en regiones cercanas del genoma [6]. Es por tanto de esperar que sean las funciones biológicas las que cambian en las diferentes condiciones del experimento, ya que los cambios de expresión no afectan únicamente a genes concretos de modo individual, sino que son rutas enteras las que se activan o desactivan para reaccionar a los cambios. Lamentablemente, la mayoría de los programas que permiten análisis funcionales, sólo se pueden usar cuando se trabaja con especies modelo, salvo FatiScan [6] (`http://babelomics3.bioinfo.cipf.es`), que permite analizar organismos para los que las anotaciones están personalizadas u obtenidas en el propio laboratorio. Puesto que el pino no está entre los organismos modelo habituales, era el programa que más se adecuaba a las necesidades de nuestro grupo de investigación. Por eso, cuando se utiliza el Pinarray1, MADE4-2C genera los dos ficheros necesarios para realizar un análisis funcional de la expresión génica con FatiScan. Uno contiene los va-

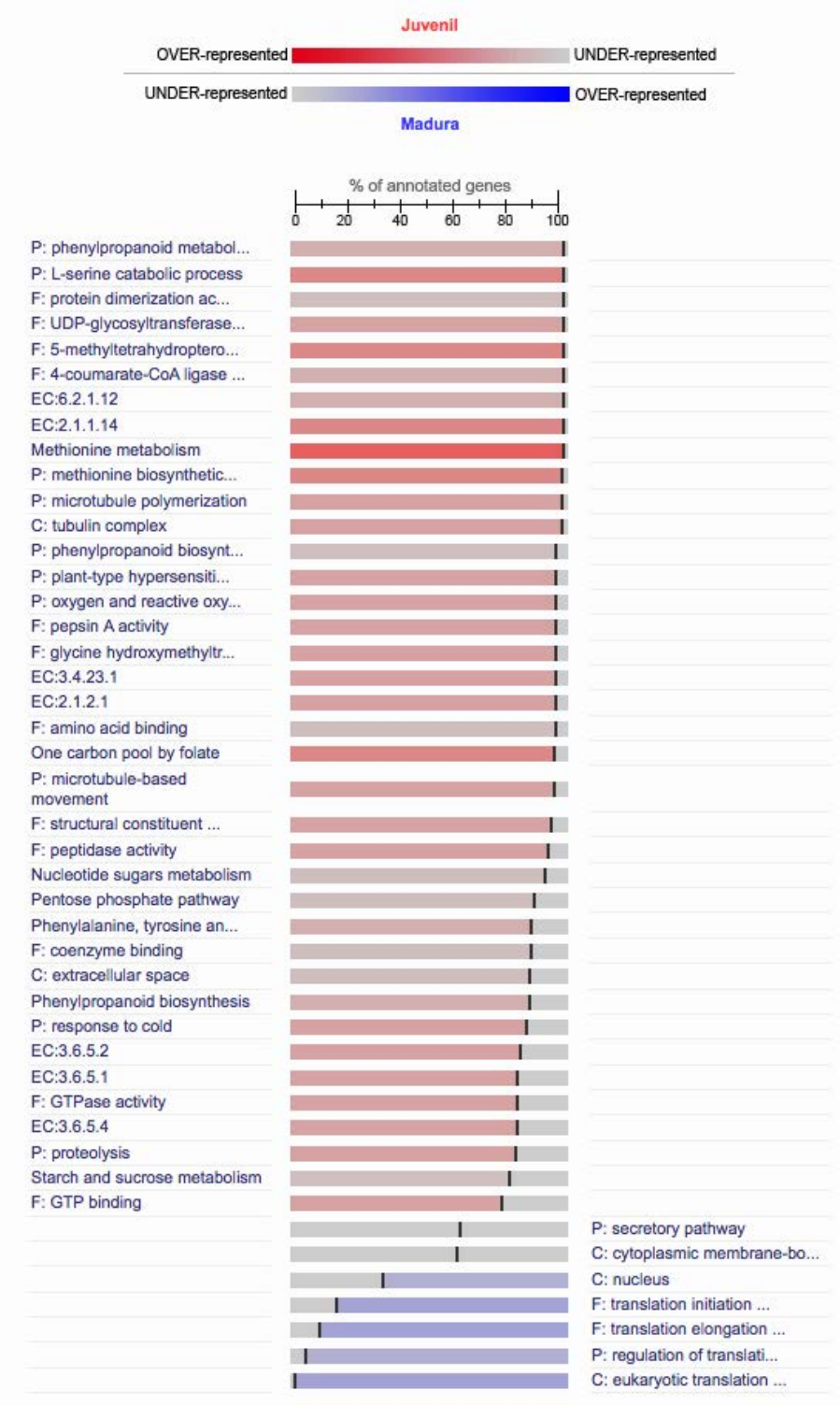


Figura 9.12: *Anotación funcional realizada con FatiScan para el experimento de micromatrices que compara madera juvenil frente a madera madura en la hibridación con el Pinarray1. P: Procesos Biológicos, C: Componentes Celulares, M: Funciones Moleculares, EC: Enzimas. El resto de descripciones corresponden a rutas metabólicas KEGG*

lores de t de Student ordenados de mayor a menor para cada una de las sondas de la micromatriz. El otro fichero contiene las anotaciones de cada una de las sondas impresas en la micromatriz, principalmente términos de la *Gene Ontology*, junto con los códigos de las enzimas y las rutas metabólicas caracterizadas en KEGG (apartado 7.2.4).

Como con las hibridaciones realizadas con el Pinarray1 se puede realizar un estudio funcional, gracias a que se cuenta con el fichero GAL anotado (véase el apartado 9.1), se decidió utilizar FatiScan con los datos del experimento de madera juvenil frente madera madura llevado a cabo por S. Díaz-Moreno [58], para conocer las funciones biológicas que cambiaban diferencialmente en ambos tejidos. En los resultados obtenidos con FatiScan (figura 9.12) se observa que la madera juvenil (que presenta el doble de GED que la madura, tanto en la tabla 9.1 como en la tabla 9.2) muestra mayor alteración de las funciones biológicas que la madera madura. Esto se podría explicar por el hecho de que la madera madura se encuentra en un estado de mantenimiento (expresa posiblemente genes que sirven para seguir viva; por eso se obtienen funciones de secreción y traducción, principalmente), mientras que la madera juvenil, que se supone que está en crecimiento y desarrollo intenso, muestra un número significativamente mayor de funciones biológicas, muchas de ellas relacionadas con la síntesis de madera, de celulosa, y con el metabolismo de los microtúbulos. Se ve claramente que hay muchas más funciones en la madera juvenil que en la madura (figura 9.12), más del doble, que es la proporción que se observó en el análisis de genes expresados diferencialmente. Lo que indica que los genes de madera juvenil están repartidos en un mayor número de funciones, y que los GED de la madera madura están concentrados en muy pocas funciones.

Capítulo 10

Análisis del transcriptoma del pino

El genoma del pino no se ha secuenciado, y el grupo de las *angiopermas*, donde es posible encontrar genomas de referencia de árboles, es una referencia demasiado lejana, ya que ambos grupos se diversificaron hace unos 200-300 millones de años [65]. Además, los árboles incluidos en el grupo de las coníferas tienen tanto en común con los árboles de *angiopermas* como con las plantas herbáceas de este mismo grupo. Esto se debe al hecho de que la formación del xilema secundario, es decir, el modo de formar la madera, ha aparecido y desaparecido en linajes de plantas independientes a lo largo de la evolución [90, 186]. Por lo tanto, siendo los posibles genomas de referencia tan lejanos, y con lo complejo y grande que es el genoma de pino (véase el apartado 1.2 de la introducción, pág. 3), se intuye que una buena aproximación para conocer los genes del pino pasa por conocer primero su transcriptoma. Como una de las aplicaciones más importantes de la tecnología NGS es la caracterización de transcriptomas de especies no modelo [113], a lo largo de este capítulo se describirá cómo obtener el transcriptoma de pino desde las secuencias que se obtienen con el secuenciador automático hasta la anotación de los unigenes obtenidos.

10.1. Preprocesamiento

Como se ha comentado en la introducción (apartado 3.6.1, pág. 25), el preprocesamiento es un paso esencial que puede afectar drásticamente a pasos posteriores como el ensamblaje. En este trabajo se han utilizado secuencias obtenidas por el método de Sanger y por NGS de tipo pirosecuenciación (plataforma 454/FLX de Roche). Como ambos tipos de secuencias necesitan diferentes tratamientos en el preprocesamiento y en el ensamblaje, a continuación se abordará el preprocesamiento del transcriptoma de pino en ambas tecnologías.

En nuestro laboratorio se han diseñado dos algoritmos para el preprocesamiento de secuencias, tanto transcriptómicas como genómicas. Uno de ellos es SeqTrim (apartado 10.1.1), con el que se procesan lecturas de tipo Sanger y se pueden preprocesar también muchas de las obtenidas por NGS (como se verá), aunque no resulta práctico y no es capaz de manejar lecturas pareadas. Su adaptación y completa reescritura para adaptarse al volumen y características de las lecturas de NGS llevó al desarrollo de SeqTrimNext (apartado 10.1.2).

10.1.1. De lecturas de tipo Sanger

Las secuencias de tipo Sanger, por su alto coste y mayor dedicación de tiempo en el laboratorio, no suelen suponer una carga de trabajo mayor de unas pocas de miles secuencias en los proyectos de secuenciación de EST. Por este motivo, SeqTrim es un programa que se diseñó para recuperar el inserto clonado utilizando una sola CPU. Sin embargo, en los casos en los que se necesitó analizar decenas de miles de secuencias, como fue el caso en el que se analizó una librería de Roche-454 con 951 641 lecturas (véase el artículo de EuroPineDB, apartado 11.1, pág. 139), se realizó la ejecución en paralelo utilizando un *array-job* para acelerar el proceso (véase el apartado 7.3.12, pág. 57). Sea de un modo o de otro, SeqTrim recorta las secuencias eliminando fragmentos de vector, adaptadores, secuencias de baja calidad y elementos ajenos al inserto, descartando las secuencias que muestren una alta similitud con organismos contaminantes.

Mi contribución para el desarrollo de SeqTrim consistió en comparar el programa con otras herramientas existentes, como SeqClean (`http://compbio.dfci.harvard.edu/tgi/software/`), Phred [69, 68], ESTPass [131] y los módulos de limpieza de EGAssembler [150], y probar SeqTrim con una gran variedad de secuencias en diferentes formatos, procedentes de diferentes especies secuenciadas con diferentes protocolos (lo que queda reflejado en las figuras 3 y 4, y la tabla 1 del artículo que viene a continuación). También puse a prueba el programa ejecutándolo con el mayor tipo de situaciones posibles, generando secuencias

artificiales (tabla 1 del artículo de SeqTrim), y probando secuencias reales, para poner a prueba y ajustar la capacidad de detectar vectores, adaptadores, contaminantes, colas poli-A y Poli-T, posibles quimeras, y adaptadores y vectores en posiciones inesperadas.

Gracias a estos análisis se ajustó el algoritmo y se corrigieron algunos fallos, como los restos de indeterminaciones en los extremos, los insertos formados únicamente por indeterminaciones o colas Poli-A y poli-T, errores en el punto de corte del vector y el adaptador, o eliminar secuencias con contaminantes en lugar de recortarlas, lo que dejaba restos de contaminantes. Las pruebas realizadas posibilitaron la mejora del programa, sobre todo en relación con:

- la localización de vectores y adaptadores
- la delimitación de los bordes de inserto
- la detección de artefactos y quimeras
- las bases de datos de contaminantes
- la detección de errores de programación en diversas situaciones.

El resultado fue un algoritmo más robusto y capaz de resolver un mayor número de situaciones correctamente. Prueba de ello ha sido su popularidad, ya que el artículo que se adjunta recibió las etiqueta de *highly accessed* de BMC (`http://www.biomedcentral.com/1471-2105/11/38/about#accesses`) por haber sido consultado más de 7500 veces desde que se publicó en 2010. Además, en la web de SeqTrim se han contabilizado hasta 445 usuarios, aunque hay que tener en cuenta que los proyectos de mayor tamaño y los usuarios avanzados suelen realizar la ejecución de SeqTrim a través de la terminal de comandos por lo que estos usos no se contabilizan, al igual que los de los usuarios que tienen una instalación propia del programa. A medida que han aumentado los proyectos de NGS, el uso de SeqTrim se ha visto reducido en favor de otros programas como SeqTrimNext. De hecho, hoy en día la web de SeqTrim redirige automáticamente a la de SeqTrimNext, aunque siempre dando la posibilidad acceder a la primera.

Entre los usos de SeqTrim para las necesidades del grupo de investigación se utilizó para preprocesar las secuencias de todos los clones de ADNc incluidas en EuroPineDB (véase el apartado 11.1), y para preprocesar las secuencias impresas en el Pinarray1 en el proceso de anotación de su fichero GAL (apartado 9.1).

A continuación se incluye el artículo donde se describe en detalle el programa.

Falgueras et al. BMC Bioinformatics 2010, 11:38
http://www.biomedcentral.com/1471-2105/11/38

BMC Bioinformatics

SOFTWARE Open Access

SeqTrim: a high-throughput pipeline for pre-processing any type of sequence read

Juan Falgueras[1], Antonio J Lara[2], Noé Fernández-Pozo[3], Francisco R Cantón[3], Guillermo Pérez-Trabado[2,4], M Gonzalo Claros[2,3*]

Abstract

Background: High-throughput automated sequencing has enabled an exponential growth rate of sequencing data. This requires increasing sequence quality and reliability in order to avoid database contamination with artefactual sequences. The arrival of pyrosequencing enhances this problem and necessitates customisable pre-processing algorithms.

Results: SeqTrim has been implemented both as a Web and as a standalone command line application. Already-published and newly-designed algorithms have been included to identify sequence inserts, to remove low quality, vector, adaptor, low complexity and contaminant sequences, and to detect chimeric reads. The availability of several input and output formats allows its inclusion in sequence processing workflows. Due to its specific algorithms, SeqTrim outperforms other pre-processors implemented as Web services or standalone applications. It performs equally well with sequences from EST libraries, SSH libraries, genomic DNA libraries and pyrosequencing reads and does not lead to over-trimming.

Conclusions: SeqTrim is an efficient pipeline designed for pre-processing of any type of sequence read, including next-generation sequencing. It is easily configurable and provides a friendly interface that allows users to know what happened with sequences at every pre-processing stage, and to verify pre-processing of an individual sequence if desired. The recommended pipeline reveals more information about each sequence than previously described pre-processors and can discard more sequencing or experimental artefacts.

Background

Sequencing projects and Expressed Sequence Tags (ESTs) are essential for gene discovery, mapping, functional genomics and for future efforts in genome annotations, which include identification of novel genes, gene location, polymorphisms and even intron-exon boundaries. The availability of high-throughput automated sequencing has enabled an exponential growth rate of sequence data, although not always with the desired quality. This exponential growth is enhanced by the so called "next-generation sequencing", and efforts have to be made in order to increase the quality and reliability of sequences incorporated into databases: up to 0.4% of sequences in nucleotide databases contain contaminant sequences [1,2]. The situation is even worse in the EST databases, where vector contamination rate reach 1.63% of sequences [3]. Hence, improved and user friendly bioinformatic tools are required to produce more reliable high-throughput pre-processing methods.

Pre-processing includes filtering of low-quality sequences, identification of specific features (such as poly-A or poly-T tails, terminal transferase tails, and adaptors), removal of contaminant sequences (from vector to any other artefacts) and trimming the undesired segments. There are some bioinformatic tools that can accomplish individual pre-processing aspects (e.g. TrimSeq, TrimEST, VectorStrip, VecScreen, ESTPrep [4], crossmatch, Figaro [5]), and other programs that cope with the complete pre-processing pipeline such as PreGap4 [6] or the broadly used tools Lucy [7,8] and SeqClean [9]. Most of these require installation, are difficult to configure, environment-specific, or focused on specific needs (like a design only for ESTs), or require a change in implementation and design of either the program or the protocols within the laboratory itself.

* Correspondence: claros@uma.es
[2]Plataforma Andaluza de Bioinformática, Universidad de Málaga, 29071 Málaga, Spain

10.1.2. De lecturas de nueva generación

La diferencia principal entre las tecnologías clásicas y nuevas es la cantidad de datos generados, es decir, el número de lecturas producido. Mientras que con la tecnología clásica se producían miles de secuencias, con las nuevas se pueden obtener millones, y eso se traduce en una cantidad de datos mayor en varios órdenes de magnitud. Además, con estas tecnologías se obtienen las secuencias aplicando diferentes pasos de preparación en el laboratorio (apartado 3.4, pág. 20), por lo que se introducen nuevos elementos en la secuencia que hay que eliminar, pues se trata de elementos para los que los programas desarrollados para el preprocesamiento de secuencias de tipo Sanger no están preparados. Así pues, SeqTrim no resulta adecuado para las necesidades de las secuencias de NGS y se tuvo que desarrollar SeqTrimNext, que es especialmente útil para lecturas de Roche-454, y también está adaptado al preprocesamiento de otras tecnologías como Illumina o SOLiD.

SeqTrimNext incluye fases de preprocesamiento específicas para las tecnologías de NGS, entre las que se encuentran retirar los MID y las claves (*keys*), detectar y extraer las secuencias pareadas con más eficacia que ningún otro programa, o eliminar lecturas clonales. Támbién se le ha introducido la capacidad de paralelizar o distribuir el trabajo para acelerar el tratamiento de los millones de secuencias que se obtienen con la NGS. Finalmente, es capaz de devolver las lecturas preprocesadas útiles clasificadas según los MID que se utilizaron.

SeqTrimNext está desarrollado en Ruby utilizando una estructura de *plugins*, de modo que es fácil añadir nuevas funciones, reordenar el momento en el que se ejecutan las existentes y personalizar el preprocesamiento al elegir qué pasos realizar y cuáles omitir. Por ejemplo, se puede eliminar únicamente las claves y recortar por calidades, lo que sería muy rápido, o ejecutar SeqTrimNext con más *plugins* (análisis de la calidad, secuencias pareadas, contaminantes, adaptadores, MIDs, claves, baja complejidad, etc.) para obtener el preprocesamiento más fiable, aunque requiera más tiempo. Puesto que la combinación de *plugins* puede resultarle complicada al usuario, y dado que cada experimento puede requerir un preprocesamiento diferente, SeqTrimNext permite personalizar la ejecución al utilizar plantillas en las que se indican los *plugins* que se desean utilizar y el orden correspondiente. Así pues, un usuario que analice siempre el mismo tipo de secuencias puede crear una plantilla de ejecución que se ajuste a sus necesidades y utilizarla tantas veces como desee, sin necesidad de configurar el programa cada vez que lo ejecuta. Entre las plantillas incluidas en la web de SeqTrimNext (`http://www.scbi.uma.es/seqtrimnext`) se pueden encontrar algunas diseñadas para preprocesar datos genómicos de 454 con y sin pareadas, datos transcriptómicos de 454, amplicones y para un preprocesamiento rápido de las secuencias de baja calidad y complejidad.

Mi contribución al desarrollo de SeqTrimNext consistió en reunir las posibles secuencias contaminantes, junto con un *script* para descargarlas y formatearlas en la instalación del programa o cuando se desee actualizarlas. También desarrollé un *script* que construye el informe personalizado que aparece al finalizar la ejecución de SeqTrimNext y que contiene un resumen de cómo fue el preprocesamiento para que el usuario se haga una idea de lo útil que resultará su análisis. Ambos aspectos se cuentan con más detalle a continuación, antes de ofrecer el manuscrito en el que se describe SeqTrimNext, ya que no se describen con suficiente detalle en el mismo.

Detección de contaminantes

En cualquier laboratorio se convive con una gran cantidad de microorganismos que pueden acabar contaminando una muestra. Por ejemplo, *Delftia* es un contaminante frecuente en los reactivos [67]. También se consideran contaminantes los restos de tejidos humanos procedentes de los investigadores, así como las secuencias de ADN de los microorganismos utilizados con frecuencia en el laboratorio, como *E.coli* [63] y *Agrobacterium tumefaciens* [110]. Los hongos del polvo domestico también pueden contaminar muestras, siendo los más frecuentes *Penicillum sp., Aspergillus sp.* y las levaduras [77]. Además, es habitual encontrar secuencias de ARNr como contaminante por su alto número de copias, como se muestra en un estudio en el que aparecen ARNr de *Duganella, Acinetobacter, Stenotrophomonas, Escherichia, Leptothrix*, y *Herbaspirillum* al amplificar por PCR fragmentos de ADNr 16S sin haber añadido ADN molde al que puedan pegarse los cebadores y amplificarlo [216].

Otra fuente de contaminantes pueden ser los orgánulos del organismo de estudio (cloroplastos y mitocondrias) y las poblaciones microbianas que crecen sobre dicho organismo [213], algo muy frecuente en las muestras del intestino animal, o de las raíces de las plantas, de las que no es fácil eliminar completamente los restos de la rizosfera. El gran problema es que todas estas contaminaciones puede dificultar el posterior ensamblaje, o lo que es peor, incorporarse en las secuencias finales como si fueran secuencias del organismo en estudio.

Por estos motivos se ha dotado a SeqTrimNext

de un conjunto de genomas de organismos contaminantes en potencia, como bacterias, hongos y humano, además de orgánulos y ARNr. Se utiliza genómico y no ARNm porque cualquier fragmento de secuencia puede contaminar un experimento, y no únicamente las secuencias que se expresan. Las secuencias proceden del National Center for Biotechnology Information (NCBI) (`ftp://ftp.ncbi.nih.gov/genomes/`), a excepción de las secuencias de ARNr, que provienen del repositorio arb-silva `ftp.arb-silva.de`. La colección de contaminantes se puede descargar al ejecutar el *script* de descarga de contaminantes (apéndice E, pág. 257) que se incluye en la instalación de SeqTrimNext. Como estas colecciones de contaminantes no son redundantes se pueden combinar, de modo que el usuario pueda elegir las que más se adapten a su experimento. SeqTrimNext también ofrece la posibilidad de añadir un fichero en el que el investigador especifique las secuencias que quiere considerar contaminantes.

Como es muy probable que no todos los usuarios se vean capacitados para personalizar la colección de contaminantes, y usar sistemáticamente todas ellas enlentece mucho el proceso (datos no mostrados), se han preparado dos colecciones de contaminantes diferentes. La que se utiliza por omisión contiene una selección de los contaminantes más corrientes, y recibe el nombre de `core_bacteria_fungi`. Se procuró que contuviera el mayor número de familias diferentes de hongos y bacterias, sin que el tamaño del conjunto se disparara. De todas formas, añadir o eliminar algún representante de esta selección de contaminantes `core_bacteria_fungi` es fácil: basta con añadir o eliminar su nombre del fichero `core_seqs.txt`, ya que el *script* de descarga de contaminantes consulta este fichero para montar la colección de secuencias. A continuación se detallan las especies que contiene por omisión:

Bacterias:

- *Acidaminococcus fermentans*
- *Acinetobacter baumannii*
- *Agrobacterium tumefaciens*
- *Bacillus subtilis*
- *Bacteroides fragilis*
- *Chlamydia trachomatis*
- *Clostridium botulinum*
- *Delftia acidovorans*
- *Escherichia coli K12 substrain DH10B*
  Se utiliza esta cepa de *E.coli* por ser una de las más utilizadas para la clonación [63]. De hecho, cuando se utilizó otra, los contigs resultantes aún podían contener alguna secuencia de este microorganismo.
- *Haemophilus influenzae*
- *Herbaspirillum seropedicae*
- *Leptothrix cholodnii*
- *Mycobacterium tuberculosis*
- *Mycoplasma hominis*
- *Pseudomonas aeruginosa*
- *Rickettsia rickettsii*
- *Staphylococcus aureus*
- *Stenotrophomonas maltophilia*
- *Streptococcus mutans*

Hongos:

- *Aspergillus niger*
- *Candida albicans*
- *Neurospora crassa*
- *Penicillium chrysogenum*
- *Pichia pastoris*
- *Podospora anserina*
- *Saccharomyces cerevisiae*
- *Schizosaccharomyces pombe*

En caso de que se desee realizar una búsqueda de contaminantes más exhaustiva, SeqTrimNext dispone de otros ficheros que contienen más organismos contaminantes en potencia. Todos ellos se pueden combinar entre sí o con `core_bacteria_fungi` simplemente añadiéndolos al campo `contaminants_db` de la plantilla, separando los nombres por un espacio:

```
contaminants_db =
   "contaminants.fasta cont_ribosome.fasta"
```

Las colecciones más exhaustivas de contaminantes son:

- **Bacteria:** contiene el genoma de un único representante de cada género disponible de bacterias, excepto los ya utilizados en `core_bacteria_fungi` para que no haya redundancia cuando se combina con ella.

- **Fungi:** contiene el genoma de un único representante de cada género disponible de hongos, excepto los ya utilizados en `core_bacteria_fungi` para que no haya redundancia cuando se combina con ella.
- **human:** contiene el genoma humano. Útil siempre que las secuencias de estudio no sean de humano, ya que los investigadores pueden contaminar las muestras.
- **organelle:** contiene genomas de orgánulos, cloroplastos y mitocondrias. Se debe incluir si el estudio se centra únicamente en los genes nucleares.
- **rrna:** contiene los ARNr descargados de `ftp.arb-silva.de` y posteriormente tratados con CD-HIT (véase el apartado 7.3.7 de Materiales y métodos) para reducir la redundancia de secuencias al 90 %.

Informe del preprocesamiento

Para ofrecer a los usuarios de SeqTrimNext un modo cómodo y sencillo de visualizar cómo funcionó el preprocesamiento de sus secuencias se diseñó un informe en formato PDF (véase el apéndice F, pág. 273). Con el contenido de este informe, los usuarios podrán determinar si la secuenciación fue correcta y el preprocesamiento adecuado, o conocer los motivos por los que la secuenciación o el preprocesamiento han hecho desperdiciar gran parte de las lecturas. Este informe se genera automáticamente al finalizar cualquier ejecución, utilizando el *script* descrito en el apéndice G, página 273.

Para generar el informe se utiliza la información contenida en los ficheros de estadísticas generados por SeqTrimNext (`used_params.txt`, `initial_stats.json` y `stats.json`) junto con otros ficheros (`plugin_nts.json` y `plugin_seqs.json`) que contienen los valores máximos permitidos. Así pues, el informe contendrá mensajes de advertencia en los casos en los que alguno de los elementos analizados aparezca con demasiada frecuencia. Para generar el PDF se necesita LaTeX, por lo que el *script* graba todos los ficheros necesarios divididos entre los que contienen textos fijos (`main.tex`, `input_graph.tex`, `output_files.tex`, `output_graph.tex` y `qv_graph.tex`) y los que contienen información específica del preprocesamiento (`UsedParams.tex`, `stats.tex` y `rejected.tex`), de una forma equivalente a como se hizo en el informe para MADE4-2C (véase apartado 9.2, pág. 66).

El informe de preprocesamiento consta de cuatro apartados:

1. **Output Files**, en el que se describen los ficheros de salida que produce SeqTrimNext.
2. **Relevant parameters**, donde se muestran los valores de los principales parámetros elegidos por el usuario para la ejecución de SeqTrimNext, además de los *plugins* utilizados en la misma.
3. **Pre-processing statistics**, en el que se muestran varias gráficas para valorar si la distribución de la longitud de las lecturas antes y después del preprocesamiento resulta aceptable (figuras 1 y 3, apéndice F). También se adjunta una gráfica con los valores de calidad de las lecturas en función de la longitud para valorar a partir de qué tamaño comienzan a ser menos fiables (figura 2, apéndice F). A continuación se presentan cuatro tablas con los vectores, adaptadores y contaminantes más encontrados, además de un resumen de los motivos por lo que se han recortado las secuencias. Al final de este apartado aparecen los mensajes de advertencia en caso de que haya algún motivo de rechazo demasiado frecuente.
4. **Rejected reads**, donde se resumen las lecturas rechazadas y los motivos del rechazo.

Utilidad de SeqTrimNext

A pesar de que SeqTrimNext está pendiente de publicación, ya muestra una gran aceptación por la comunidad científica, lo que se refleja en las más de 2400 veces que se ha descargado la gema de ruby necesaria para su instalación (`http://rubygems.org/gems/seqtrimnext`), y en su utilización en la PAB con 42 usuarios, y en otras instalaciones, como la universidad belga de Gante (`http://www.ugent.be/hpc/en/pubs/newsletter/newsletter3`), donde está instalado para su ejecución en paralelo en un *cluster*.

En los apartados 11.4.1 y 11.4.2 se mostrará con ejemplos prácticos la utilidad de SeqTrimNext, al encontrar que su resultado afecta profundamente al posterior ensamblaje. El preprocesamiento con SeqTrimNext también ha servido para detectar errores en el laboratorio a la hora de preparar las secuencias antes de la secuenciación, para evaluar la puesta a punto del FLX+ que se ha instalado en al Universidad de Málaga, y ha sido de gran utilidad para descartar artefactos y secuencias sin información biológica. Se ha comprobado que las secuencias preprocesadas acelerarán el ensamblaje y mejoran su fiabilidad (véase el artículo de SeqTrimNext incluido a continuación), lo que ha repercutido directamente en el transcriptoma obtenido para *Pinus pinaster* en este trabajo (véase los apartados 11.4.1 y

11.4.2). Tampoco hay que olvidar que SeqTrimNext se puede utilizar no solo para preprocesar lecturas de transcriptómica, sino que también está preparado para preprocesar lecturas para genómica, donde la mejora del ensamblaje es más notable (R. Bautista, comunicación personal). A continuación se presenta el manuscrito de SeqTrimNext que se preparó como *Application Note* para *Bioinformatics*, aunque se está modificando para su publicación en un número especial de la revista *Biology* dedicado a la NGS:

BIOINFORMATICS

Vol. 00 no. 00 2011
Pages 1–3

SeqTrimNext: pre-processing sequence reads for next-generation sequencing projects

Darío Guerrero-Fernández [1], Almudena Bocinos [1,2], Rocío Bautista [1], Noé Fernández-Pozo [3], Juan Falgueras [2] and M. Gonzalo Claros [1,3*]

[1]Plataforma Andaluza de Bioinformática (Edificio de Bioinnovación), [2]Departamento de Leguajes y Ciencias de la Computación, and [3]Departamento de Biología Molecular y Bioquímica, Universidad de Málaga, 29071 Málaga, Spain.

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXX

ABSTRACT

Summary: SeqTrimNext is a quick, reliable, distributed and/or parallelised, customisable, pipeline for pre-processing next-generation sequencing reads. It is also available as a friendly web tool and as a REST web service. Minimal user intervention is required for obtaining pre-processed reads, which are ready for any downstream analysis. A detailed summary statistics for quality-control of input and output data is given. SeqTrimNext usage increases the accuracy of consensus sequences and accelerates further assembly, saving curation time for sequencing projects.

Availability: SeqTrimNext is distributed under the GNU Affero Public License (AGPL) as a Ruby gem (`gem install seqtrimnext`) and is freely available to both academic and commercial users. Web site: `http://www.scbi.uma.es/seqtrimnext`.

Contact: dariogf@uma.es

Supplementary information: Supplementary figure is available at the journal's web site

INTRODUCTION

The advent of the technologies so-called next-generation sequencing (NGS) is giving non-specialist laboratories the ability to obtain large amounts of sequences. Despite new pre-processing needs, this has not been accompanied by a quality control of the sequence accurateness (Schmieder and Edwards, 2011). In addition to the typical polyA/T, adaptors, contamination or low quality portions that are a common source of sequencing errors, pre-processing should now take into account new NGS specific features such as (i) processing time and computing capability for a very large number of reads; (ii) presence of artificial read duplicates (e.g. clonality; Huang *et al.*, 2010); (iii) location and removal of sample identifiers (tags, barcodes and key; Schmieder *et al.*, 2010); (iv) grouping reads according to barcodes; (v) managing of paired-end reads; (vi) discarding low complexity artefacts introduced by the sequencing technology; and (vii) coping with standard formats (Cock *et al.*, 2010). Widely-used pre-processors such as SeqClean or Lucy2 do not face most of these particular challenges, since they were developed for Sanger sequences. Others such as SeqTrim (Falgueras *et al.*, 2010), PyroCleaner, or the trimming steps of ngs_backbone (Blanca *et al.*, 2011) and Newbler (Roche), face only some of them. Other recent pipelines were designed only for specific problems [e.g. CANGS (Pandey *et al.*, 2010), TagCleaner (Schmieder *et al.*, 2010), CLOTU (Kumar *et al.*, 2011), PRINSEQ (Schmieder and Edwards, 2011)]. Finally, some authors use in-house scripts for pre-processing (e.g. Diguistini *et al.*, 2009). As a result, even if a lot of time is invested on curation by hand, databases are contaminated with NGS linkers and adaptors (mainly HTGS database, see e.g. AC225459.16, AC122759.1, AC239084.1). Therefore, there is a need for new, fast, efficient, reliable, easy-to-install, user-friendly, pre-processing software for NGS reads for a wide range of computers and experimental conditions (e.g. *de novo* assembling, mapping, amplicons). SeqTrimNext has been developed to fill these necessities and to cope with all NGS peculiarities, providing output files that can be used as input for downstream analyses.

DESCRIPTION

SeqTrimNext is a parallelised evolution of SeqTrim (Falgueras *et al.*, 2010), completely re-written in Ruby 1.9.2, as a command line tool. The web interface and the REST-based web service have been built using InGeBiol —a Ruby-on-Rails 2.0 framework giving a web interface and web service commands for any command-line (Guerrero-Fernández and Claros, in preparation)—. The modular architecture of SeqTrimNext, based on a pipeline of orthogonal plugins, is especially suitable for addition, removal, or reordering of plugins and easy adaptation for future evolution. Internal configuration files are in JSON for efficiency in the storage and parsing phases. Similarity searches were carried out with BLAST+ customised calls. Other required programs are CD-HIT (Huang *et al.*, 2010), GNU_PLOT (`http://www.gnuplot.info`), and optionally latex and the gem *seqtrimnext_report* (on trial) for constructing a summary PDF. Distributed and parallel execution of SeqTrimNext based on one manager and a variable number of independent workers is provided by the Ruby gem *SCBI_MapReduce* (Guerrero-Fernández et al, in preparation). The number of workers can be optimally made equal to the number of cores in each machine, minimising any idle time. Additional acceleration has been achieved by (i) processing chunks of 100

*to whom correspondence should be addressed

10.2. Verificación de ensamblajes *de novo* del transcriptoma

Una vez que se tienen lecturas fiables, hay que ensamblarlas para reconstruir el mejor transcriptoma posible (apartado 3.6.2 de la introducción, pág. 25). Esto no es fácil de comprobar cuando se trabaja con una especie no modelo como el pino, dado que no se cuenta con ningún transcriptoma ni genoma de referencia lo suficientemente fiable. Por ese motivo, lo más recomendable es utilizar varios algoritmos de ensamblaje distintos y luego ser capaz de encontrar cuál de ellos produce los mejores resultados sin tener ninguna referencia. En el grupo de investigación ya se había diseñado anteriormente el programa Full-Lengther [130] para trabajar con secuencias procedentes de EST obtenidas por las técnicas de Sanger. Partiendo de su filosofía se desarrolló FULL-LENGTHERNEXT, un algoritmo que fuera, entre otras cosas, capaz de trabajar con lecturas de NGS. FULL-LENGTHERNEXT emplea como entrada las secuencias de los unigenes obtenidos por cualquier método de ensamblaje y va a permitir determinar, de manera resumida, lo siguiente:

- Cuántos y cuáles unigenes tienen un ortólogo en las bases de datos.
- Cuántos ortólogos diferentes se han encontrado.
- Cuántos y cuáles unigenes codifican una proteína completa; también proporciona la secuencia de la posible proteína.
- Cuántos y cuáles unigenes codifican un trozo de proteína, y cuál es ese trozo.
- En cuántos unigenes diferentes se ha reconstruido completo el marco abierto de lectura.
- Cuántos y cuáles unigenes pueden ser específicos de especie y no errores de ensamblaje o de secuenciación.
- Qué unigenes serían ARN no codificantes.
- Cuántos y cuáles unigenes pueden corresponder a secuencias artefactuales que se podrían eliminar del ensamblaje porque no serían realmente parte del transcriptoma.

Gracias a la información anterior se pueden comparar distintos ensamblajes para conocer cuál es el mejor en función, principalmente, del número de unigenes completos diferentes que se consiguen reconstruir. Esto también nos ha permitido establecer repetidamente con lecturas de 454 de distintas especies que los mejores resultados de ensamblaje se obtienen cuando las lecturas se ensamblan con MIRA3 [44] y con Euler-SR [175], y luego se unifican con CAP3 [105].

A pesar de que el artículo de FULL-LENGTHERNEXT está en evaluación, y que su gema de instalación se hizo pública solo desde el mes de marzo de 2012, ya ha sido descargada más de 550 veces (`http://rubygems.org/gems/full_lengther_next`). En el artículo se propone que el valor añadido de FULL-LENGTHERNEXT reside en su capacidad para validar ensamblajes *de novo* de transcriptomas y obtener una anotación preliminar de los mismos. El investigador podrá decidir entonces si el ensamblaje está listo para una anotación más intensa que puede durar varias semanas o meses, o si hay que invertir más esfuerzos (bioinformáticos o de laboratorio) en mejorar el transcriptoma en estudio.

A continuación se describe en detalle este algoritmo en un manuscrito en evaluación en *Bioinformatics*.

BIOINFORMATICS

Vol. 00 no. 00 2012
Pages 1–17

FULL-LENGTHERNEXT: A tool for fine-tuning *de novo* assembled transcriptomes of non-model organisms

Noé Fernández-Pozo [1], Darío Guerrero-Fernández [2], Rocío Bautista [2] and M. Gonzalo Claros [1,2*]

[1]Departamento de Biología Molecular y Bioquímica, Facultad de Ciencias, Universidad de Málaga, 29071, Spain.
[2]Plataforma Andaluza de Bioinformática, Centro de Supercomputación y Bioinformática, Edificio de Bioinnovación, Universidad de Málaga, 29590 Málaga, Spain.

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Motivation: *De novo* transcriptome assemblies devoid of any genomics or transcriptomics reference occur commonly when working with non-model species. There is no easy way to distinguish between artefacts and species-specific transcripts, or to identify the best possible assembly before deep annotation or complementary laboratory research.

Results: FULL-LENGTHERNEXT has been designed as a parallelisable and distributable command-line, web-tool and REST web-service pipeline adapted to high-throughput transcriptomics. It has applications in (1) the classification of unigenes among coding for complete or incomplete proteins, (2) the construction of an open reading frame fixing the likely frame-shifts in unigene assemblies; (3) the discovery of putative species-specific unigenes; and (4) the extraction of putative non-coding RNAs. Combining the previous information, it provides a quick overview for the selection of the best *de novo* transcriptome assembly. Particularly, it has been shown that independent *de novo* assemblies using MIRA3 and Euler-SR can be reconciled using CAP3 in a more successful transcriptome. Also, unigenes receiving an `Unknown` status usually are short sequences that can be discarded for subsequent processings.

Availability: FULL-LENGTHERNEXT can be freely downloaded or executed via web at http://www.scbi.uma.es/full_lengther2

Contact: claros@uma.com

Supplementary Information: Suppl. Files 1, 2, 3, 4, 5.

1 INTRODUCTION

Next-generation sequencing (NGS) platforms can sequence a particular transcriptome in a fast and cost-effective way (see for example Angeloni *et al.*, 2011; Malik *et al.*, 2011; Kristiansson *et al.*, 2009; Vera *et al.*, 2008). Assemblers for transcriptomics reads (e.g. Li *et al.*, 2009; Chevreux *et al.*, 2004; Pavel A. Pevzner and Waterman, 2001) will produce contigs (usually considered unigenes) that, in the ideal case, should be close to the number of real genes expressed under the studied conditions. When analysing *de novo* transcriptomes, it is quite difficult to distinguish between artefacts, true new transcripts, and the estimated number of identified transcripts. Annotations cannot be relied on for distinguishing between them since they are based on sequence similarity, and most of the problematic cases will not have any orthologue in databases. In fact, in transcriptomics projects, 40-60% of unigenes do not match any similar sequences in databases (Paterson *et al.*, 2010; Parchman *et al.*, 2010; Fernández-Pozo *et al.*, 2011). Most of the times, researchers hypothesise that they derive from lineage- or species-specific genes, but unfortunately a lot of them might simply be the result of misassembly or assembly of inaccurate reads (Paterson *et al.*, 2010). For that reason, efforts should be invested in distinguishing real new genes from misassembled sequences.

In *de novo* transcriptome assemblies, the identification of reconstructed transcripts coding for a complete protein has not received enough attention, even though they are extremely useful for (i) inferring the protein coded by the transcript, (ii) determining the genomic structure of genes in non-model organisms, (iii) facilitating gene identification efforts, and (iv) catalysing experimental research (Team, 2002). Identification of full-length cDNAs is a milestone in non-model species (Ralph *et al.*, 2008; 1KP Project `http://www.onekp.com`). No bioinformatics tool is available for identification of full-length transcripts in NGS projects. Tools such as TargetIdentifier (Min *et al.*, 2005) or Full-Lengther (Lara *et al.*, 2007), did exist for full-length transcript prediction in classical EST (expressed sequence tag) experiments. These softwares were prepared for low-throughput, cDNA-cloning based ESTs. In spite of their widespread use (Wang *et al.*, 2010; Koop *et al.*, 2008; Kim *et al.*, 2008; Semova *et al.*, 2006), both algorithms are ineffectual for working with unigenes from NGS for two reasons: (1) there is no clone supporting any sequence (which is the main rationale of these algorithms), and (2) they are not prepared for high-throughput, and cannot cope with new sequencing technologies.

Another aspect of *de novo* transcriptome assembly is the constant development and improvement of NGS assemblers. But, there is no easy way to determine which is the best possible assembly when working with a non-model species. It is proposed here that this task could be assisted by determining the amount of long reconstructed transcripts and the number of contigs coding for different, complete proteins.

*to whom correspondence should be addressed

10.3. Anotación

Al realizar la anotación de transcriptomas como el del pino (más detalles sobre la anotación en el apartado 3.6.3, pág. 27), al igual que ocurre con otras especies de plantas, gran parte de las anotaciones encontradas por similitud provienen de especies recientemente secuenciadas, que pueden contener anotaciones sin información [213]. Una solución a esta limitación puede ser combinar varias herramientas que integren diferentes tipos de anotaciones o la extracción de la información desde diferentes grupos de datos [213]. Por estos motivos, además de la información obtenida con Full-LengtherNext, hay que emplear otras herramientas de anotación complementarias, de libre acceso y, a ser posible, en mantenimiento, como Blast2GO [49] y AutoFact [126].

El principal problema de los programas de anotación es que consumen mucho tiempo de cálculo, incluso aunque se ejecuten en paralelo con varias CPU (apartado 7.3.2). Este tiempo se emplea no solo en obtener la mayor información posible para cada secuencia, sino en procesar «demasiadas» secuencias similares. Ya se ha comentado en el artículo de Full-LengtherNext que entre un 40-60 % de los unigenes de un transcriptoma ensamblado *de novo* pueden llegar a ser artefactos. Y también se han dado ejemplos de casos en los que, con mucha frecuencia, se acaban obteniendo transcriptomas con más de 80 000 e incluso más de 200 000 unigenes, cuando en ningún caso se espera que una especie contenga más de 40 000 genes. Este es uno de los motivos por lo que se ha decidido que antes de pasar a la anotación, primero hay que obtener el balance de Full-LengtherNext que va a indicar qué ensamblajes merece la pena anotar (en otras palabras, anotar sólo el mejor ensamblaje). Después se seleccionarán los unigenes que merece la pena anotar en profundidad (todos los que reciban un estado de Full-LengtherNext que no sea `Unknown`). Todo lo demás, si interesa, se podrá anotar «sin prisas», o directamente se puede descartar.

10.3.1. Asignación de definiciones para un unigén

Las definiciones consisten en una frase que proporciona información inmediata sobre el posible producto del gen. De nuestra experiencia con los investigadores de laboratorio se vio que era importante contar con al menos una definición para cada unigén, dado que es casi la única información que resultará interpretable a simple vista por un humano. Es más, se ha comprobado que, cuando se incluyen varias definiciones para un unigén y todas ellas dicen cosas parecidas, el usuario queda convencido de que la anotación es correcta. En cambio, cuando las definiciones son incongruentes, el usuario pondrá en duda con razón las anotaciones que vayan asociadas a dicho unigén. Una mejora de futuro para ello será proporcionar al usuario un valor de concordancia entre las definiciones para evitar que tenga que ir de una en una por las que le interesan.

Aunque con Full-LengtherNext ya se proporciona una definición, la mejor herramienta para esto es AutoFact [126] porque compara la secuencia de cada unigén con varias bases de datos (hasta 10) de una manera jerárquica, empezando con las más informativas para poco a poco acabar con las menos informativas, como las formadas por EST. Lamentablemente, el programa es extremadamente lento, incapaz de usar varias CPU, y aparentemente lleva unos años sin mantenimiento. Por eso, en colaboración con Darío Guerrero-Fernández, se modificó la versión 3.4 disponible en internet (véase apartado 7.3.2, pág. 52), incluyéndole:

- Utilización la última versión de Blast [34].
- Paralelización de la ejecución de los Blast.
- Filtración de la salida de Blast por un valor de $E < 10^{-3}$ (y no de $E < 10$).
- Devolución de solo las 12 mejores secuencias (y no las 500 primeras).
- Actualización las bases de datos.
- Aceleración de la ejecución al dividir las secuencias entrantes en grupos de 500 y ejecutarlos en el sistema de colas mediante *array-jobs* (véase cómo en el apartado 7.1.1, pág. 47).

En la última versión obtenida del transcriptoma de pino (89 544 unigenes, véase el apartado 11.4), AutoFact asignó anotación funcional a 39 921 unigenes (44,58 %), clasificó a 28 570 (31,91 %) como *Unassigned protein* y 21 053 (23,51 %) como *unclassified* (*Anotación funcional*, *Sin descripción útil* y *Desconocido* respectivamente en la tabla 10.1). Los unigenes con anotación funcional fueron anotados utilizando un ortólogo de pino en el 5,30 % de los casos, con coníferas en un 8,69 %, con otro representante de *Gimnospermas* en un 0,79 % (*Ginko* o *Cyca*, que son especies con poca información en las bases de datos), con otra planta en el 67,75 % y con otro organismos en el 17,47 %. Además, de los 39 921 unigenes anotados funcionalmente, a 93 se les asignó un código EC, los cuales pertenecían a 48 enzimas diferentes. Los unigenes clasificados como *Unassigned protein* no aportan definiciones del producto del gen que contienen, pero son indicativos

Tabla 10.1: *Resumen del tipo de definiciones aportadas por cada programa utilizado a los 89 544 unigenes de la última versión obtenida del transcriptoma de pino (apartado 11.4).*

Programa	Anotación funcional		Sin descripción útil		Desconocido		E	tiempo
	#unigenes	%	#unigenes	%	#unigenes	%		
AutoFact	39 921	44,58	28 570	31,91	21 053	23,51	10^{-3}	>1mes
FLN	10 707	11,96	36 379	40,63	42 458	47,41	10^{-25}	15 horas
Blast2GO	26 653	29,77	18 003	20,11	44 888	50,70	10^{-10}	>1mes

de que se parecen a EST y a otras secuencias de las bases de datos consultadas, por lo que seguramente contengan una secuencia o fracción de un gen desconocido. Los unigenes clasificados por AutoFact como *unclassified* no muestran ninguna similitud en las bases de datos, ni en proteínas ni EST, por lo que seguramente en su mayoría son secuencias sin información biológica, producto de fallos de manipulación en el laboratorio, el preprocesamiento o el ensamblaje.

Full-LengtherNext tardó 14 horas, 53 minutos y 26 segundos (véase tabla 10.1) en aportar definiciones (con un valor de $E < 10^{-25}$) a los 89 544 unigenes de SPDB v1.2, la última versión del transcriptoma de pino (tabla 10.1). Anotó 47 034 (52,53 %) unigenes con similitud con proteínas de las bases de datos de UniProt, de los cuales 34 507 (73,37 %) se parecen a proteínas de coníferas, aunque por desgracia, estas proteínas no están anotadas en su mayoría, y únicamente el 11,72 % de ellas (4044 proteínas) aportan una definición útil a nuestro transcriptoma. El resto de unigenes, 12 527 (26,63 %), fueron anotados con proteínas de plantas que no forman parte de las coníferas, de los cuales el 52,77 % aportaron una definición útil. Por desgracia, de Blast2GO (tabla 10.1) no se dispone de las especies de origen de las secuencias con las que se ha anotado, y su ejecución se demoró más de un mes.

El diferente nivel de adjudicación de definiciones mostrado en la tabla 10.1 se debe a la filosofía de cada programa, por lo que resultan complementarios. Así, el objetivo de Full-LengtherNext no es el de anotar con la mejor definición. Además, utiliza el valor de E más bajo y emplea una base de datos de referencia que solo contiene proteínas completas (de coníferas en este caso), aunque estas no aporten una anotación funcional útil. En el otro extremo se puede situar AutoFact, cuyo objetivo es aportar alguna definición útil de los productos de los genes, buscando la mejor entre varias bases de datos y entre varios ortólogos de cada base de datos. Por eso es el que utiliza el valor de E más alto y compara los unigenes con varias bases de datos de proteínas y nucleótidos, sin importar si las secuencias están completas, ni la especie de la que provienen o ni siquiera si son de EST. De ahí que sea el programa que aporta más definiciones a los unigenes del transcriptoma de pino y el que menos unigenes deja sin asignarle alguna información. En el caso intermedio se encuentra Blast2GO, en el que se aplicó un valor de E bastante bajo (aunque no tan bajo como Full-LengtherNext) para asegurarse de que las anotaciones que se obtienen provienen de ortólogos suficientemente fiables. También proporciona un gran número de unigenes con anotación útil porque en lugar de utilizar la definición del mejor ortólogo (como Full-LengtherNext), trata de extraer la mejor definición de entre todos los hits que devuelve Blast (como AutoFact), intentado descartar los ortólogos encontrados con definiciones sin utilidad, del tipo *Unknown protein* o *uncharacterized protein*. Full-LengtherNext y Blast2GO muestran números parecidos de unigenes desconocidos porque ambos utilizan bases de datos de proteínas, y AutoFact tiene menos genes desconocidos porque además de las proteínas también busca secuencias por similitud en bases de datos de nucleótidos, incluidas las EST.

En cuanto al tiempo de ejecución de cada programa (véase la tabla 10.1), se conoce el tiempo exacto empleado por Full-LengtherNext para realizar el análisis (con 8 CPU), porque se ejecutó a través del sistema de colas añadiendo el comando `time` a su ejecución. Sin embargo, con respecto a AutoFact y Blast2GO, no es posible conocer el tiempo exacto que necesitaron para anotar las secuencias. En el caso de AutoFact, al utilizarse un *array-job* con más de 100 paquetes de secuencias, es muy difícil saber el tiempo exacto que tardó, puesto que dependiendo de la carga del sistema de colas se ejecutaban simultáneamente más o menos paquetes, y a veces, durante periodos de tiempo desconocido, algunos paquetes se mantenían pausados. Con Blast2GO, al utilizarse a través de una interfaz gráfica, no había ningún modo de conocer cuanto tardó.

En conclusión, y basándose en los resultados de la tabla 10.1, para la anotación de transcriptomas de especies no modelos es recomendable combinar varias herramientas que se complementen, obteniendo diferentes anotaciones procedentes de diferentes fuentes. Por ejemplo, es imprescindible una herramienta rápida como Full-LengtherNext que permita verificar la calidad del ensamblaje y estimar

el número de unigenes con ortólogo fiables, utilizando un valor de E bastante restrictivo. Por otro lado, es necesario la aplicación de otras herramientas como Blast2GO, que complementen la anotación con términos GO, EC, InterPro y KEGG, y otras como AutoFact que busquen descripciones para el mayor número de unigenes posibles, incluyendo la búsqueda por similitud en bases de datos de nucleótidos. Sin embargo, con la carga de trabajo que se genera con NGS, es necesario resolver el problema del tiempo empleado para la anotación, y programas como AutoFact y Blast2GO tendrán que evolucionar para ser más rápidos o serán sustituidos por otros nuevos que sean capaces de cumplir la misma función en menos tiempo.

10.3.2. Otras anotaciones

Además de las descripciones de los productos génicos se obtuvieron otras anotaciones como los términos de la *Gene Ontology* (GO), los códigos de la *Enzimme Commission* (EC), las rutas metabólicas de KEGG, los códigos InterPro, los polimorfismos mononucleotídicos (SNP) y las repeticiones de secuencias simples (SSR). De ellas, los términos GO, EC e InterPro se obtuvieron con la herramienta Blast2GO [49] (apartado 7.3.4, pág. 53). Como Blast2GO mostró problemas para trabajar con más de 10 000 secuencias en cualquiera de sus versiones y de nuestros equipos, los unigenes del transcriptoma de entrada se dividieron en paquetes de 10 000 secuencias mediante el *script* `split_fasta.rb` (disponible en el apéndice A, pág. 197).

Las anotaciones de la primera versión del transcriptoma, EuroPineDB, se pueden encontrar en su artículo, incluido en el siguiente capítulo (apartado 11.1). En la última versión del transcriptoma de pino recogida en este trabajo (SPDB v1.2, con 89 544 unigenes, véase el apartado 11.4), 15 140 unigenes se anotaron con 3697 términos GO diferentes, 2144 unigenes se clasificaron como enzimas, obteniendo 532 EC diferentes y 22 680 unigenes con 5190 códigos InterPro diferentes. A pesar de que Blast2GO también obtiene las rutas metabólicas KEGG, éstas no se toman de aquí porque a veces, aparecen rutas que ni siquiera existen en las plantas en los unigenes de pino. Esto se debe a que al basarse en los códigos EC, a cada enzima encontrada se le asignan las rutas KEGG en las que participan, y puede darse el caso de que, como hay enzimas que participan en varias rutas de plantas y de animales, se les pueda asignar incorrectamente una ruta que no existe en plantas. Por este motivo, es necesario filtrar los KEGG para descartar los que no tienen sentido en el organismo de estudio (como se verá en el apartado 11.3.5, pág. 160).

La detección de los polimorfismos mononucleotídicos (SNP) de EuroPineDB se realizó con AlignMiner [91]. Pero en en SPDB v1.2 se empleó Gigabayes (este análisis fue llevado a cabo por H. Benzekri) porque es una herramienta ampliamente utilizada y específicamente diseñada para este propósito, mientras que la detección de SNP con AlignMiner era más bien una estimación de posibles SNP, sin profundizar mucho en su fiabilidad. Así, en SPDB v1.2 se clasificaron 695 404 SNP repartidos en 44 051 unigenes, lo que hace una media de 15,79 SNP por unigén (teniendo en cuenta únicamente los que mostraron SNP) (H. Benzekri, comunicación personal).

La búsqueda de repeticiones de secuencias simples (SSR) se realizó con MREPS tanto en EuroPineDB como en SPDB v1.2. En SPDB V1.2 se caracterizaron 3724 SSR, 282 con repeticiones de 2 pb, 2068 con repeticiones de 3 pb, y 1372 con repeticiones de más de 3 pb (H. Benzekri, comunicación personal).

Los SNP y SSR encontrados en SPDB y EuroPineDB no son comparables, puesto que el ensamblaje realizado con MIRA3 en EuroPineDB es más redundante que el realizado en SPDB con el flujo de trabajo descrito en este capítulo. Además, en el caso de los SNP, SPDB cuenta con muchas más lecturas de partida, por lo que su secuencias son más heterogéneas, ya que provienen de un mayor número de condiciones, tejidos y poblaciones. Esto repercute en su ensamblaje y en los SNP obtenidos, y además hay que tener en cuenta, que los algoritmos utilizados para la detección de SNP en ambos casos, son también diferentes.

10.3.3. Anotación de secuencias genómicas

A la vez que se estaba caracterizando el transcriptoma de pino, en el grupo de investigación se comenzaron a obtener los primeros resultados de clonación y secuenciación de clones BAC de *Pinus pinaster*, que contenían diversos genes de interés. La secuenciación estos BAC se realizó con tecnología NGS (454/FLX con y sin secuencias pareadas) y las lecturas se preprocesaron con SeqTrimNext. Para localizar con precisión la posición del gen que contenían, una vez ensambladas las lecturas, se diseñó un programa (GENote v 1.β) que se aprovechaba de la información disponible sobre el transcriptoma que se estába elaborando. Esta herramienta también ha servido para inspeccionar visualmente si el ensamblaje del BAC contenía artefactos (figura 1 del artículo adjunto). Las principales ventajas de GENote v 1.β son:

- su capacidad para utilizar de secuencia de en-

trada varios contigs o *scaffolds*.

- su capacidad para ordenar los distintos contigs o *scaffolds* obtenidos al ensamblar el BAC.
- que muestra gráficamente los resultados de forma intuitiva.

La aplicación de GENote v 1.β a los ensamblajes de los BAC de *Pinus pinaster* sirvió para evaluar visualmente su calidad, para detectar y corregir los errores de ensamblaje, para distinguir los genes verdaderos de los pseudogenes, y para mostrar un gran número de elementos transponibles y secuencias codificantes aisladas, sin otros genes en miles de bases alrededor (R. Bautista, comunicación personal).

A continuación se presenta el artículo de Genote v 1.β incluido en el libro de actas *Bioinformatics for Personalized Medicine* publicado por Springer-Verlag.

Ana T. Freitas Arcadi Navarro (Eds.)

Bioinformatics for Personalized Medicine

10th Spanish Symposium, JBI 2010
Torremolinos, Spain, October 27-29, 2010
Revised Selected Papers

Springer

Volume Editors

Ana T. Freitas
INESC-ID/Instituto Superior Técnico
R. Alves Redol 9
1000-029 Lisboa, Portugal
E-mail: atf@kdbio.inesc-id.pt

Arcadi Navarro
Institut de Biologia Evolutiva (UPF-CSIC)
Doctor Aiguader 88
08003 Barcelona, Spain
E-mail: arcadi.navarro@upf.edu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-28061-0 e-ISBN 978-3-642-28062-7
DOI 10.1007/978-3-642-28062-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011945508

CR Subject Classification (1998): H.3, H.2.8, F.2.1, H.4, C.2, H.5, D.2

LNCS Sublibrary: SL 8 – Bioinformatics

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

GENote v.β: A Web Tool Prototype for Annotation of Unfinished Sequences in Non-model Eukaryotes

Noé Fernández-Pozo[1], Darío Guerrero-Fernández[2], Rocío Bautista[2], J. Gómez-Maldonado[1], C. Avila[1], Francisco M. Cánovas[1], and M. Gonzalo Claros[1,2]

[1] Departamento de Biología Molecular y Bioquímica, Universidad de Málaga, Campus de Teatinos, 29071 Málaga, Spain
{noefp,pgomez,cavila,canovas,claros}@uma.es
http://www.bmbq.uma.es/fmp

[2] Plataforma Andaluza de Bioinformática, Universidad de Málaga, Severo Ochoa 34, 29590 Málaga, Spain
{dariogf,rociobm,claros}@scbi.uma.es
http://www.scbi.uma.es/pab

Abstract. *De novo* identification of genes in newly-sequenced eukaryotic genomes is based on sensors, which are not available in non-model organisms. Many annotation tools have been developed and most of them require sequence training, computer skills and accessibility to sufficient computational power. The main need of non-model organisms is finding genes, transposable elements, repetitions, etc., in reliable assemblies. GENote v.β is intended to cope with these aspects as a web tool for researchers without bioinformatics skills. It facilitates the annotation of new, unfinished sequences with descriptions, GO terms, EC numbers and KEEG pathways. It currently localises genes and transposons, which enable the sorting of contigs or scaffolds from a BAC clone, and reveals some putative assembly inconsistencies. Results are provided in GFF3 format and in tab-delimited text readable in viewers; a summary of findings is provided also as a PNG file.

Keywords: Annotation, web tool, unfinished sequence, gene finding, non-model species.

1 Introduction

Annotation is the process of interpreting raw genomic sequences into useful biological information by integrating computational analyses, auxiliary biological data and biological expertise. It should begin as early in a project as is possible, because the analysis of the assembled sequence will often identify problems in the raw sequence or in its assembly [2]. Genome annotation is best carried out by combining several methods, being very successful *cis*- and *trans*-alignments, and also the *de novo* gene prediction when a related species is well known (*de novo* gene predictors have repeatedly proven to be more challenging than expected) [2].

A.T. Freitas and A. Navarro (Eds.): JBI 2010, LNBI 6620, pp. 66–71, 2012.

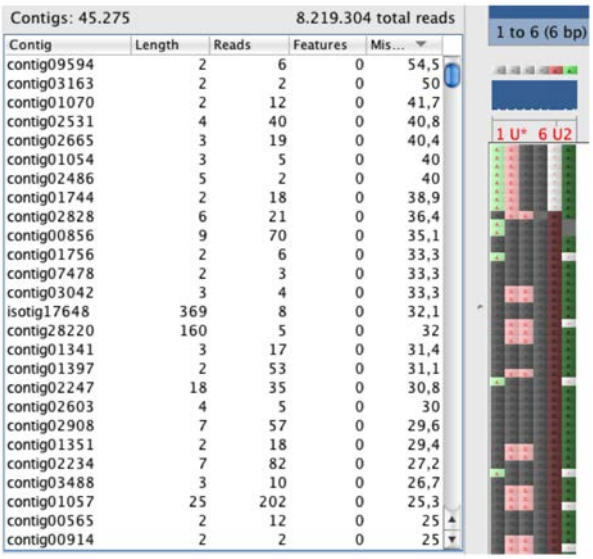
10.4. Flujo de trabajo resultante

El desarrollo de las técnicas de NGS ha hecho más patente la necesidad de disponer de un modo de analizar las secuencias de una forma lo más automática posible. Toda la experiencia técnica acumulada durante los análisis realizados nos permite proponer un flujo de trabajo para obtener el transcriptoma de *P. pinaster* y, por extensión, el de cualquier especie eucariota que no disponga de un genoma de referencia adecuado. El flujo que se propone (figura 10.1) está dividido en 4 pasos: preprocesamiento, ensamblaje, verificación y anotación.

Preprocesamiento: En la explicación de SeqTrim y de SeqTrimNext y la posterior validación con FULL-LENGTHERNEXT ha quedado patente que el preprocesamiento es clave para obtener buenos resultados. Cuando en un proyecto de secuenciación se utilizan secuencias de tipo Sanger, o bien EST clásicas de las bases de datos, siempre es recomendable preprocesarlas con SeqTrim, o con la nueva versión de SeqTrimNext, ya que dispone de los *plugins* necesarios para ello. Por tanto, las secuencias obtenidas del secuenciador o de una base de datos pública deberán pasar el filtro de preprocesamiento que garantice que solo se utilizarán las secuencias verdaderamente útiles. Por supuesto, nuestra recomendación es que el preprocesamiento se realice con SeqTrimNext.

Ensamblaje: Para el ensamblaje del transcriptoma con datos de NGS, en nuestro grupo de investigación se probaron varios de ellos: Newbler [147] y MIRA3 [44] basados en solapamiento (de tipo OLC, del inglés *overlap-layout-consensus*), y Velvet [237] y Euler-SR [175] de tipo euleriano.

De los ensambladores basados en solapamiento se descartó utilizar la versión 2.3 de Newbler porque en la bibliografía se recoge que comete un gran número errores [128, 161], llegando a recomendar la repetición de los ensamblajes de transcriptomas que se hayan realizado con esa versión o anteriores [128]. De hecho, cuando se utilizó la última versión disponible de Newbler, la 2.6, con secuencias altamente heterocigóticas como son las procedentes del transcriptoma de pino, se observaron contigs con hasta un 54,5 % de discrepancias (figura 10.2, primera línea). En otros casos había contigs de longitud extremadamente pequeña: de 45 275 unigenes generados por Newbler 2.6, 749 de ellos eran menores de 10 pb, a pesar de que las secuencias de entrada de menor tamaño eran de 40 pb porque es el tamaño más pequeño que SeqTrimNext deja pasar con los parámetros por omisión. En cambio, al realizar el ensamblaje del mismo grupo de secuencias con MIRA3 se observó como mucho un 10,7 % de discrepancias, y contigs con una longitud mínima de 40 pb (cuando con Newbler podía llegar incluso a obtener unigenes de 1, 2 y 3 nucleótidos de longitud), lo que sí era congruente con el tamaño mínimo de las lecturas utilizadas. Además, en un estudio publicado este año [161] en el que se comparan CAP3, MIRA3, Oases y Newbler se demuestra que Newbler produce un alto número de quimeras, que Oases obtiene pobres resultados para secuencias de 454 y que MIRA3 y CAP3 son muy conservativos, por lo que reconstruyen secuencias muy fiables, más segmentadas y con un número de quimeras bajo, aunque producen transcritos redundantes [161]. Con estos datos se decidió descartar el uso de Newbler y utilizar MIRA3, ya que en nuestra opinión es preferible que dos secuencias de un mismo gen queden separadas en dos unigenes, a que dos secuencias de genes diferentes se unan en un mismo unigén y formen una quimera.



Contig	Length	Reads	Features	Mis...
contig09594	2	6	0	54,5
contig03163	2	2	0	50
contig01070	2	12	0	41,7
contig02531	4	40	0	40,8
contig02665	3	19	0	40,4
contig01054	3	5	0	40
contig02486	5	2	0	40
contig01744	2	18	0	38,9
contig02828	6	21	0	36,4
contig00856	9	70	0	35,1
contig01756	2	6	0	33,3
contig07478	2	3	0	33,3
contig03042	3	4	0	33,3
isotig17648	369	8	0	32,1
contig28220	160	5	0	32
contig01341	3	17	0	31,4
contig01397	2	53	0	31,1
contig02247	18	35	0	30,8
contig02603	4	5	0	30
contig02908	7	57	0	29,6
contig01351	2	18	0	29,4
contig02234	7	82	0	27,2
contig03488	3	10	0	26,7
contig01057	25	202	0	25,3
contig00565	2	12	0	25
contig00914	2	2	0	25

Figura 10.2: *Contigs del ensamblaje realizado con Newbler 2.6 ordenados de mayor a menor número de discrepancias, visualizados con Tablet.*

Con respecto a los ensambladores elegidos que utilizan algoritmos de tipo euleriano o basados en los grafos de De Bruijn, al comparar Euler-SR y Velvet (en ese momento aún no existía Oases, la versión adaptada a transcriptómica de Velvet) utilizando un k-mero de 25 se observó que Velvet obtuvo 87 508 unigenes, de los cuales el más largo era de 2239 pb y tan solo 8025 (9,17 %) de estos unigenes eran mayores de 500 pb, obteniéndose un número muy elevado de contig muy cortos (R. Bautista, comunicación personal). Teniendo en cuenta que en esta misma prueba se obtuvieron 51 630 unigenes

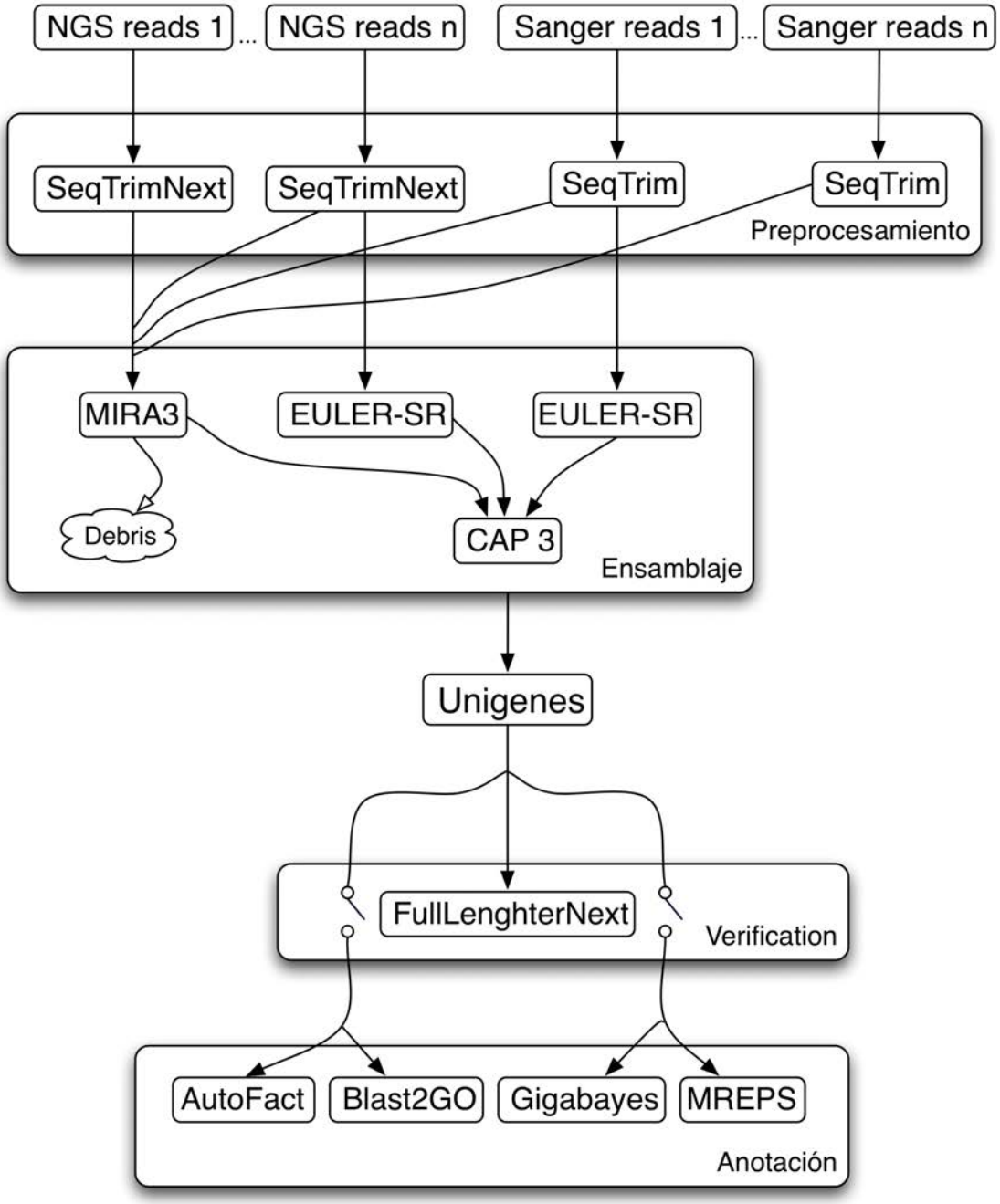


Figura 10.1: *Flujo de trabajo propuesto para obtener un transcriptoma de pino anotado a partir de lecturas de NGS y de tipo Sanger. En el caso de que solo haya lecturas de NGS se puede suprimir la parte correspondiente a Sanger. Las secuencias de tipo Sanger se preprocesan con SeqTrim y las de NGS con SeqTrimNext. Todas las lecturas limpias se ensamblan juntas con MIRA3, y las secuencias limpias de NGS se ensamblan además con Euler-SR, pero sin mezclar los diferentes experimentos. Luego se re-ensamblan los contigs de Euler-SR y MIRA3 utilizando CAP3, posteriormente se verifican los unigenes del ensamblaje con* Full-LengtherNext *y si se considera que el ensamblaje es correcto, se anotan los unigenes obtenidos con AutoFact, Blast2GO, Gigabayes y MREPS.*

con MIRA3, llama aún más la atención que Velvet obtenga casi 36 000 unigenes más. Al comprobar que todos estos unigenes eran muy pequeños se descartó el uso de Velvet en favor de Euler-SR, que es capaz de reconstruir un menor número de contigs con una longitud mayor, debido a que incorpora un sistema de corrección de errores [40], pudiendo alargar los caminos eulerianos, con lo que se disminuye

el número final de contigs generados. Con Euler-SR se obtuvieron 24 254 unigenes (63 254 menos que Velvet), de los cuales 11 295 (3270 más que Velvet) eran mayores de 500pb y cuyo unigén de mayor longitud tenía 4479pb (2240 pb más largo que el unigén más largo obtenido con Velvet).

Como no era posible decidir a priori qué ensamblaje era el mejor, y dado la combinación de diferentes algoritmos de ensamblaje aumenta la posibilidad de descubrir nuevos genes [108], en la estrategia de ensamblaje que se propone (figura 10.1) se combinan dos programas que utilizan estrategias diferentes, MIRA3 [44] y Euler-SR [175], de tipo OLC y euleriano respectivamente, y seguidamente, se unifican los resultados con CAP3 para tratar de corregir la redundancia de MIRA3, de forma equivalente a la recomendada en trabajos anteriores [238, 128, 108, 213, 148].

Es muy importante al ejecutar MIRA3 aplicar el parámetro `-CL:ascdc` (apartado 7.3.10, pág. 56) para evitar la formación de quimeras como la que se muestra en la figura 10.3, donde no se aplicó este parámetro. En esta figura se ve claramente una quimera en la que una sola lectura une dos bloques de secuencias, procedentes realmente de dos genes diferentes, como se muestra en el análisis realizado con BLASTx frente a Uniprot (panel inferior de la figura 10.3). De igual forma, también es importante que en MIRA3 se ensamblen las lecturas preprocesadas procedentes de todos los experimentos a la vez para que el algoritmo disponga de todas las lecturas para buscar las mejores regiones solapantes. En caso de realizar el ensamblaje de cada experimento por separado se obtendría una redundancia aún mayor (R. Bautista, comunicación personal). De esta forma, MIRA3 evitará que se unan en un mismo contig genes parálogos y ayustes alternativos de un mismo gen, aunque mostrando un número de unigenes elevado en los que algunos genes estarán representados por más de un contig.

Euler-SR es muy sensible a los errores de secuenciación y sólo completa las secuencias consenso cuando los caminos eulerianos no muestran incoherencias. Por eso es preferible realizar los ensamblajes con Euler-SR por separado, porque al mezclar varios experimentos procedentes de diferentes individuos, tejidos y condiciones se aumenta el número de isoformas/alelos/parálogos y los grafos de Bruijn se hacen mucho más complejos. Tal diversidad de secuencias hace aumentar los caminos alternativos y hace que el ensamblador produzca secuencias más cortas (R. Bautista, comunicación personal). Por lo general, los contigs obtenidos con este ensamblaje son muy fiables, pero cuantitativamente reconstruirá un numero menor de genes que los algoritmos de tipo OLC.

Al combinar dos algoritmos que utilizan estrategias de ensamblaje distintas, al igual que hizo para la detección de genes expresados diferencialmente en el apartado 9.2.6 del capítulo dedicado a las micromatrices de expresión, se espera que ambas herramientas se complementen y den un resultado mejor juntas que cada una por separado. Para reensamblar los unigenes obtenidos con los ensambladores anteriores se eligió CAP3 por ser conocido anteriormente como uno de los mejores ensambladores de secuencias más largas, como las de tipo Sanger, que son las más parecidas a las de los unigenes que se producen con Euler-SR y con MIRA3 [140]. De este modo, CAP3 puede complementar a MIRA3 porque corrige los errores de ensamblaje en los que secuencias del mismo transcrito no se ensamblan juntas [238], y disminuye así la redundancia. La unión de ensamblajes realizada con un programa diferente, CAP3 en nuestro caso, es una estrategia frecuentemente utilizada en proyectos genómicos, especialmente en los que utilizan varias tecnologías de secuenciación. La aplicación de este método a los ensamblajes transcriptómicos *de novo* resulta particularmente útil, generando un ensamblaje con unigenes más largos y más fiables [128].

Verificación: Como la anotación de los unigenes es el paso que más tiempo consume del flujo, con diferencia, después de realizar el ensamblaje, es necesario llevar a cabo un paso de verificación (figura 10.1). Este paso se realiza con FULL-LENGTHERNEXT, y sirve para evaluar de un modo rápido si el ensamblaje tiene una buena proporción de genes completos, si está muy fragmentado o si posiblemente contiene un alto número de secuencias sin información biológica. En caso de encontrar ensamblajes muy fragmentados o con una gran proporción de unigenes carentes de información, es recomendable evaluar con FULL-LENGTHERNEXT los unigenes procedentes de los ensamblajes de MIRA3 y Euler-SR, y revisar el preprocesamiento para buscar razones que expliquen la fragmentación y tratar de mejorar los resultados obtenidos. Por el contrario, si los resultados obtenidos fueran aceptables como para dar el ensamblaje por bueno, se puede pasar a realizar una anotación en profundidad de los unigenes. Si el número de unigenes con un estado de `Unknown` fuera demasiado alto (en general superior a 20 000 unigenes) se recomienda descartarlos para el análisis inicial, y centrar el estudio solamente sobre los unigenes que presenten cualquiera de los otros estados asignados por FULL-LENGTHERNEXT.

Anotación: La anotación propuesta en el flujo (figura 10.1) se completa utilizando los programas AutoFact, Blast2GO, Gigabayes y MREPS, tal y

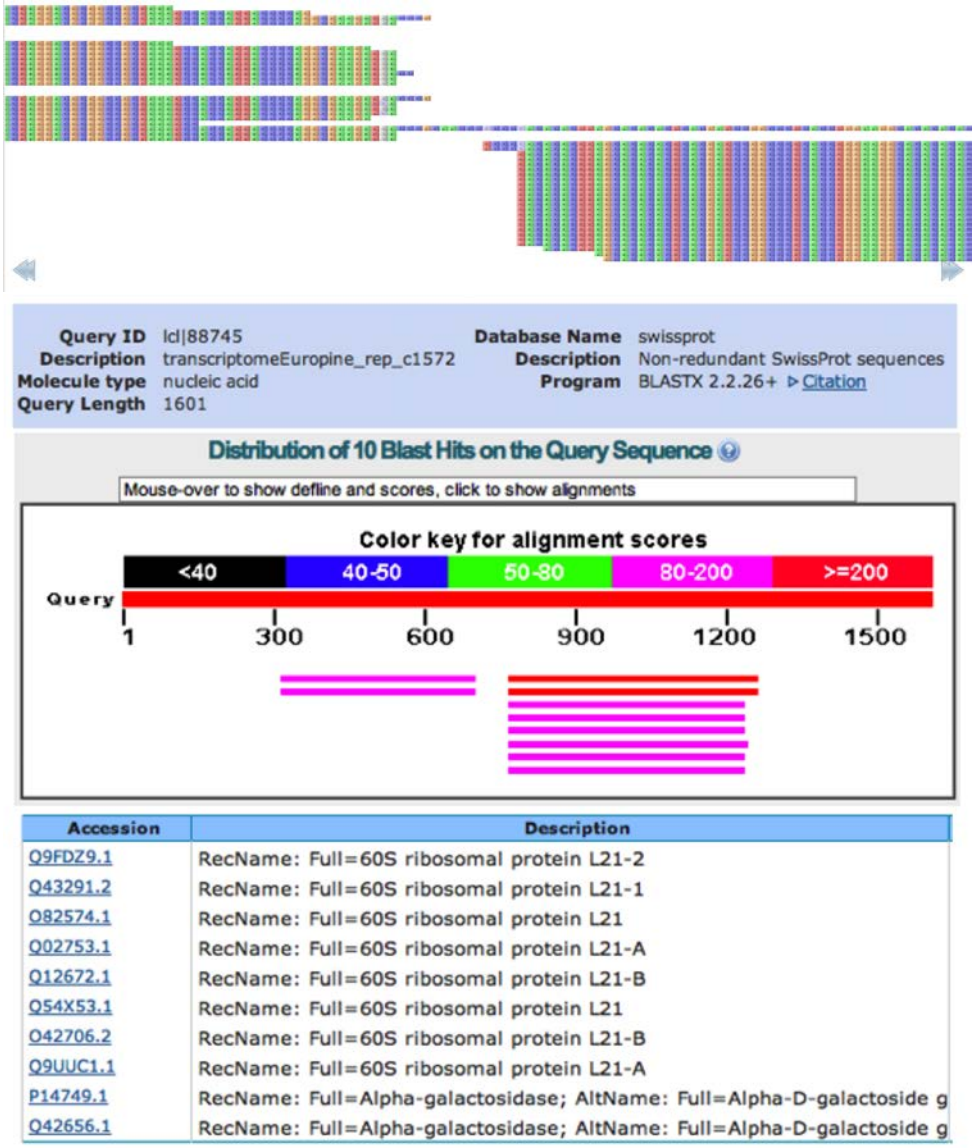


Figura 10.3: *Ejemplo de secuencia quimérica obtenida por MIRA3 cuando no se usa el parámetro* `-CL:ascdc`. *En el panel superior se observa que una sola lectura une dos bloques de secuencias procedentes de dos genes diferentes. En el panel inferior se comprueba que los dos bloques realmente corresponden a genes diferentes, uno de una proteína ribosómica, y el otro de una α-galactosidasa.*

como se comenta en el apartado 10.3 y en los Materiales y métodos (apartado 7.3). De este modo, los unigenes del transcriptoma quedan anotados con tres descripciones del producto del gen, que permiten a los investigadores confirmar si tres programas con bases de datos diferentes coinciden en la información encontrada, y también se dispone de términos GO, códigos EC, InterPro, rutas KEGG, SNP y SSR que permitirán realizar estudios computerizados de los resultados con programas de enriquecimiento biológico.

Como perspectiva de futuro, debido a lo que se demora la anotación con Blast2GO y AutoFact, a que las últimas versiones de Blast2GO han pasado a ser de pago, y a la ausencia de mantenimiento de AutoFact, se están empezando a probar otras herramientas de anotación. El director de esta tesis ha colaborado en el desarrollo de Sma3s [153], una herramienta de búsqueda de descripciones, GO y códigos EC e InterPro, que es mucho más rápida que Blast2GO y AutoFact (de hecho, es solo un poco más lenta que Full-LengtherNext; R. Bautista, comunicación personal). Además, las pruebas realizadas por los autores muestran que las anotaciones obtenidas, son tanto o más fiables en sensibilidad y especificidad (M. G. Claros, comunicación personal). Así que las futuras versiones del transcriptoma de pino probablemente se anoten con Sma3s.

Capítulo 11

Transcriptoma de referencia para *Pinus pinaster*

Con la aparición de las NGS, el análisis de datos se ha convertido en un cuello de botella, y el almacenamiento de datos está llegando a convertirse en un problema [20]. Por eso resulta esencial almacenar y compartir la información obtenida con la comunidad científica [213] de una forma ordenada, a ser posible con una interfaz gráfica y herramientas intuitivas que permitan el acceso a los usuarios independientemente de sus conocimientos informáticos.

En un principio, para almacenar la información de cada clon de nuestro grupo de investigación y de los clones impresos en la micromatriz Pinarray1 (apartado 8.1) se construyeron ficheros de texto tabulado, como el fichero GAL que contiene las anotaciones de los clones (apartado 9.1, pág. 65). Los ficheros de texto tabulado pueden consultarse en cualquier editor de texto o como hojas de cálculo en programas como Excel (Microsoft Office) o Numbers (incluido en el paquete Apple iWork). Sin embargo, la información mostrada en los editores de texto resulta muy incómoda de consultar, pudiendo saturar la capacidad de algunos editores cuando se manejan ficheros muy grandes. Aunque visualizar la información en Excel puede resultar aparentemente mucho más cómodo, esto puede provocar errores debido a que este programa en concreto siempre realiza una interpretación de los datos de forma automática, por lo que decide cambiar el formato de éstos, sobre todo en relación con la interpretación de espacios, comas y puntos entre números. De hecho, se ha descrito que no es una buena idea pasar los datos por este programa [236].

En el marco de este trabajo se ha desarrollado el primer transcriptoma de *Pinus pinaster* que relaciona información del Pinarray1, de secuencias del transcriptoma de pino de genotecas de ADNc y de una librería de 454, junto con sus anotaciones y almacenamiento de los clones. Posteriormente se ha completado dicho transcriptoma con otras librerías de 454 procedentes de varios grupos de investigación europeos. En el siguiente apartado se presenta el primer transcriptoma de pino marítimo, disponible en la base de datos EuroPineDB, y más adelante se describirán las mejoras técnicas aplicadas en esta base de datos para almacenar la nueva versión obtenida del transcriptoma de pino.

11.1. EuroPineDB: el primer transcriptoma de referencia de *P. pinaster*

A partir de los datos con los que se generó el Pinarray1 (capítulo 9, pág. 65) y de los datos de secuenciación del transcriptoma de pino, obtenidos en diferentes condiciones, por nuestro grupo de investigación y por otros laboratorios con los que se colaboró, se elaboró EuroPineDB [73], la primera base de datos del transcriptoma de pino marítimo. Se puede navegar por la información que contiene a través de las genotecas y sus placas de 96 pocillos que guardan los clones que se utilizaron para la secuenciación. También se pueden realizar búsquedas por palabras clave o por similitud de secuencia con Blast. Además, contiene las relaciones entre la secuencia, las anotaciones y los clones impresos en Pinarray1 (apartado 8.1).

A continuación se presenta el artículo publicado en *BMC Genomics* en el que se detallan las características y el contenido de esta base de datos.

Fernández-Pozo et al. BMC Genomics 2011, 12:366
http://www.biomedcentral.com/1471-2164/12/366

BMC Genomics

DATABASE Open Access

EuroPineDB: a high-coverage web database for maritime pine transcriptome

Noé Fernández-Pozo[1], Javier Canales[1], Darío Guerrero-Fernández[2], David P Villalobos[1], Sara M Díaz-Moreno[1], Rocío Bautista[2], Arantxa Flores-Monterroso[1], M Ángeles Guevara[3], Pedro Perdiguero[4], Carmen Collada[3,4], M Teresa Cervera[3,4], Álvaro Soto[3,4], Ricardo Ordás[5], Francisco R Cantón[1], Concepción Avila[1], Francisco M Cánovas[1] and M Gonzalo Claros[1,2*]

Abstract

Background: *Pinus pinaster* is an economically and ecologically important species that is becoming a woody gymnosperm model. Its enormous genome size makes whole-genome sequencing approaches are hard to apply. Therefore, the expressed portion of the genome has to be characterised and the results and annotations have to be stored in dedicated databases.

Description: EuroPineDB is the largest sequence collection available for a single pine species, *Pinus pinaster* (maritime pine), since it comprises 951 641 raw sequence reads obtained from non-normalised cDNA libraries and high-throughput sequencing from adult (xylem, phloem, roots, stem, needles, cones, strobili) and embryonic (germinated embryos, buds, callus) maritime pine tissues. Using open-source tools, sequences were optimally pre-processed, assembled, and extensively annotated (GO, EC and KEGG terms, descriptions, SNPs, SSRs, ORFs and InterPro codes). As a result, a 10.5× *P. pinaster* genome was covered and assembled in 55 322 UniGenes. A total of 32 919 (59.5%) of *P. pinaster* UniGenes were annotated with at least one description, revealing at least 18 466 different genes. The complete database, which is designed to be scalable, maintainable, and expandable, is freely available at: http://www.scbi.uma.es/pindb/. It can be retrieved by gene libraries, pine species, annotations, UniGenes and microarrays (i.e., the sequences are distributed in two-colour microarrays; this is the only conifer database that provides this information) and will be periodically updated. Small assemblies can be viewed using a dedicated visualisation tool that connects them with SNPs. Any sequence or annotation set shown on-screen can be downloaded. Retrieval mechanisms for sequences and gene annotations are provided.

Conclusions: The EuroPineDB with its integrated information can be used to reveal new knowledge, offers an easy-to-use collection of information to directly support experimental work (including microarray hybridisation), and provides deeper knowledge on the maritime pine transcriptome.

1 Background

Conifers (*Coniferales*), the most important group of gymnosperms, represent 650 species, some of which are the largest, tallest, and oldest non-clonal terrestrial organisms on Earth. They are of immense ecological importance, dominating many terrestrial landscapes and representing the largest terrestrial carbon sink. Currently present in a large number of ecosystems, they have evolved very efficient physiological adaptation systems. Given that trees are the great majority of conifers, they provide a different perspective on plant genome biology and evolution taking into account that conifers are separated from angiosperms by more than 300 million years of independent evolution. Studies on the conifer genome are revealing unique information which cannot be inferred from currently sequenced angiosperm genomes (such as poplar, *Eucaliptus*, *Arabidopsis* or rice): around 30% of conifer genes have little or no sequence similarity to plant genes of known function [1,2]. Unfortunately, conifer genomics is hindered by the very large genome (e.g. the pine genome is approximately 160 times larger than *Arabidopsis* and seven times larger

* Correspondence: claros@uma.es
[1]Departamento de Biología Molecular y Bioquímica, Facultad de Ciencias, Campus de Teatinos s/n, Universidad de Málaga, 29071 Málaga, Spain
Full list of author information is available at the end of the article

11.2. Primeras aplicaciones de EuroPineDB

En este apartado se comentarán algunos avances producidos gracias a contar con el transcriptoma de referencia de *Pinus pinaster* incluido en EuroPineDB. Por un lado se verá, que los unigenes del ensamblaje *P.pinaster* de EuroPineDB se utilizaron para encontrar los genes que codifican las enzimas de dos rutas metabólicas, y por otro que la información de los clones de ADNc de las genotecas se utilizaron para incrementar el Pinarray1.

11.2.1. Conexión del metabolismo C1, la biosíntesis de monolignoles y la asimilación de amonio

Para realizar una prueba práctica con los datos del transcriptoma de *Pinus pinaster* contenido en EuroPineDB se buscaron los genes de las enzimas que intervienen en la ruta que conecta el metabolismo C1, la biosíntesis de monolignoles y la asimilación de amonio mediante el ciclo GS/GOGAT (figura 11.1), propuesta por Cantón y colaboradores [39]. Se buscaron los genes de las 14 enzimas que participan en esta ruta por tres vías diferentes: Por un lado se realizó una búsqueda por palabras clave en EuroPineDB, introduciendo el nombre del producto del gen en la ventana de búsqueda de la web de EuroPineDB. Por otro lado, y del mismo modo, se utilizó el código EC de cada enzima en la búsqueda de la web, y por último se trató de encontrar estos genes con la implementación de Blast de la web de EuroPineDB. Para la búsqueda por similitud con Blast era necesario contar con un ortólogo de cada gen, por lo que se descargó de UniProt [15] las secuencias de proteínas de los ortólogos más cercanos al pino (véase la columna *Ortólogo* de la tabla 11.1). Para encontrar estos ortólogos se introdujo en el campo de búsqueda de la web de UniProt (`http://www.uniprot.org/`) los códigos EC de cada enzima (véase la columna *Enzima* de la tabla 11.1).

Los resultados obtenidos (véase las columnas *palabras clave, EC y* Blast de la tabla 11.1) muestran que cada una de las enzimas esta representada por varios unigenes, algo que era de esperar, puesto que el ensamblaje *P.pinaster* de EuroPineDB se realizó únicamente con MIRA3, un ensamblador altamente redundante [161], como se comentó en el apartado 10.4. El número de enzimas encontrado utilizando cualquiera de los tres métodos propuestos (palabras claves, EC y Blast), es equivalente en casi todos los casos, salvo para la MS, que muestra un número inesperadamente bajo en la búsqueda con palabras clave (3 unigenes encontrados), frente al número de unigenes encontrados con las EC y Blast (27 y 30 respectivamente). Algo similar, pero no tan llamativo, ocurre con las enzimas SAMS y SAHH (véase la tabla 11.1). La explicación es que las descripciones de los genes no siguen un vocabulario controlado, y hay que tener en cuenta los múltiples nombres que se pueden asignar a un mismo gen. Por ejemplo, en el caso de la MS, en la búsqueda por palabras clave se encontró 3 unigenes utilizando la descripción *methionine synthase*, pero además, hay otros 20 unigenes de la MS que contienen la descripción *5-methyltetrahydropteroyltriglutamate-homocysteine expressed*, y otros 3 con *vitamin-b12 independent methionine 5-methyltetrahydropteroyltriglutamate-homocysteine*, lo que hace un total de 26 Unigenes de MS. De igual modo, en el caso de las SAMS, el resultado varía según se introduzca *S-adenosylmethionine synthetase*, con *synthase* en lugar de *synthetase* o separando *S-adenosylmethionine* en *S-adenosyl* y *methionine*. Al realizar todas las combinaciones se encuentran un total de 37 unigenes diferentes. En el caso de la SAHH, aparecieron 11 unigenes buscando la descripción *adenosyl homocysteine hydrolase* y 6 más como *adenosylhomocysteinase*. Sin embargo, aunque la búsqueda por palabras clave en la descripción del producto del gen no tenga un vocabulario controlado, este método es imprescindible, puesto que no todos los genes codifican enzimas que se pueden encontrar con su código EC, y no siempre se puede contar con un ortólogo para realizar la búsqueda con Blast.

En el caso de las cSHMT y mSHMT, dos enzimas muy parecidas, pero de localización citosólica y mitocondrial respectivamente se encuentran los mismos unigenes por búsqueda de palabras clave y de EC en ambas (16 y 18 respectivamente) (véase la tabla 11.1), puesto que no hay ninguna información que les permita discriminar cuál es cada una de las dos. En cambio, utilizando Blast se encuentran 7 unigenes más del gen de localización citosólica (cSHMT) que del de localización mitocondrial (mSHMT) (tabla 11.1). Al utilizar Blast, los unigenes más parecidos a los genes que se buscaron aparecen de modo ordenado, del que más se parece al que menos. En este caso, ambos genes son muy parecidos y aparecerán mezclados, aunque seguramente se encontrarán los candidatos más probables en la parte superior de cada una de las listas de ortólogos obtenidas con Blast. Los valores mostrados en la tabla 11.1, muestran que hay más unigenes de cSHMT que de mSHMT, ya que al menos 7 de ellos solo muestran similitud con cSHMT.

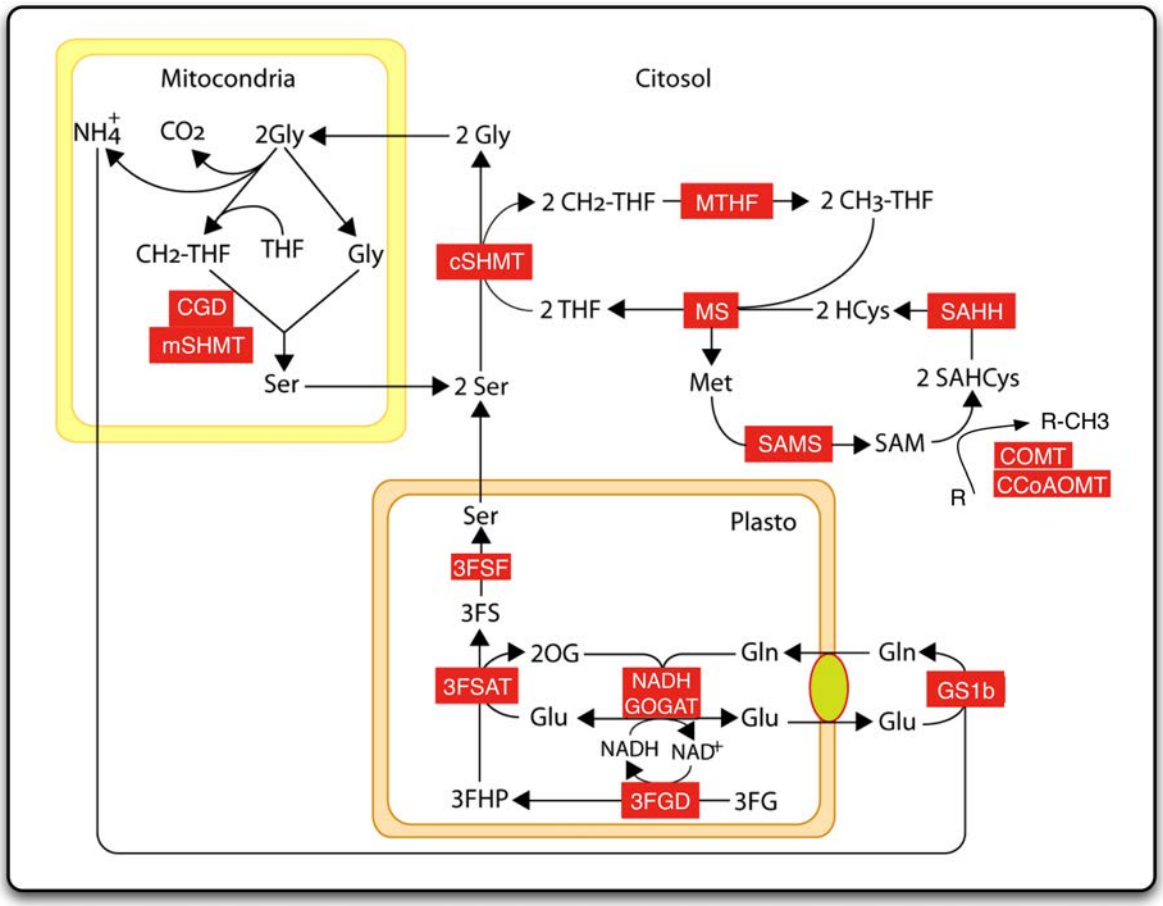


Figura 11.1: *Representación esquemática de la conexión entre el metabolismo C1, la biosíntesis de monolignoles y la asimilación de amonio mediante el ciclo GS/GOGAT. El significado de los acrónimos de los genes se encuentra en la tabla 11.1. Imagen tomada de [226]*

Los 14 genes de esta ruta cumplen funciones básicas para el crecimiento del pino, y todos están representados en EuroPineDB. Sin embargo, son muy pocos genes para poder saber si el transcriptoma incluido en EuroPineDB contiene una buena representación del transcriptoma del pino. No obstante, los resultados anteriores sugieren que EuroPineDB contiene una buena representación de los genes que cumplen funciones básicas y de mantenimiento en el pino. Las enzimas de esta misma ruta se buscaron también en la nueva versión obtenida del transcriptoma de pino (véase el apartado 11.4.5).

Tabla 11.1: *Identificadores de las enzimas de la ruta que conecta el metabolismo C1, la biosíntesis de monolignoles y la asimilación de amonio.*

Nombre del gen	Producto del gen	Longitud (Aas)	Ortólogo		Enzima	Palabras clave	EC	Blast
MS	Metionina sintasa	766	G3C8U7	*P. pinaster*	EC:2.1.1.14	3	27	30
SAMS	S-adenosilmetionina sintetasa	391	G3C8Z6	*P. pinaster*	EC:2.5.1.6	10	37	38
SAHH	S-adenosilhomocisteína hidrolasa	485	G3C8Z5	*P. pinaster*	EC:3.3.1.1	11	18	18
cSHMT	Serina hidroximetiltransferasa cit.	470	G3C904	*P. pinaster*	EC:2.1.2.1	16	18	19
msSHMT	Serina hidroximetiltransferasa mit.	523	G3C8Z4	*P. pinaster*	EC:2.1.2.1	16	18	12
3FGD	3-fosfoglicerato deshidrogenasa	624	O04130	*A. thaliana*	EC:1.1.1.95	8	7	8
NADH-GOGAT	Glutamato sintasa dep. de NADH	2208	Q9LV03	*A. thaliana*	EC:1.4.1.14	7	5	10
MTHFR	Metilene-tetrahidrofolato reductasa	594	G3C900	*P. pinaster*	EC:1.5.1.20	5	5	5
COMT	Ácido cafeico O-metiltransferasa	364	Q5NDD5	*P. abies*	EC:2.1.1.68	29	16	28
CCoAOMT	Cafeoil-CoA O-metiltransferasa	259	Q9ZTT5	*P. taeda*	EC:2.1.1.104	35	16	31
3FSAT	3-fosfoserina aminotransferasa	433	G3C902	*P. pinaster*	EC:2.6.1.52	2	2	2
3FSF	3-fosfoserina fosfatasa	295	O82796	*A. thaliana*	EC:3.1.3.3	3	4	3
GS1b	Glutamina sintetasa 1b	355	Q9ZS52	*P. sylvestris*	EC:6.3.1.2	19	19	17
CGD [1]	Complejo glicina descarboxilasa	167	G3C8Z3	*P. pinaster*	-	5	-	6

[1] Se utilizó como referencia la proteína H del complejo glicina descarboxilasa.
Los resultado de la columna Blast se obtuvieron utilizando tBlastn en la web de EuroPineDB y filtrando la salida con un valor de $E < 10^{-25}$.

11.2.2. Genes del metabolismo del nitrógeno en EuroPineDB

Como complemento para comprobar hasta qué punto las rutas están completas se decidió buscar genes del metabolismo del nitrógeno en EuroPineDB. Para ello se utilizó la ruta del metabolismo del nitrógeno de *KEGG pathways* (figura 11.2), donde hay 49 enzimas porque se incluyen las reacciones metabólicas de todos los organismos. Marcados en verde en la figura 11.2 se muestran las 14 enzimas que existen en las plantas, mientras que el resto de las enzimas representan reacciones que se existen en otros organismos pero no en las plantas.

Al realizar la búsqueda de los códigos EC en la aplicación de búsqueda por palabras clave de EuroPineDB se encontraron 11 de las 14 enzimas marcadas en verde en la figura 11.2 (resultados no mostrados), todas menos la EC:1.4.1.13 (glutamato sintasa dependiente de NADPH), la EC:1.4.1.4 (glutamato deshidrogenasa (NADP(+)) y la NitF (flavodoxina o dinitrógeno oxidoreductasa). La última (marcada con una flecha y un círculo rojo en la figura 11.2) se descartó de la búsqueda puesto que interviene en la fijación del nitrógeno atmosférico, una reacción que solo la llevan a cabo unas bacterias que se encuentran en simbiosis en las raíces de las leguminosas. Por tanto se buscó un ortólogo de EC:1.4.1.13 y otro de EC:1.4.1.4 en UniProt, y ambos se utilizaron para localizar sus homólogos en EuroPineDB. Se encontraron unigenes parecidos a estas enzimas, pero entre estos, se encontraron también los que están anotados como las enzimas EC:1.4.1.14 (glutamato sintasa dependiente de NADH) y EC:1.4.1.3 (glutamato deshidrogenasa (NAD(P)(+))) respectivamente, puesto que su secuencia es muy parecida.

Por tanto, EuroPineDB parece contener los 14 genes de este fragmento del metabolismo del nitrógeno que podrían tener los pinos, lo que confirma que posee una buena representación del transcriptoma de pino, y que podrá ser útil para localizar otros muchos genes.

11.2.3. Diseño del Pinarray2

En cuanto se contó con la base de datos EuroPineDB se planteó el diseño de una nueva micromatriz de pino que completase el Pinarray1 (apartado 8.2, pág. 61) con nuevas sondas procedentes de las distintas genotecas de ADNc que se utilizaron para la construcción de EuroPineDB. Para ello se decidió realizar una selección de secuencias separadas por especies (*P. pinaster*, *P. sylvestris* y *P. pinea*), que no fueran redundantes entre sí, ni con las secuencias ya impresas en el Pinarray1. Los unigenes seleccionados necesitan tener respaldo de los clones de ADNc, puesto que en la micromatriz hay que imprimir una molécula de ADNc amplificada por PCR. Por este motivo, el ensamblaje incluido en EuroPineDB para *P. pinaster*, que contenía secuencias de 454 no pudo utilizarse tal cual, sino que se utilizó el denominado *P.pinaster_sanger* (`http://www.scbi.uma.es/pindb/assemblies`) que tenía solo lecturas de tipo Sanger, que hay que recordar, que fueron preprocesadas con SeqTrim y ensambladas con CAP3.

De cada ensamblaje (*P.pinaster_sanger*, *P.sylvestris* y *P.pinea*), el grupo de secuencias que se iba a imprimir en la micromatriz debía cumplir los siguientes criterios con vistas a evitar la redundancia de secuencias:

- Si en un contig hay una secuencia que contiene a todas las demás se selecciona esa secuencia (representada en azul en la figura 11.3-A).
- Se seleccionan un máximo de 2 secuencias por contig, de modo que se escoge siempre la que está más hacia 5', y si ésta no solapa con la secuencia del extremo 3', se seleccionan las 2 (secuencias en azul, figura 11.3-B). Si solapan se selecciona únicamente la secuencia del extremo 5' (figura 11.3-C). Hay que recordar que las secuencias proceden de un clon, por lo que si la secuencia es solapante, el clon más hacia 5' contendrá también la secuencia más a 3'. El clon en 3' se seleccionó cuando la secuencia no era solapante porque siempre existe la duda de que no se trate del mismo gen, alelo o parálogo.
- Se seleccionaron todos los singulones.
- En caso de igualdad de condiciones se seleccionan de modo prioritario las secuencias procedentes de una genoteca de nuestro grupo de investigación.

A continuación se descartaron los clones que ya estaban representados por otra secuencia de Pinarray1 para eliminar redundancia. Finalmente se entregó la lista de clones seleccionados a Sara Díaz Moreno, que se encargaría de preparar las sondas que se iban a imprimir en la nueva micromatriz.

Cada vez que algún clon no se podía amplificar o no se podía utilizar por cualquier motivo se buscó un sustituto según los siguientes criterios:

- Los singulones no tienen sustituto, por lo que simplemente se descartaban.
- Las secuencias del extremo 5' se sustituyen por la siguiente secuencia de 5'.
- Las secuencias del extremo 3' se sustituyen por la secuencia anterior de 3'.

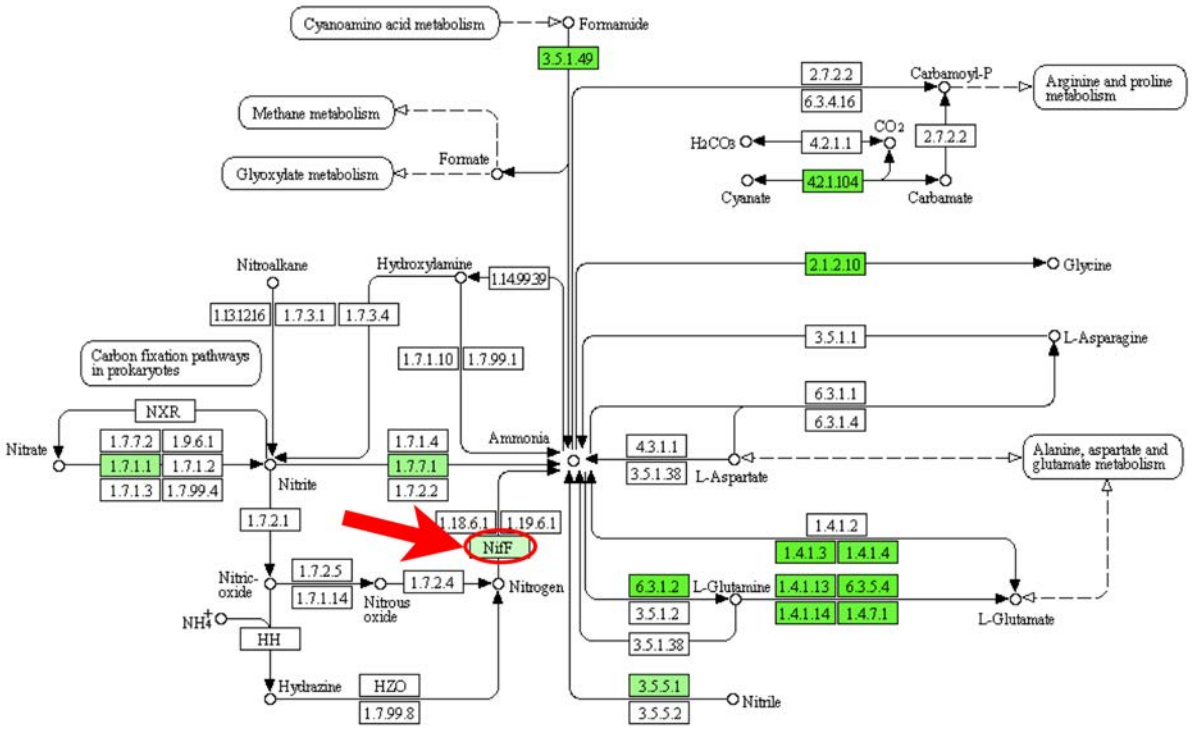


Figura 11.2: *Conjunto de enzimas relacionadas con el metabolismo del nitrógeno según KEGG, en donde aparecen enzimas de todos los organismos. Se resaltan en verde los genes que KEGG tiene catalogado que existen en las plantas. Con flecha y círculo rojo se resalta el gen NitF, que codifica una enzima presente solo en las leguminosas.*

Los clones se ordenaron en listas en formato de texto tabulado que representan las coordenadas de placas de 96 pocillos y en el formato informático requerido para el software del robot de impresión. Entre las placas originales de los clones y las placas de 384 pocillos definitivas que se usaron para imprimir la micromatriz se produjeron varios reordenamientos intermedios (Sara Díaz Moreno, comunicación personal) que se describen a continuación (figura 11.4):

1. Se parte de las placas de microtitulación de 96 pocillos con los cultivos bacterianos originales de los clones. Los clones seleccionados para su impresión en el Pinarray1 se pasan ordenados a una nueva placa de 96 pocillos para hacerlos crecer (figura 11.4.1).

2. Los clones que crecieron se pasaron a placas de 96 pocillos para PCR (figura 11.4.2), donde se realizan las PCR para amplificar los insertos a partir de los cultivos bacterianos. Un 10 % del volumen (10 μl) de cada PCR se analiza en un gel para comprobar si hay amplificado. En caso negativo, o que sea muy tenue, o que haya dos amplificados, ese pocillo descarta (pocillos con X en la figura 11.4.2).

3. Se eliminan los huecos vacíos reordenando las PCR validadas en nuevas placas de 96 pocillos (figura 11.4.3). Las PCR que no funcionaron se intentan amplificar de nuevo en otras placas y se realiza el mismo procedimiento (análisis en gel y eliminación de los huecos). Una vez que están todos los productos de PCR en placas sin huecos se purifica el ADN y se vuelve a analizar en gel. Puede que algunos amplificados se pierdan, en cuyo caso se vuelve a reordenar en placas de 96 nuevas para eliminar los huecos.

4. Por último, los ADN amplificados por PCR, purificados y reordenados, pasan a las placas de 384 pocillos (figura 11.4.4) que se utilizarán para imprimir la micromatriz (figura 11.4.5).

Todo estos reordenamientos quedaron cuidadosamente registrados porque, a partir de ellos, se realizó una compleja redistribución que permitió relacionar las coordenadas originales y finales de cada clon utilizado, para poder conocer qué clon se había impreso en cada punto. Este proceso se realizó utilizando varios *scripts* de Ruby y realizando minuciosas comprobaciones de los datos para evitar cualquier error que pudiera provocar una mala identificación de los clones impresos en Pinarray2. Con esta información y con el fichero `data_tracking` que devuelve el software del robot de impresión se cons-

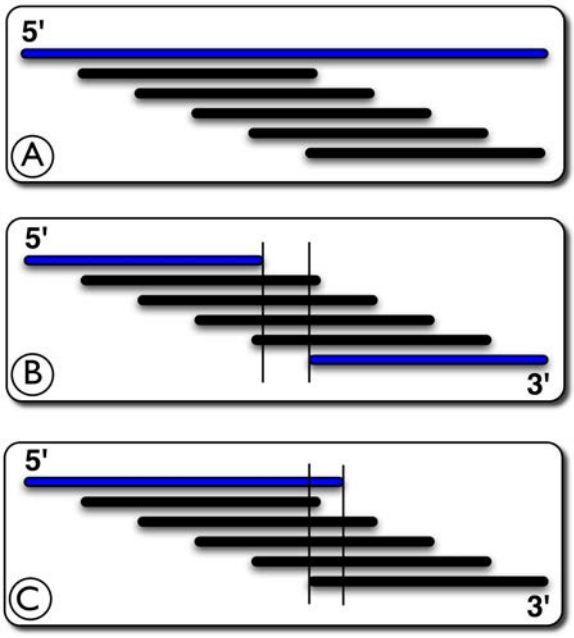


Figura 11.3: *Selección de las secuencias impresas en Pinarray2. Las secuencias seleccionadas se marcan en azul. (A) Se selecciona la secuencia del clon que contiene a todas las secuencias del contig. (B) Se seleccionan la secuencias de los extremos 5' y 3' cuando no solapan entre sí. (C) Si la secuencia del extremo 5' solapa con todas las secuencias del contig, solo se selecciona la secuencia del extremo 5'.*

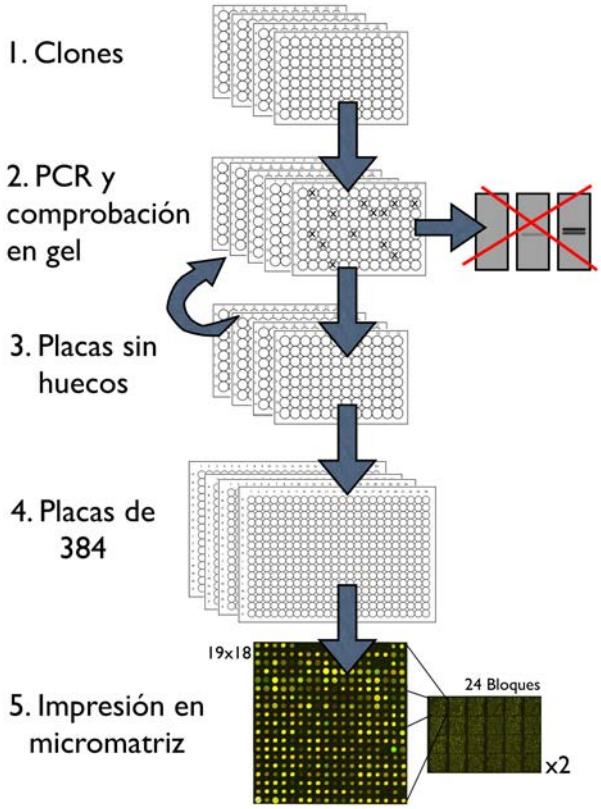


Figura 11.4: *Reordenamiento de clones desde las placas de origen hasta su impresión en el Pinarray2.*

truyó el fichero GAL correspondiente a esta nueva micromatriz. Gracias a que ya se tenía la base de datos EuroPineDB se sabía el nombre del clon que había en cada coordenada, su número de acceso en EMBL, y su descripción. Todo ello se incorporó también al fichero GAL, como se hizo anteriormente para el Pinarray1.

En un principio se seleccionaron 10 036 clones, (3456 de Pinarray1 + 6580 secuencias nuevas). De los clones nuevos, 6261 eran de *Pinus pinaster*, 305 de *Pinus sylvestris* y 14 de *Pinus pinea*. Sin embargo, tras descartar los clones inservibles y sustituirlos por otros, cuando fue posible, se acabaron imprimiendo 8208 sondas en el Pinarray2, distribuidas en 24 bloques de 19 filas y 18 columnas (figura 11.4,5). Al igual que en el Pinarray1, en cada portaobjetos se imprimieron dos copias de las sondas, con lo que con una hibridación ya se obtenían dos réplicas técnicas.

Esta nueva micromatriz se ha probado recientemente en nuestro laboratorio para comparar plantas de diferentes poblaciones y para analizar líneas transgénicas en las que se ha silenciado un gen. Los resultados indican un funcionamiento correcto de la micromatriz (J. Canales, comunicación personal).

11.3. Mejoras técnicas de la base de datos

Es fácil comprobar que la base de datos EuroPineDB no se preprocesó con el programa más adecuado (la genoteca de 454 que contiene se limpió con SeqTrim en lugar de SeqTrimNext) ni se ensambló con el protocolo que hoy consideramos que es el más óptimo (se utilizó CAP3 para las secuencias de Sanger y MIRA3 para las de 454, con lo que el transcriptoma está seguramente sobrevalorado), pero está siendo de utilidad para los grupos de investigación que contribuyeron a su creación, ya que se han clonado varios ADNc y promotores a partir de la información que contienen (C. Collada, C. Ávila, F.R. Cantón, comunicación personal). Por otra parte, nuestro grupo de investigación comenzó a participar en un proyecto internacional en el que, entre otras cosas, había que completar el transcriptoma del pino marítimo, lo que puso a nuestra disposición nuevas lecturas de NGS y esto obligó a desarrollar una nueva base de datos adaptada a la nueva situación. Por razones de confidencialidad, esta nueva base de datos, que llamaremos SPDB, no está aún accesible al público, sólo a los participantes en dicho proyecto.

11.3.1. Versiones y terminología

EuroPineDB es una base de datos descrita anteriormente, que abreviaremos a veces como EPDB para referirse a la versión que se describe en el apartado 11.1, y que se denomina EPDB2 cuando se refiere a la nueva versión obtenida con el flujo de trabajo (apartado 10.4, pág. 135). Para SPDB, la base de datos que contiene más secuencias de NGS se cuenta con el contenido de EPDB junto con 3 librerías más de 454/FLX (véase el apartado 8.3, pág. 62). Se emplearán distintos sufijos para identificar las diferentes versiones que de ella se han realizado. obtenido en EuroPineDB.

En EuroPineDB se utilizaban los términos genoteca y ensamblaje del siguiente modo: cada *ensamblaje* contiene los unigenes obtenidos, y cada *genoteca* contiene los clones de la base de datos, de igual modo que los del laboratorio. Sin embargo, como las nuevas aportaciones de lecturas NGS no cuentan con el respaldo de clones, esta terminología se ha tenido que variar levemente en la nueva base de datos, y los datos ahora se organizan en ensamblajes, librerías y proyectos:

- **Librerías**: contienen las lecturas de NGS de un experimento; se corresponden con el término «genoteca» en EPDB.
- **Ensamblajes**: al igual que en EPDB, están formados por los unigenes que se obtienen de una librería.
- **Ensamblaje global**: hace referencia al ensamblaje de todas las librerías a la vez para obtener una única colección de unigenes que representa al transcriptoma.
- **Proyectos**: engloban al conjunto de datos de las librerías, del ensamblaje (global o no), y de las anotaciones de un mismo experimento.

Muchas de las mejoras realizadas en la nueva base de datos están destinadas a agilizar su actualización. Cada vez que se añade una librería nueva, ésta se incluye en un nuevo proyecto, junto con el ensamblaje realizado con sus lecturas. Además, al añadir una nueva librería, también se crea otro proyecto nuevo, donde incluir una nueva versión del ensamblaje global con las lecturas de todas las librerías disponibles hasta ese momento. Cada nuevo ensamblaje global se considera una ***versión*** (1, 2, 3, etc.) del transcriptoma de pino diferente. Cuando se recalcula el ensamblaje global porque alguno de los programas del flujo incluye alguna mejora significativa se cambia la *subversión* del ensamblaje (1.1, 1.2, 1.3, etc.). En este trabajo se presentará hasta la versión 1.2, aunque en el grupo de investigación ya se está trabajando con las versiones 2.0 y 2.1.

11.3.2. Simplificación de las tablas

En primer lugar se simplificaron las tablas de la base de datos relacional porque ya no se necesitaba almacenar información relacionada con los clones de ADNc, las placas de 96 y 384 pocillos y las micromatrices, como se muestra en la figura 11.5. Como las nuevas librerías que se iban a incorporar (y que acabarán suponiendo más del 99 % del contenido) proceden de experimentos de NGS, el eje principal de la estructura de las tablas relacionales pasó de depender de los clones de cada genoteca en EuroPineDB (tabla *Sequence* de la figura 11.5-A) a los unigenes de cada ensamblaje en SPDB (figura 11.5-B).

También se ha modificado significativamente la manera de almacenar las anotaciones. En EuroPineDB, todas las anotaciones se encontraban en tablas diferentes (figura 11.5-A), mientras que ahora se encuentran reunidas en una sola tabla (figura 11.5-B): la tabla *Annotation*. Esta tabla relaciona cada unigén con sus términos GO, el código EC para cada proteína que tiene actividad enzimática, los términos de InterPro para identificar familias génicas, y varias descripciones de los posibles productos génicos (véase como se obtienen en el apartado 10.3). Pero no todas las anotaciones se pudieron simplificar de igual forma. Por ejemplo, Las rutas metabólicas de KEGG se almacenan en una tabla aparte porque están directamente relacionadas con las enzimas que aparecen en la ruta metabólica (tabla *Annotation* de la figura 11.5-B). De este modo se puede controlar las enzimas que están contenidas en cada ruta metabólica y se pueden recuperar fácilmente, mostrando el mapa de la ruta metabólica con las enzimas que contiene la base de datos resaltadas en amarillo (figura 11.6). Para ello que se utiliza el interfaz de programación de aplicaciones (del inglés *Application Programming Interface*) (API) de la web de KEGG con BioRuby (véase **bio v1.4.2**, apartado 7.1.3, pág. 48).

La información de los SNP y de los SSR, al igual que la de las rutas KEGG, tampoco quedó incluida dentro de la tabla *Annotation*, porque la información que se necesita almacenar de los SNP y las SSR es diferente a la de otras anotaciones: en el caso de las descripciones, GO, EC, InterPro y KEGG, la información viene dada por un código y una descripción, mientras que en los SNP y SSR, la información que aportan indica en que posición de la secuencia se encuentra y cuales son los nucleótidos que varían o se repiten (tablas *Snp* y *Ssr* en la figura 11.5-B).

Por último, también se simplificó la forma de navegar por la web: compárese la figura 11.7 correspondiente a la nueva versión, y la figura 3 del artículo de EuroPineDB, en el apartado anterior. Puede

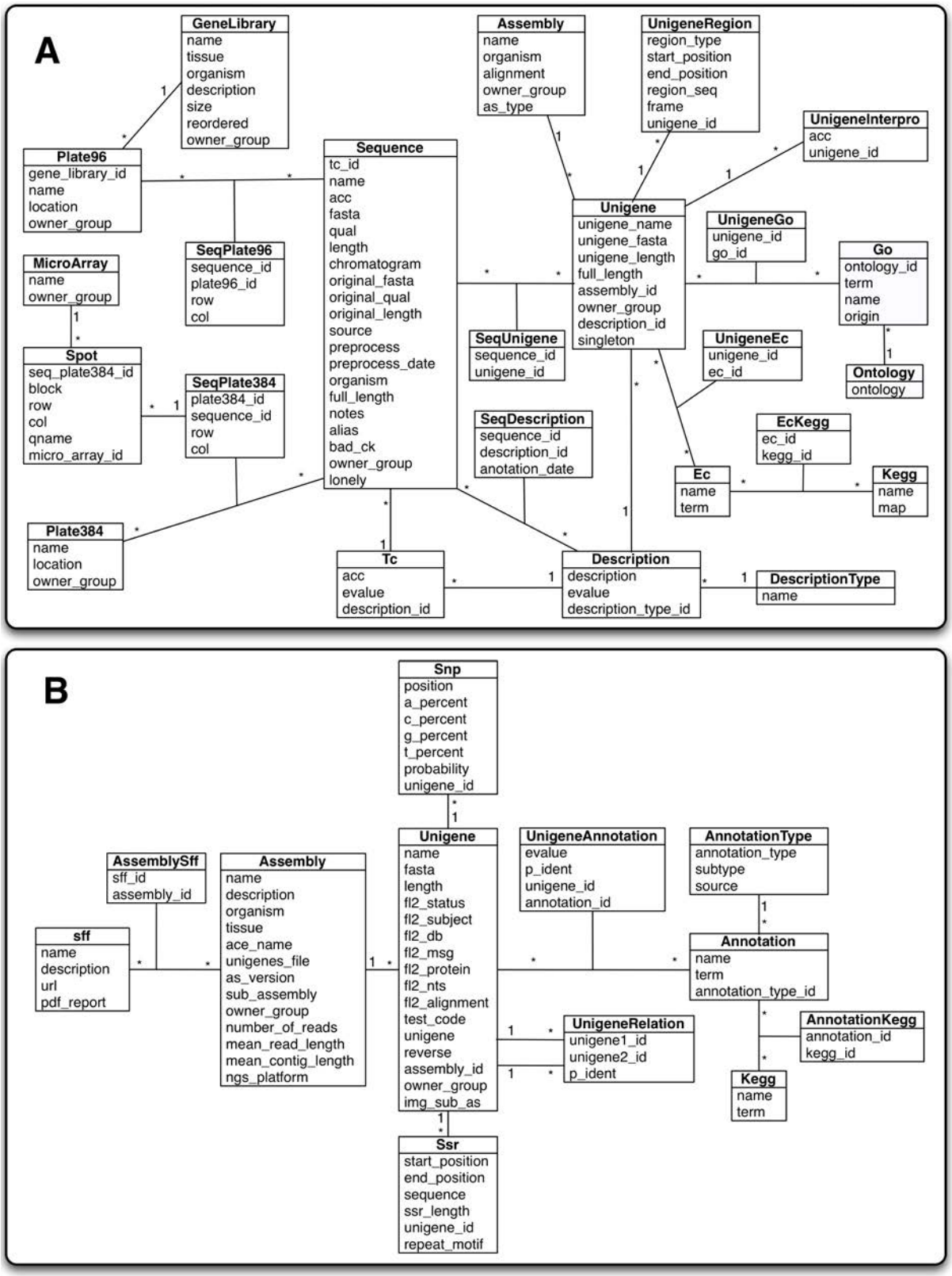


Figura 11.5: *A: Esquema de las tablas de la base de datos EuroPineDB, que se centra sobre cada secuencia. B: Esquema de las nuevas tablas simplificadas, centrada en los unigenes, que se está utilizando para las nuevas bases de datos, entre ellas SPDB.*

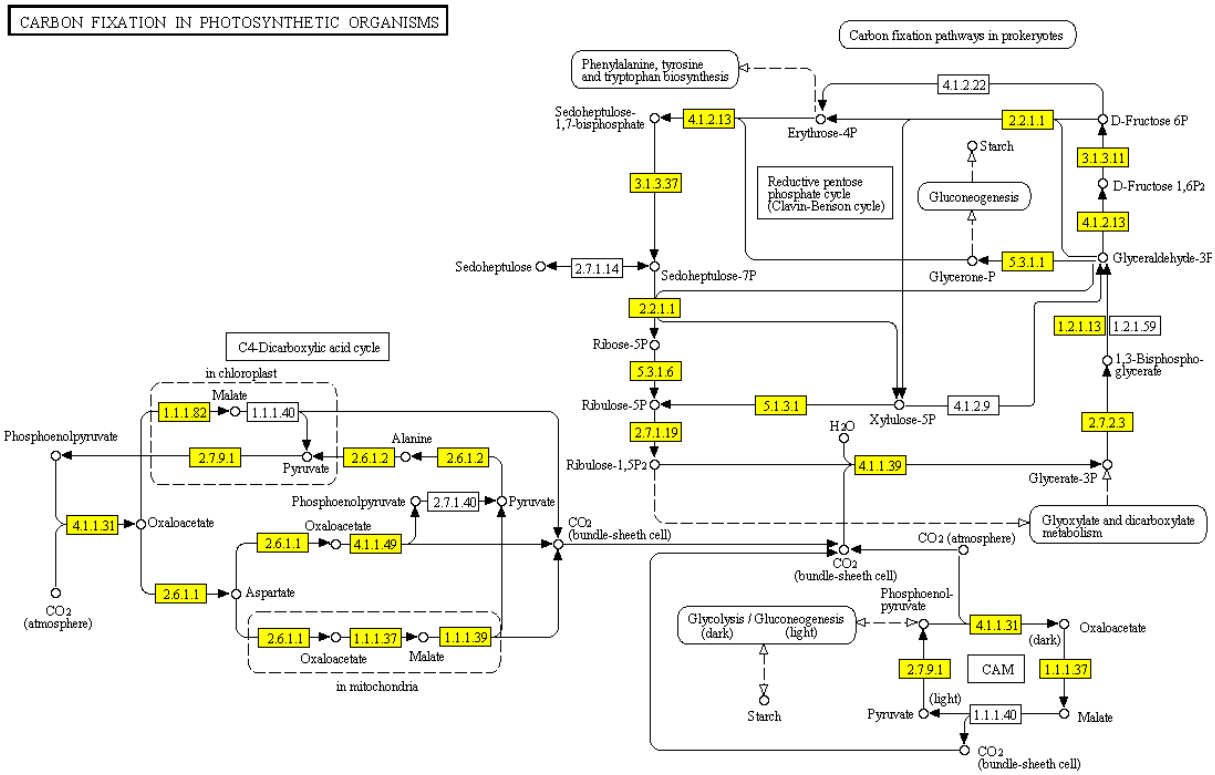


Figura 11.6: *Ejemplo de un mapa de KEGG, con la ruta fijación de carbono en organismo fotosintéticos, en el que las enzimas anotadas en SPDB aparecen resaltadas en amarillo.*

verse que la navegación a través de los ensamblajes y sus unigenes ocupa la posición central en el esquema de la nueva versión, mientras que antes ocupaba una posición lateral, puesto que eran una información adicional junto con los clones y su distribución en las placas de 96 y 384 pocillos, y el Pinarray. En SPDB, lo importante es la información que contiene cada versión del transcriptoma (cada ensamblaje) y sus unigenes, y para llegar a esa información, al igual que en la web de EuroPineDB, hay tres caminos: a través de búsquedas por palabras claves, con BLAST, o navegando por la información de cada unigén del ensamblaje.

11.3.3. Anotación más ágil

Ya se ha comentado que la anotación de los unigenes es lo que más tiempo consume dentro del flujo de trabajo propuesto para transcriptómica (apartado 10.4). Para reducir este tiempo, los ensamblajes que no son globales, solo se anotan con SSR, SNP y la información procedente de FULL-LENGTHERNEXT al ser procesos rápidos. La anotación con Blast2GO y AutoFact se realiza únicamente para el ensamblaje global que incluye a todas las librerías. De este modo, cada uno de los unigenes de ensamblajes de una sola librería heredan las anotaciones del unigén con mayor número de lecturas en común con el ensamblaje global (relación que se almacena en la tabla *UnigenRelation* de la figura 11.5-B).

En un futuro próximo se intentará añadir mejoras en la visualización de las anotaciones. Por ejemplo, ya se han hecho algunas pruebas para resaltar en color las descripciones de los productos de los genes cuando aparecen palabras en común (D. Guerrero, datos no mostrados). De este modo se permitiría al usuario verificar de un vistazo si todos los programas de anotación proporcionan descripciones coherentes.

11.3.4. Ficheros auxiliares para la base de datos

Uno de los motivos por los que se utilizan bases de datos es para poder acceder rápidamente a los datos. Pero esta ventaja se convierte en inconveniente cuando la base de datos contiene tablas con muchas entradas, o las entradas son muy voluminosas. Por eso se ha optado por indexar los campos más consultados y por almacenar parte de la información más voluminosa en ficheros, algunos de los cuales también están indexados a su vez. Estos ficheros son, por ejemplo, ficheros SFF originales que contienen las lecturas de cada librería, el informe en PDF de SeqTrimNext de cada librería, y los ficheros

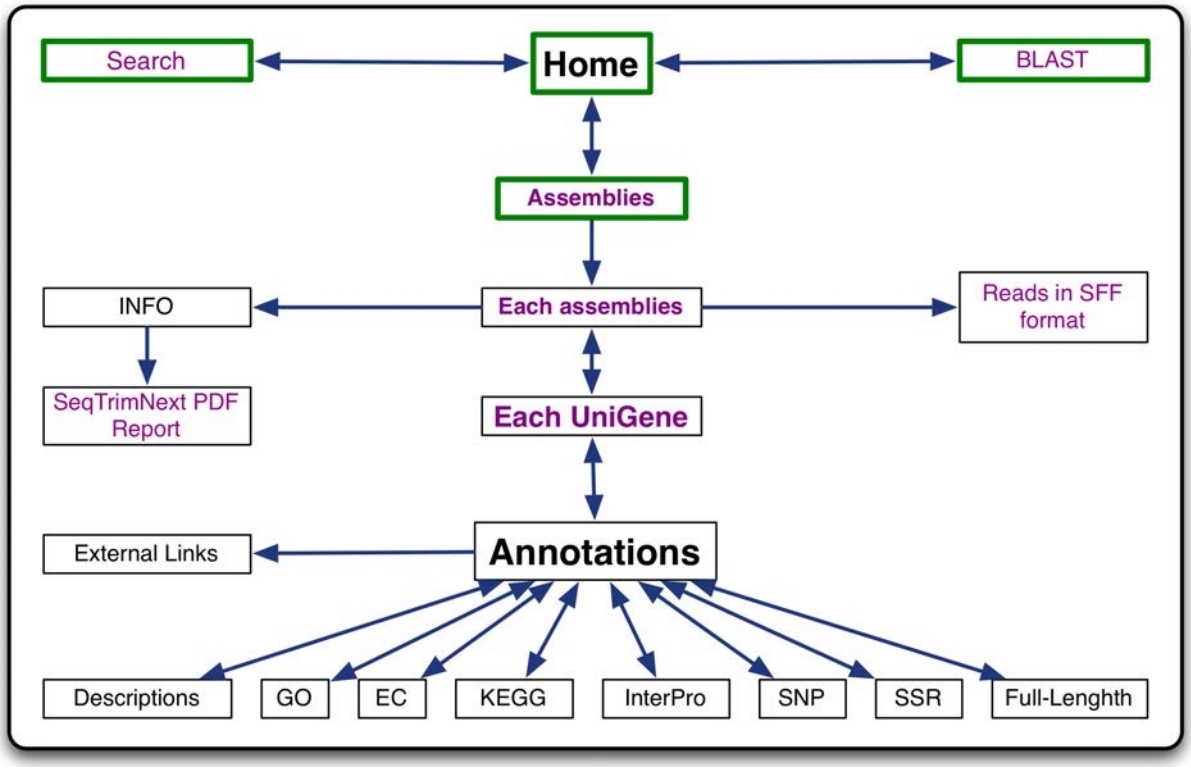


Figura 11.7: *Nuevo esquema de navegación de la nueva versión de la base de datos. Las flechas indican la dirección de la navegación. Las cajas verdes corresponden a las páginas principales de la web. El texto en color violeta indica que contiene la opción de descarga.*

en formato ACE del ensamblaje. Este último es de los que están indexados, gracias a un *script* de D. Guerrero, por lo que es posible acceder al ensamblaje de cualquier unigén con rapidez y sin tener que descargar ni recorrer el fichero completo. El fichero ACE descargado con el alineamiento de las lecturas que forman el unigén puede visualizarse fácilmente con herramientas bioinformáticas como Tablet (apartado 7.3.14, pág. 59).

11.3.5. Automatización de la importación de datos

Para facilitar el tedioso proceso de importación de datos a las bases de datos se crearon *scripts* de importación que la automatizaban. Así pues, una vez que se tiene la información necesaria para montar o actualizar la base de datos se ejecutarán estos *scripts* para importar desde pequeñas actualizaciones de los datos hasta varios proyectos nuevos simultáneamente. Por supuesto, la importación de datos solo la realizan los administradores de las bases de datos, por lo que no es necesario que los usuarios conozcan esta información ni cómo manejar los *scripts*.

Entre los directorios del proyecto de Ruby-On-Rails que controla la base de datos (apartados 4.1, 5 y 6.2) se ha añadido una serie de carpetas para organizar la importación de los proyectos (resaltadas en verde en la figura 11.8). Por un lado está la carpeta `import_new_projects`, que contiene un fichero `readme.txt` en el que se indica cómo y dónde colocar los ficheros que contienen la información que se va a importar. También hay una carpeta por cada proyecto que se desea importar, ya que el *script* de importación permite importar varios proyectos a la vez y de modo automático. Dentro de cada carpeta de proyecto se añaden las siguientes carpetas (resaltadas en azul en la figura 11.8):

- `annotations`: incluye a su vez una carpeta por cada uno de los programas utilizados para la anotación. A continuación se describen los que se han utilizado en este trabajo, pero el *script* se puede personalizar para incluir información de otros programas utilizados (por ejemplo, anotación con Sma3s).
  - `autofact`: se debe incluir el fichero con extensión *out* que devuelve AutoFact y que contiene la información obtenida por el programa con los unigenes del proyecto.
  - `blast2go`: debe contener dos ficheros de texto tabulado con las anotaciones de

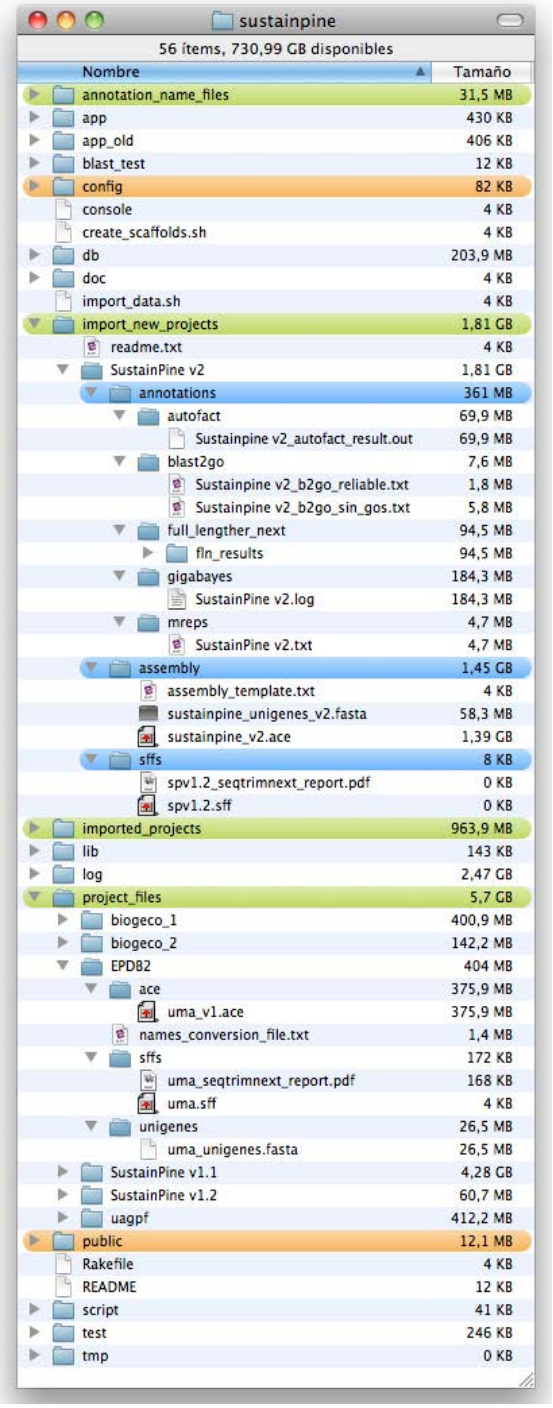


Figura 11.8: *Estructura de ficheros necesaria para manejar la interfaz y la base de datos. Las carpetas resaltadas en verde corresponden a las que contienen información necesaria para la importación automática. Se ha desplegado el contenido de* `import_new_projects`*, que muestra el contenido de proyecto que va a ser importado, y de* `project_files`*, que muestra los ficheros de un proyecto ya importado. Las carpetas en naranja contienen ficheros para controlar los datos específicos de cada base de datos.*

Blast2GO, exportados como se indica en el apartado 7.3.4, para poder diferenciar los términos de la *Gene Ontology* fiables de los que no lo son. Estos se nombrarán como `project_b2go_reliable.txt` y `project_b2go_sin_gos.txt` respectivamente.

- `full_lengther_next`: dentro únicamente hay que incluir la carpeta de resultados, `fln_results`, obtenida con FullLengtherNext.
- `gigabayes`: se debe incluir el fichero con extensión *log* que contiene los resultados de GigaBayes para la detección de SNP.
- `mreps`: en esta carpeta se introduce un fichero de texto con el resultado de MREPS para la búsqueda de SSR.

- `assembly`: en esta carpeta deben introducirse tres ficheros:
  - `assembly_template.txt`: contendrá toda la información del proyecto requerida por el *script* de importación, como el nombre y la descripción del proyecto, y el nombre de los ficheros de lecturas (SFF), alineamientos (ACE) y unigenes (fasta) que se van a importar, además de indicar si esta importación va a actualizar a otra con el mismo nombre. Esta información es aportada por los administradores de la base de datos. En el apéndice H puede encontrarse un ejemplo de este fichero.
  - `project.ace`: fichero en formato ACE con el ensamblaje de las secuencias del proyecto (en el caso de utilizar el flujo de trabajo comentado en el apartado 10.4 se unirán los ficheros ACE de MIRA3 y CAP3 con un *script* desarrollado por H. Benzekri).
  - `project_unigenes.fasta`: fichero en formato fasta con los unigenes obtenidos al ensamblar las secuencias del proyecto.

- `sffs`: en esta carpeta se deben introducir los ficheros SFF con las lecturas obtenidas en la secuenciación, y el informe del preprocesamiento en formato PDF que genera SeqTrimNext.

También se han añadido dos carpetas más para almacenar los datos importados. Por un lado, `project_files` (resaltado en verde en la figura 11.8), donde se almacenan los ficheros que van a solicitarse desde la web y que son incluidos en esta carpeta de modo automático por el *script* de importación. Estos ficheros son las lecturas en formato SFF,

el informe del preprocesamiento en PDF creado por SeqTrimNext, los ficheros en formato ACE con la información del ensamblaje y el fichero de unigenes en formato fasta. Por otro lado, `imported_projects` (resaltado en verde en la figura 11.8), donde se guardan los datos de importación de los proyectos que ya han sido importados. Al acabar la importación del proyecto a la base de datos, el último paso del *script* de importación consiste en mover los ficheros desde la carpeta `import_new_projects` a esta carpeta. De este modo, lo que está dentro de `import_new_projects` está aún por importar y si se necesita repetir alguna importación o acceder a alguno de los ficheros originales de un proyecto ya importado se sabe que toda esa información queda almacenada en `imported_projects`. Además, con el fin de homogeneizar los datos, los unigenes reciben un nombre correlativo al importarse —que lleva el nombre corto del proyecto y su versión, por ejemplo `sp_v1_unigene1`, donde «sp» sería el nombre corto para el proyecto SustainPine, «v1» indicaría la versión y «unigene1» el nombre de ese unigén—. La relación entre los nombres originales de los unigenes (los que da el ensamblador) y los nombres nuevos asignados en la base de datos quedan almacenados en el fichero `names_conversion_file.txt`, lo que resulta imprescindible para relacionar posteriormente ficheros de anotación que tengan el identificador asignado por el ensamblador con la información final de la base de datos.

Se ha añadido una carpeta `annotation_name_files` (resaltada en verde en la figura 11.8) que incluye la relación de los términos GO, EC, InterPro y KEGG con la descripción que les corresponde. Así, utilizando el *script* de importación a la base de datos, una vez se han introducido los códigos de las anotaciones, a cada uno de ellos se les añade su descripción, lo que permite que las personas entendamos que información almacenan esos códigos. Por ejemplo, para el código EC 6.3.1.2 se añade la descripción *Glutamine synthetase*, y para término de la *Gene Ontology* GO:0002213 su descripción será *defense response to insect*. En el caso de las rutas metabólicas KEGG, sus entradas en la base de datos se obtienen a partir de la enzimas que participan en las rutas, con un fichero de KEGG que relaciona las enzimas con las rutas en las que participan (`ec_map.txt`). Además, al anotar las bases de datos de pino se utilizó un fichero elaborado manualmente (en colaboración con D. Pacheco) con las rutas metabólicas que no son de plantas (`keggs_to_filter.txt`), de modo que estas se filtran y no se incluyen en las bases de datos. Cualquier administrador de la base de datos puede personalizar este fichero para filtrar rutas metabólicas que no pertenezcan a la especie en estudio. Este filtro suele ser necesario debido a que la anotación automática puede incorporar información incorrecta cuando una secuencia contiene una anotación incorrecta en los repositorios públicos de secuencias.

11.3.6. Modelo final de base de datos de transcriptómica mejorada

La estructura mejorada se aplicó en primer lugar para el desarrollo de una nueva base de datos del transcriptoma de *P. pinaster*, aunque luego ha servido de modelo de base de datos para transcriptómica, independiente del organismo de estudio. Con las mejoras técnicas introducidas se puede controlar el desarrollo de la web de varias bases de datos a la vez. La vista de la web de cada base de datos se personaliza con dos ficheros, uno de tipo CSS y otro en formato JSON. El fichero CSS (estilos para páginas web) contenido en la carpeta `stylesheets` de `public` (resaltado en naranja en la figura 11.8) sirve para personalizar el aspecto de las páginas. En el fichero JSON se indican los textos e imágenes específicos de la web, y se incluye en la carpeta `config` (resaltado en naranja en la figura 11.8). La información que se almacena es diferente para cada base de datos, pero las tablas relacionales utilizadas (figura 11.5) son las mismas. Como resultado se ahorra mucho tiempo al no tener que desarrollar desde cero ni las tablas ni las vistas de la web, lo que permite centrarse en obtener el mejor análisis posible del transcriptoma. En la figura 11.9 pueden compararse dos bases de datos distintas realizadas con una estructura y control de la web común pero con diferentes datos, figuras y textos, como se ha descrito en este trabajo. Uno de los ejemplos (figura 11.9) en los que se aplica este modelo de base de datos de transcriptómica es, SoleaDB (`http://www.scbi.uma.es/soleadb/home_page`), realizada en colaboración con el Instituto de Investigación y Formación Agraria y Pesquera (IFAPA) «El Toruño» de Cádiz, con secuencias del transcriptoma del lenguado (*Solea senegalensis*).

La web y la base de datos del transcriptoma se han construido en una estructura de 3 máquinas virtuales para portarlas con facilidad a distintos dominios y evitar los problemas de instalación e incompatibilidades. En una de las maquinas se controla la web, en otra la base de datos y en la última la ejecución de BLAST. De este modo se reparten las tareas y se evita que la web quede inaccesible a otros usuarios cuando uno está realizando consultas (D. Guerrero, datos no publicados).

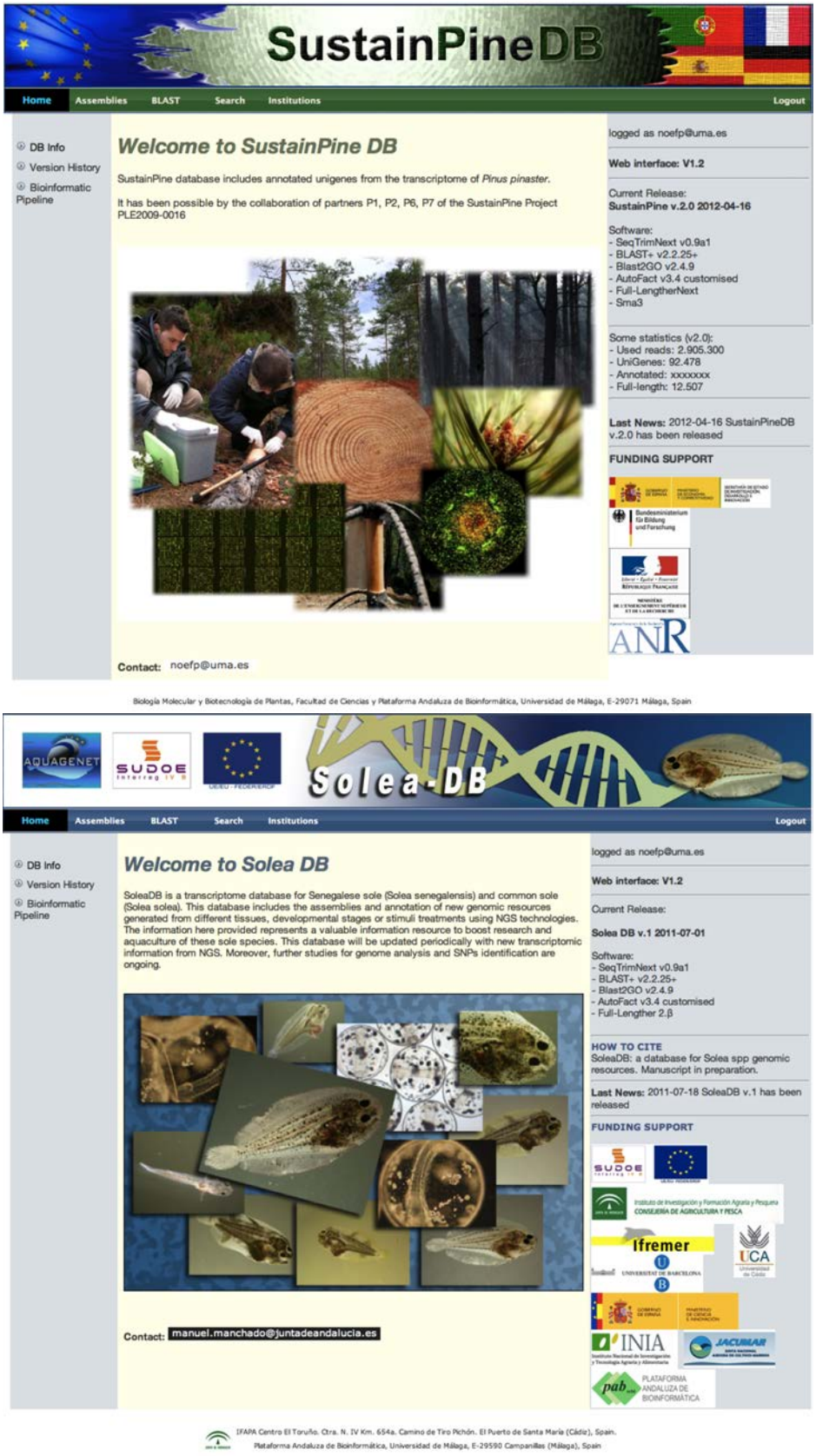


Figura 11.9: *Interfaz de dos bases de datos diferentes realizadas con el modelo de base de datos de transcriptómica propuesto. En la parte superior se muestra la de SustainPineDB y en la parte inferior la de SoleaDB.*

11.4. Nuevo transcriptoma de referencia

A continuación se describe el contenido de SPDB v1.2, una nueva versión mejorada del transcriptoma de pino. Esta base de datos sirve como base para futuras versiones del transcriptoma de pino, y sus secuencias pueden resultar útiles para caracterizar los genes del genoma de pino, especialmente cuando se completen los proyectos de secuenciación genómicos de pino planteados para los próximos años (para más información ver apartado 1.3), o incluso para experimentos de RNA-Seq.

11.4.1. Calidad de las nuevas librerías

La calidad de los datos de partida tras el preprocesamiento con seqTrimNext se muestra en la tabla 11.2. Llama la atención la enorme proporción de lecturas rechazadas de las librerías Biogeco 1 (45 %) y 2 (52 %). Gracias al análisis de SeqTrimNext se puso de manifiesto la existencia de errores durante la preparación de estas librerías en el laboratorio, al presentar un mayor número de secuencias contaminadas y contener numerosos artefactos (principalmente por concatenación de los adaptadores utilizados durante la amplificación del ADNc). Si no se hubieran eliminado todos estos artefactos con SeqTrimNext se hubiesen introducido una gran cantidad de errores en los unigenes formados, ya que habrían afectado gravemente al proceso de ensamblaje y habrían supuesto la formación de unigenes sin información biológica útil.

Cada librería se ensambló después por separado según el flujo propuesto en el apartado 10.4 y los unigenes resultantes se validaron con FullLengtherNext (Tabla 11.3). Ahora llama la atención los datos de UAGPF, la librería con mayor número de unigenes y mayor número de unigenes con ortólogos, pero donde el unigén de mayor longitud es más pequeño. Además, el número de unigenes completos de UAGPF, tanto de forma global como considerando sólo los únicos, es demasiado pequeño para la cantidad de lecturas útiles que contiene. La librería Biogeco 2 también contiene valores relativamente bajos de unigenes completos y de unigenes completos diferentes, pero en este caso se justifica porque el número de lecturas útiles que contiene es más pequeño (casi la tercera parte). UAGPF también contiene un elevado número de secuencias no codificantes en comparación con el resto de librerías, llegando a englobar a casi el 40 % de los unigenes. Estos datos ponen de manifiesto que la librería UAGPF esta más fragmentada que las demás, pues presenta casi la mitad (1,93 veces menos, 2,98 basándose en los porcentajes) de unigenes completos diferentes que Biogeco1, una librería con casi 20 000 unigenes menos que UAGPF.

11.4.2. El preprocesamiento altera el ensamblaje del transcriptoma

En este apartado se mostrarán las diferencias que hay entre las distintas versiones obtenidas del transcriptoma. Por un lado está EPDB (véase la tabla 11.4), que representa al transcriptoma incluido en EuroPineDB. También se cuenta con EPDB2, otra versión del transcriptoma en la que las mismas lecturas de EPDB (a excepción de 6899 secuencias del EMBL de *pinaster*) se preprocesaron y ensamblaron como se indica en el flujo de trabajo del apartado 10.4. Por otro lado, están las nuevas versiones del transcriptoma de *Pinus pinaster*, SPDB v1.1 y 1.2, ambas creadas a partir de las 4 librerías de 454 comentadas en el apartado anterior, siendo la versión de SeqTrimNext utilizada para el preprocesamiento, la única diferencia entre ellas. Las distintas versiones de SPDB también se ensamblaron siguiendo nuestro flujo de trabajo. En la tabla 11.4 se comparan los ensamblajes obtenidos de todas ellas en función de los resultados obtenidos con FullLengtherNext.

Las 2 versiones de EPDB (véase la tabla 11.4) son difíciles de comparar entre sí, puesto que se han obtenido por métodos distintos, y como se mostró en el apartado 11.3, no fueron los más adecuados en la primera versión de EPDB, ya que los unigenes obtenidos están sobrevalorados y son redundantes. Aún así se observa que la aplicación del flujo de trabajo en EPDB2 disminuye el número de unigenes obtenidos y el resto de valores de la tabla 11.4, a excepción de los unigenes completos con ortólogos diferentes. Estos datos apoyan que con el nuevo flujo se reduce la redundancia y se consigue completar la secuencia de más genes diferentes. Además, porcentualmente, EPDB2 mejora el número de unigenes >500pb, ortólogos diferentes y unigenes completos. El número de unigenes con y sin ortólogo, y del resto de categorías incluidas dentro de los unigenes sin ortólogo mantienen la misma proporción entre ambas versiones del transcriptoma. Por tanto se puede pensar que EPDB2 es un mejor reflejo del transcriptoma que EPDB.

Al comparar el transcriptoma de EPDB2 con las 2 versiones de SPDB se observa claramente que se obtienen valores más altos en las dos versiones de SPDB para todas los datos mostrados en la tabla 11.4. Esto se debe a que el transcriptoma de SPDB

Tabla 11.2: *Comparativa de las librerías de 454 preprocesadas por SeqTrimNext v2.0.35*

	EPDB2[1]		Biogeco1		Biogeco2		UAGPF	
	#seqs	%	#seqs	%	#seqs	%	#seqs	%
Lecturas brutas	913 786	100	1 571 741	100	768 224	100	990 405	100
Lecturas útiles	717 128	78,48	674 405	42,91	266 944	34,75	742 795	75,00
Baja complejidad	109 377	11,97	185 913	11,83	96 898	12,61	161 821	16,34
Lecturas rechazadas[2]	87 281	9,55	711 423	45,26	404 382	52,64	85 789	8,66
Repetidas	55 071	63,10	173 519	24,39	60 277	14,91	55 852	65,10
Contaminadas	3087	3,54	126 082	17,72	21 554	5,33	3409	3,97
Otros motivos[3]	29 123	33,37	411 822	57,89	322 551	79,76	26 528	30,92
Baja calidad[4]	-	41,68	-	41,72	-	40,92	-	37,97
Longitud media (pb)	236	-	250	-	203	-	263	-
Moda (pb)	209	-	**50**	-	**50**	-	309	-

[1] En este caso, EPDB2 contiene únicamente las lecturas de 454 de la librería incluida en EuroPineDB.
[2] Los porcentajes dentro de este bloque se calcularon considerando el número total de secuencias rechazadas como el 100 %.
[3] Aquí se incluyen las que tienen inserto demasiado corto, demasiadas indeterminaciones y artefactos de secuenciación.
[4] En este caso no se rechazan las lecturas, sino que se recortan los segmentos que contienen nucleótidos de baja calidad. El porcentaje refiere el 100 % al número de nucleótidos que hay en las lecturas brutas.

Tabla 11.3: *Verificación con* Full-LengtherNext *de las librerías de 454 que forman parte del nuevo transcriptoma de pino (SPDB v1.2)*

	EPDB2[1]		Biogeco1		Biogeco2		UAGPF	
	#seqs	%	#seqs	%	#seqs	%	#seqs	%
Unigenes	39 222	100	36 614	100	20 362	100	56 521	100
Unigenes >500pb	18 939	48,29	20 299	55,44	8096	39,76	19 139	33,86
Unigén más largo (pb)	4568	-	4212	-	3851	-	3490	-
Con ortólogo [2]	23 585	67,52	24 992	68,07	14 331	70,38	29 278	51,80
Nº ortólogos diferentes	12 760	54,10	12 948	51,81	8766	61,17	13 262	45,30
Nº unigenes completos	5402	22,90	6129	24,52	2414	16,84	2979	10,17
Completos diferentes	4492	19,05	4680	18,73	2032	14,18	2426	8,29
Sin ortólogo [2]	11 343	32,47	11 692	32,93	6031	29,62	27 243	48,20
Posible codificante	2908	25,64	2990	25,57	1221	20,25	4960	18,21
No codificante	8427	74,29	8679	74,23	4795	79,50	22 272	81,75

[1] En este caso, EPDB2 contiene únicamente las lecturas de 454 de la librería incluida en EuroPineDB.
[2] Los porcentajes dentro de este bloque se calcularon considerando el número de secuencias con/sin ortólogo como el 100 %.

se ha construido con 3 librerías más (Biogeco 1, Biogeco 2 y UAGPF) que suponen aproximadamente el triple de lecturas útiles. Sin embargo, porcentualmente se observa un incremento en el número de secuencias sin ortólogo y no codificantes, una buena parte de las cuales serán secuencias artefactuales (véase el artículo de Full-LengtherNext, en el apartado 10.2).

En cuanto a las dos versiones de SPDB se puede comprobar cómo una serie de mejoras llevadas a cabo en el algoritmo de SeqTrimNext entre la versión 2.0.27b (utilizada en SPDB v1.1) y la versión 2.0.35 (SPDB v1.2) pueden producir grandes cambios en los ensamblajes realizados, lo que se verá que se traduce en una versión mejor del transcriptoma. Así, en SPDB v1.2 se redujo el número de unigenes en 1542, de 91 086 en la versión 1.1 a 89 544 en la 1.2, por lo que se puede pensar que en esta versión del transcriptoma se ha reducido la sobrevaloración del transcriptoma. Como la única variación entre SPDB v1.1 y 1.2 es el preprocesamiento se puede suponer que esta reducción se ha conseguido eliminado más artefactos. No deja de ser sorprendente a priori que retirar 66 302 lecturas más en SPDBv1.2 (un 1,55 % de diferencia en las lecturas útiles, ambas con un conjunto de partida de 4 275 112 lecturas) consiga que el mismo ensamblaje aumente el número de unigenes mayores de 500 pb (38 287 frente a 36 932), el número de unigenes con ortólogo en UniProt (47 034 frente a 46 615), y el número de unigenes con un gen completo (11 985 frente 10 356). Además, el número de unigenes completos diferentes pasa a ser de 7365 en la versión 1.2 frente a 6718 en la 1.1, lo que indica que gracias a eliminar más

Tabla 11.4: *Diferentes versiones del transcriptoma de* Pinus pinaster*, desde el primero que se obtuvo con EuroPineDB hasta el de SPDB V1.2, en los que se han empleado distintas versiones de SeqTrimNext. La comparación se fundamenta en el análisis con* Full-LengtherNext.

	EPDB		EPDB2[1]		SPDB v1.1		SPDB v1.2	
	#seqs	%	#seqs	%	#seqs	%	#seqs	%
SeqTrimNext	v0.111[2]		v2.0.35		v2.0.b27		v2.0.35	
Lecturas brutas	951 641	100	944 742	100	4 275 112	100	4 275 112	100
Lecturas útiles	877 523	92,21	748 084	79,18	2 498 530	58,44	2 432 228	56,89
Nº Unigenes	55 332	100	41 246	100	91 086	100	**89 544**	100
Unigenes >500pb	24 937	45,07	20 115	48,77	36 932	40,55	**38 287**	42,76
Con ortólogo[3]	38 115	68,88	28 314	68,65	46 615	51,18	**47 034**	52,53
Nº ort. diferentes	14 696	38,56	13 452	47,51	16 999	36,47	16 879	35,89
Nº unig. completos	6675	17,51	6164	21,77	10 356	22,22	**11 985**	25,48
Completos difer.	4427	11,61	**4640**	16,39	6718	14,41	**7365**	15,66
Sin ortólogo[3]	17 217	31,12	12 932	31,35	44 471	48,82	42 510	47,47
P. codificantes[4]	4430	25,67	3305	25,55	7841	17,63	7829	18,42
No codificantes	12 777	74,21	9619	74,38	36 578	82,25	**34 629**	81,46

[1] EPDB2 no contiene las secuencias de EMBL de *P.pinaster* incluidas en EPDB.
[2] Las lecturas de EPDB estaban preprocesadas con SeqTrim y ensambladas con MIRA3.
[3] Los porcentajes dentro de este bloque se calcularon considerando el número de secuencias con/sin ortólogo como el 100 %.
[4] Unigenes que posiblemente codifiquen un gen a pesar de no mostrar similitud con un ortólogo en el análisis realizado por Full-LengtherNext.

lecturas erróneas durante en el preprocesamiento se ha conseguido completar al menos la secuencia de 647 genes más. Todos estos valores de mejora se resaltan en negrita en la tabla 11.4.

Se ha descrito que es habitual que en los transcriptomas de especies no modelo aparezcan entre un 40-60 % de genes desconocidos, que se atribuyen a genes específicos de la especie o del linaje, aunque desafortunadamente muchos de ellos son realmente fallos de ensamblaje u otros artefactos [171]. Al comparar el número de secuencias no codificantes en los distintos ensamblajes (tabla 11.4) se ve que EPDB2 tiene menos secuencias no codificantes que EPDB, y que SPDB v1.2 también tiene menos que la v1.1. Esto implica los cambios en el flujo de trabajo mejoraron el transcriptoma (EPDB frente a EPDB2) y que el preprocesamiento mejorado consiguió eliminar 1542 unigenes artefactuales en SPDB v1.1 y reducir el número de unigenes no codificantes en SPDB v1.2, lo que sugiere que éste es un transcriptoma más fiable, con menos secuencias sin información biológica.

La comparación entre EPDB y EPDB2 demuestra una vez más que el flujo de trabajo propuesto en este trabajo proporciona mejores transcriptomas. Y si se comparan las dos versiones de SPDB, en las que el único cambio reside en la versión de SeqTrimNext utilizada, se deduce que el preprocesamiento afecta drásticamente al proceso de ensamblaje y que es un proceso esencial que puede significar la diferencia entre tener un mal ensamblaje y conseguir un ensamblaje fiable. Por consiguiente, queda demostrado que un buen preprocesamiento mejora el resultado final del ensamblaje, tal y como se propuso en el apartado 10.1.2, y en conclusión, se proponen los unigenes de SPDB V1.2 como un nuevo transcriptoma de referencia de pino.

11.4.3. Consecuencias del origen heterocigótico del transcriptoma de pino

Una de las primeras preguntas que cabe plantearse una vez se obtiene una versión del transcriptoma es, ¿cuántos genes hay?. Teniendo en cuenta la bibliografía, el número de genes en otras plantas nunca superan los 40 000 [211] y en otros organismos como el ser humano y *Arabidopsis*, está entre 20 000 y 25 000 [52, 208]. Por lo tanto, cabe esperar que el pino no sea una excepción y contenga un número de genes cercano al mencionado o, como mucho, cercano a 40 000. Sin embargo, observando los datos de SPDB v1.2 recogidos en la tabla 11.4, parece que hubiera 89 544 unigenes. Esto indica claramente que el transcriptoma está sobrevalorado, ya que a pesar de haber realizado un reensamblaje con CAP3 para disminuir la redundancia de MIRA3 (figura 10.1), el número de unigenes obtenido es más del doble del número de genes esperados. Esta situación puede deberse a dos factores: la alta variabilidad de las muestras de origen, y a la acumulación de errores en el laboratorio, el preprocesamiento y ensamblaje,

de lo que da pistas la presencia de 34 629 unigenes no codificantes de la tabla 11.4. Vamos a centrarnos en la variabilidad de las muestras de origen.

La secuenciación de transcriptomas se realiza habitualmente a partir de muestras de diversos tejidos y condiciones para tratar de obtener el mayor número de genes diferentes, es decir, tratar de obtener muestras en las que aparezcan los genes comunes y los que solo se expresen en un tejido o condiciones concretas. Además, las duplicaciones de genes y las variaciones debidas a los alelos pueden complicar el ensamblaje de los transcritos, haciendo difícil de distinguir entre los errores de secuenciación y las variaciones debidas a la heterocigosis y a la paralogía [213]. Por estos motivos, cuando se ensamblan las lecturas, un mismo gen queda representado por varios unigenes. En nuestro caso, las muestras de origen utilizadas para generar el transcriptoma eran muy heterocigóticas, ya que provenían de una mezcla de muestras de tejidos y condiciones ambientales, tomadas en individuos de variedades naturales de poblaciones de diferentes países, lo que aumenta la variabilidad con respecto a la caracterización de transcriptomas en otras especies de plantas, en las que la variedad en estudio ha sido domesticada en laboratorios o cultivos. Otras fuentes de variabilidad pueden ser los ayustes alternativos que producen más isoformas de un mismo gen y los pseudogenes que aún se expresan, pudiendo compartir similitud con otros genes del transcriptoma, lo que hace aún más complejo el proceso de ensamblaje.

Si se compara el número de unigenes que muestran similitud con un ortólogo en UniProt (47 034) en SPDBv1.2, con el número de ortólogos diferentes (16 879) se puede interpretar que cada gen está representado por casi 3 unigenes diferentes (2,8 veces de media), lo que pone de manifiesto la alta variabilidad de las secuencias, a pesar del reensamblaje realizado con CAP3, un modo descrito para corregir los unigenes redundantes [238, 128, 108, 213, 148].

Una explicación de por qué cada unigén está representado varia veces puede ser que muchos de ellos estén fragmentados en unigenes no solapantes o con regiones solapantes por debajo de las requeridas por los ensambladores. Por ejemplo, fijándose en los ensamblajes realizados para cada una de las librerías que forman SPDB 1.2 (tabla 11.3), los unigenes con ortólogo de la librería UAGPF están representados 2,21 veces de promedio, algo más que el resto de librerías (1,85 en EPDB2, 1,93 en Biogeco 1, y 1,63 en Biogeco 2), debido a que esta librería está más fragmentada que las demás, como se mostró en el apartado 11.4.1.

11.4.4. Posibles genes específicos de *P. pinaster*

Gran parte de los genomas eucariotas secuenciados hasta la fecha, incluidos los de plantas, han mostrado un conjunto de genes específicos de la especie o del linaje, que no muestran similitud con ortólogos de especies cercanas. El estudio funcional de estos genes es muy útil para encontrar características que diferencian a grupos muy cercanos desde un punto de vista evolutivo. Desafortunadamente, parte de las secuencias que conforman estos genes pueden ser el resultado de errores de ensamblaje [171].

Se da por hecho que los unigenes del grupo `Con ortólogo`, del transcriptoma de pino representado en SPDB v1.2 corresponden a secuencias que contienen genes con certeza, de los cuales, con los datos de la tabla 11.4 se puede afirmar que al menos hay 16 879 genes diferentes, lo que indica que todavía hay que acercarse más a los 20 000-25 000 genes que podría tener el pino, si se considera válida su similitud con *Arabidopsis* [73, 208]. Puede que el número de genes con ortólogos sea realmente superior a los 16 879 genes, dado que es posible que distintos parálogos de pino puedan estar dando similitud con el mismo ortólogo en otra especie. Pero donde cabría esperar que se encuentren una buena cantidad de genes nuevos y específicos de pino sería entre los 42 510 unigenes que no tienen ortólogo en la tabla 11.4. En principio podría pensarse que los 7829 unigenes posiblemente codificantes sean genes nuevos (aunque no todos ellos diferentes), que junto con los 16 879 con ortólogo arrojarían un total de 24 708 unigenes, lo que se acercaría más al número total de genes que se esperan en pino.

Ya se ha demostrado previamente (apartado 10.2) que en el grupo de unigenes clasificado como `sin ortólogo` por Full-LengtherNext se encontrarán tanto los verdaderos genes específicos de especie como los artefactos de ensamblaje y secuenciación. Para tratar de vislumbrar cuántos unigenes de cada tipo hay entre ellos en SPDB v1.2 se analizaron con AutoFact para tratar de recuperar aquellos unigenes con algún parecido en alguna base de datos, incluso aunque esta sea de nucleótidos (tabla 11.5).

Unigenes (posiblemente) codificantes. El análisis realizado con AutoFact confirma que aproximadamente la mitad de los unigenes clasificados por Full-LengtherNext como codificantes o posiblemente codificantes (52,82 % y 45,45 % respectivamente), codifican un gen o parte de él (tabla 11.5). La gran mayoría (> 97 %) se confirma porque hay alguna secuencia de EST similar a ellos. Puesto que la base de datos de nucleótidos `est_others`

Tabla 11.5: *Anotación con AutoFact sobre las secuencias sin ortólogo de SPDB v1.2, distribuidas según la clasificación de* Full-LengtherNext *en unigenes codificantes, posiblemente codificantes, y desconocidos.*

	Sin ortólogo					
	Codificantes		P. codificantes		Desconocidos	
	#seqs	%	#seqs	%	#seqs	%
Numero de unigenes[1]	3726	4,16	4103	4,58	34 629	38,67
Anotadas con AutoFact[2]	1968	52,82	1865	45,45	7929	22,90
con est_others	1923	97,71	1855	99,46	7920	99,89
con PFam	16	0,81	2	0,11	1	0,01
con UniRef90	29	1,47	7	0,38	7	0,09
con nr	0	0,00	1	0,05	1	0,01
AutoFact sin anotar	1758	47,18	2238	54,55	26 700	77,10

[1] Se consideran como el 100 % los 89 544 unigenes del ensamblaje.
[2] AutoFact se ejecuto utilizando un valor de $E = 10^{-25}$ para utilizar el mismo rigor que se utilizó para la verificación con Full-LengtherNext. Los porcentajes mostrados de las secuencias anotadas y sin anotar por AutoFact están calculados con Codificantes (3726), Posibles codificantes (4103) y No codificantes (34 629) como referencia en sus columnas respectivas. Los porcentajes de las categorías incluidas dentro de este bloque se calcularon utilizando los valores de AutoFact anotadas (2829, 2851 y 15 777, respectivamente) como el 100 %.

incluye EST de todos los organismos disponibles en GenBank salvo humano y ratón, es lógico que se encuentren secuencias que aún no están en las bases de datos de proteínas; el problema es que estas secuencias no están revisadas por un experto y aunque la mayoría de ellas contengan un transcrito real, también las hay que pueden ser simplemente artefactos (véase el artículo de GENote, apartado 10.3.3). La presencia de algunos unigenes con similitud en Pfam, UniRef o NR se explica porque estas bases de datos contienen secuencias que no son de plantas e incluyen también trozos de genes o proteínas, mientras que la base de datos utilizada por Full-LengtherNext contiene menos proteínas porque sólo se utilizan las de plantas, y sólo las que estaban completas. Curiosamente, aparecen más anotaciones de proteínas en los unigenes codificantes que en los posibles codificantes, lo que apoya el hecho de que los primeros son más fiables que los segundos. Los que quedaron sin anotación con Autofact no pueden ser descartados como codificantes ya que (1) fueron considerados genes codificantes por TestCode [75], y este análisis tiene un porcentaje de acierto > 90 % (véase el artículo de Full-LengtherNext, apartado 10.2), y (2) las similitudes en nucleótidos son más difíciles de detectar que con aminoácidos.

Unigenes desconocidos. En el caso de los unigenes calificados como `desconocidos` por Full-LengtherNext, Autofact solo consigue anotar el 22,90 %, de nuevo esencialmente con EST (> 99 %). Entre las 7920 que Autofact ha conseguido anotar, 6217 (78,50 %) son con EST de pino, 1517 (19,15 %) con otras coníferas (de las que 1499 son de picea), 4 (0,05 %) con otras gimnospermas, 81 (1,02 %) con otras plantas y 101 (1,28 %) con otros organismos. Por tanto, el 98,72 % de los EST proceden de plantas, lo que indica que la mayoría de ellos seguramente son genes que existen en las coníferas, y que probablemente sean específicos de ellas. Por otro lado, los 101 unigenes que muestran similitud con EST de otros organismos suponen tan solo un 0,11 % de los 89 544 unigenes de SPDB v1.2, y algunos de ellos, los procedentes de animales (88 unigenes), probablemente puedan mostrar algún dominio en común, y otros, como los procedentes de microorganismos (13 unigenes) serán restos de contaminantes. De las secuencia anotada con bases de datos de proteínas, la de `PFam` no tiene información de la especie de origen, la de `nr` es una secuencia parcial de una proteína de pino (AAT88046), y de las 7 de `UniRef90`, 1 es un fragmento de proteína de pino (Q8L8J5), 3 de animales y 3 de bacterias, de las cuales 2 codifican transposasas. Por lo que se confirma que es lógico que ninguna de estas secuencias esté entre las secuencias completas de plantas que utiliza Full-LengtherNext en sus análisis. No obstante, sí que está claro que hay un porcentaje de secuencias `Desconocidas` superior al 20 % que podrían ser codificantes, mientras que el aproximadamente 80 % restante no. Estas secuencias restantes probablemente serán los artefactos del ensamblaje y se podrán descartar. Pero surge una nueva posibilidad que verificar: ¿las secuencias anotadas con Autofact eran menores de 200 pb o mayores? Porque hay que recordar que los unigenes de < 200 pb no se analizaban con TestCode (véase el artículo de Full-LengtherNext en el apartado 10.2).

Secuencias desconocidas en función del tamaño. En algunos casos se ha observado que los

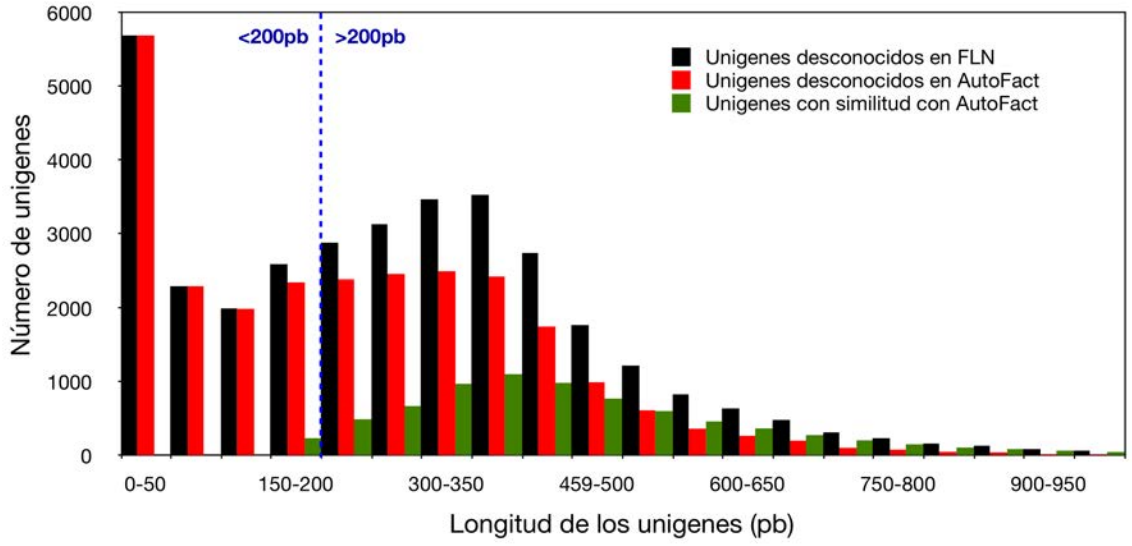


Figura 11.10: *Distribución de la longitud de los unigenes marcados como* `desconocidos` *según* FULL-LENGTHERNEXT *que luego se analizaron con AutoFact. Cada valor de barra negra se divide en un valor de barra roja y otra verde, por lo que cada pareja de barras roja y verde suma el valor de su correspondiente negra. Se ha marcado la línea divisoria de 200 pb de los unigenes.*

contaminantes suelen estar presentes en secuencias consideradas como genes específicos de la especie o el linaje, especialmente en secuencias cortas, que no muestran ninguna secuencia ortóloga en las bases de datos [9]. Como las secuencias sin ortólogo menores de 200 pb no se han analizado con TestCode (apartado 10.2), estas pasarán en la clasificación de FULL-LENGTHERNEXT al grupo de unigenes `Desconocidos`, aunque puedan puedan contener un fragmento de un gen. En esta fracción de desconocidos hay 12 515 (36,14 %) unigenes menores de 200 pb, por lo que se decidió analizar la relación entre el tamaño del unigén y la anotación con Autofact.

En la figura 11.10) se representan en barras negras la frecuencia de longitudes de los 34 629 unigenes desconocidos de SPDB v1.2 (tabla 11.5), en barra rojas, los 26 700 unigenes sin anotar por AutoFact (tabla 11.5), y en barras verdes los 7929 unigenes Anotados por AutoFact (tabla 11.5). Autofact solo anotó 237 unigenes menores de 200 pb, lo que supone únicamente el 0,68 % de los unigenes desconocidos y el 1,9 % de los unigenes desconocidos menores de 200 pb. Con estos datos se puede concluir que realmente las secuencias que FULL-LENGTHERNEXT marca como `desconocidas` que tienen menos de 200 pb realmente no presentan información útil para el experimento, y que solo en las mayores de 200 pb se pueden rescatar secuencias útiles.

También se observa que a medida que aumenta la longitud del unigén, hay más similitud con EST en proporción al número de desconocidos (compárense las barras rojas y verdes mayores de 200 pb). De hecho, en los unigenes de más de 550 pb hay más casos con anotación con Autofact que sin anotación, lo que hace pensar que solo los unigenes de más de 500 pb pueden presentar secuencias útiles para el experimento. Es lógico pensar que las secuencias largas posiblemente tengan más probabilidad de contener un gen, pero igualmente, a mayor longitud de secuencia, mayor probabilidad de que una parte de ella se parezca a algún trozo de EST de las bases de datos, lo que en modo alguno se puede considerar significativo. Por ejemplo, es posible que parte de estas secuencias contengan, únicamente, regiones UTR ensambladas y restos sin información biológica, que al igual que aparecen en nuestros experimentos, pueden aparecer en los de otros investigadores y quedar posteriormente incluidos en las bases de datos de EST. Además se ha descrito que las poblaciones de pequeños ARN de las gimnospermas, incluidas varias coníferas, muestran estructuras distintas a las observadas en angiospermas [59], por lo que es de esperar, que dentro de la fracción de genes desconocidos se encuentren más ARNnc específicos de pino que no pueden ser detectados con las bases de datos de ARNnc actuales, ya que en estas bases de datos, la mayor parte de la información de plantas, proviene de angiospermas.

Fiabilidad de las secuencias desconocidas. Para conocer si los unigenes calificados como `Desconocidos` tenían secuencias reales o eran ensamblajes artefactuales se mapearon con Bowtie2

Tabla 11.6: *Mapeo de las lecturas útiles de SPDB v1.2 sobre los unigenes de su ensamblaje. El porcentaje de los unigenes y el de las lecturas mapeadas se calculó con respecto al total de los unigenes y lecturas útiles respectivamente.*

	Con ortólogo		Sin ortólogo					
			Codificantes		P. codificantes		Desconocidos	
	#seqs	%	#seqs	%	#seqs	%	#seqs	%
Numero de unigenes	47 034	52,53	3726	4,16	4103	4,58	34 629	38,67
Lecturas mapeadas	2 009 051	83,67	82 282	3,43	95 201	3,96	223 536	9,31
Unigenes sin lecturas[1]	416	0,88	43	1,15	50	1,22	7179	20,73
Lecturas/unigén[2]	42,71	1,57	22,08	0,81	23,22	0,85	6,45	0,24
lecturas/unigén y tamaño medio[2]	0,048	1,10	0,042	0,96	0,047	1,08	0,022	0,50
Longitud media (pb)	884		522		491		294	

[1] Unigenes en los que no quedó alineada ninguna lectura durante el mapeo con Bowtie2.
[2] En la casilla del porcentaje se muestra la proporción de lecturas que alinean por unigén. Un valor de 1 indica que la proporción coincide con la media del ensamblaje.

las 2 432 228 lecturas útiles sobre los 89 544 unigenes del transcriptoma de SPDB v1.2, separándolos en los grupos que proporciona FULL-LENGTHERNEXT y que se muestran en la tabla 11.6. En esta tabla se observa que el 83,67 % de las lecturas mapeaban en los genes con ortólogo, a pesar de que éstos no son más que el 52 % de los unigenes del ensamblaje, mientras que solo el 16,3 % mapeaba en los que no tenían ortólogo (el 47,41 % de los unigenes). Resulta muy llamativo el grupo de unigenes `desconocidos`, que aportan el 38,67 % de los unigenes del ensamblaje y sobre los que tan solo mapean el 9,31 % de las lecturas. Al centrarse en los unigenes en los que no quedó ninguna lectura alineada durante el mapeo (figura 11.6), los unigenes con ortólogo, codificantes y posiblemente codificantes muestran aproximadamente un 1 % de unigenes sin lecturas mapeadas. Hay que recordar que en el mapeo con Bowtie2 se requería que la lectura hiciera un match perfecto de extremo a extremo, por lo que es probable que se trate de unigenes que tienen muchas variaciones sobre la secuencia original (en una inspección manual de algunos se observó que solían corresponder a unigenes en los que FULL-LENGTHERNEXT tenía que resolver un gran número de cambios de fase para recuperar la secuencia de la proteína). En cambio, el conjunto de los desconocidos tiene un 20,73 % de unigenes en los que no se ha alineado ninguna lectura, de los cuales, el 97,97 % corresponde a secuencias que no obtuvieron anotación con AutoFact.

Para que los datos de la tabla 11.6 sean más comparables se calculó en cada categoría, el número de lecturas que hay por unigén, teniendo en cuenta que, si el reparto de las lecturas en los unigenes fuera equitativa, en cada unigén mapearían una media de 2 432 228/89 544 = 27 lecturas. En cambio, puede verse que en los genes con ortólogo, la proporción sube a casi el doble (42,7), mientras que se mantiene cercana a este valor medio en los `codificantes` y `posibles codificantes`. Sin embargo, la proporción es de apenas 6,45 lecturas por unigén en los `desconocidos`. Además, al analizar por separado los unigenes desconocidos que fueron anotados con AutoFact y los que no lo fueron, se obtiene que los unigenes con anotación suben la proporción de mapeo a 16,49 lecturas/unigén y los unigenes sin anotación se quedan con 3,47 lecturas/unigén. De nuevo se confirma la correlación entre mapeo y anotación.

Puesto que la longitud media de los unigenes con ortólogo es mayor que la de los unigenes codificantes y posiblemente codificantes, y ésta, a su vez mayor que la de los desconocidos, podría pensarse que parte de la diferencia se deba a que unas categorías tienen más lecturas que otras porque hay más secuencia para mapear. Por tanto se normaliza el mapeo teniendo en cuenta el tamaño medio de los unigenes de cada grupo y el resultado indica que en los genes con ortólogo, codificantes y posibles codificantes presentan un índice muy similar de secuencias por nucleótido, mientras que cae a la mitad en el grupo de desconocidos (figura 11.6). En estos resultados sigue apareciendo que los unigenes calificados como `desconocidos` por FULL-LENGTHERNEXT contienen poca o ninguna información biológica, y que podrían ser eliminados de los análisis posteriores.

En conclusión se vuelve a confirmar que la mayoría de los unigenes desconocidos probablemente carecen de significado biológico y no tienen un gran respaldo entre las secuencias que se obtuvieron. Por tanto, dentro del conjunto de genes desconocidos, nuestra propuesta es conservar únicamente los 7929 unigenes que obtuvieron una anotación con AutoFact (tabla 11.5), y que se ha probado que contienen

más lecturas por unigén, descartando los 26 700 que no muestran similitud ni siquiera frente a EST, no siguen un patrón de genes codificantes según TestCode (véase el artículo de FULL-LENGTHERNEXT, apartado 10.2), y apenas alinean lecturas en su secuencia. A la vista de los resultados obtenidos, en un futuro próximo se añadirá la evaluación de las secuencias, mediante mapeo, en el flujo de trabajo propuesto en el apartado 10.4, para descartar un mayor número de secuencias sin información biológica antes de proceder a su anotación en profundidad.

11.4.5. Los genes del transcriptoma de pino

Con los datos obtenidos en los apartados anteriores se puede asegurar que el transcriptoma de pino contiene al menos 16 879 genes diferentes representados en 47 034 unigenes (tabla 11.7), obtenidos por similitud con bases de datos de proteínas (tabla 11.4) y apoyados por la mayoría de las lecturas útiles del ensamblaje (tabla 11.6).

A estos unigenes hay que añadir los 7829 que no muestran similitud con ninguna proteína ortóloga pero que podrían contener un gen al recibir la calificación de codificantes y posiblemente codificantes según FULL-LENGTHERNEXT (tabla 11.4). Incluirlos viene apoyado por el hecho de que (1) se comprobó que al menos el 50 % de ellos se podía anotar con AutoFact (tabla 11.5), y (2) la proporción de lecturas que mapean sobre ellos por unigén y nucleótido es la misma que para los que tienen ortólogos, y ampliamente superior a la de los unigenes desconocidos (tabla 11.6). Estos 7829 unigenes serían los candidatos a contener genes específicos de gimnospermas, coníferas o pino, y que no están incluidos en las bases de datos de proteínas actuales. No es descartable que entre ellos pueda haber una fracción que realmente no sean codificantes. Como valoración del nivel de redundancia de los genes sin similitud se analizaron los identificadores de los 3778 unigenes con similitud con EST (tabla 11.5) y se comprobó que había 3330 identificadores diferentes, lo que dejaría el número de genes específicos de especie en 7381 (tabla 11.7). Aún así, seguramente quedará redundancia ya que en las bases datos de EST no se controla que la secuencia de un mismo gen este representada numerosas veces. También hay que contemplar la posibilidad de que haya EST que realmente procedan de una contaminación de ADN genómico (algo que se puso de manifiesto analizando los BAC con nuestro programa GENote 0.β1). Sólo se pudo conocer mejor el nivel de redundancia al comparar las secuencias con CD-HIT [138] a un 90 % de identidad (apartado 7.3.7), con el que se obtuvieron 7075 unigenes diferentes, aunque todavía podrían estar sobrevalorados, debido a que los genes estén fragmentados. Por lo tanto, los 16 879 unigenes seguros más los 7075 bastante probables suman 23 954 genes (tabla 11.7), lo que se aproximaría a los 25 000 genes esperados basándose en otras plantas como *Arabidopsis* [208]. De todas formas, se puede obtener una primera estimación del transcriptoma mínimo que se ha logrado identificar al sumar los 16 879 unigenes con ortólogo únicos más los 3330 codificantes anotados únicos con Autofact en bases de datos de EST más los 55 anotados con otras bases de datos, que en total son 20 264 unigenes (tabla 11.7).

En cuanto a la fracción de genes desconocidos, de los 34 629 unigenes que contiene SPDB v1.2 es posible descartar los 26 700 (61,2 %) que no se anotaron con Autofact (tabla 11.5), a pesar de que existen muchas EST de coníferas (apartado 1.3 de la introducción). Hay que tener en cuenta además, que los unigenes calificados como desconocidos, son de pequeño tamaño, 294 pb de media (tabla 11.6), de las cuales 12 515 son menores de 200 pb y solo 237 de ellos se pudieron anotar con Autofact (apartado 11.4.4); además, su proporción de lecturas por unigén es mucho menor a la del resto de unigenes del ensamblaje, y proporcionalmente, muestran un número mayor de unigenes en los que no mapea ninguna lectura (tabla 11.6). Por tanto, si se descartan los desconocidos sin anotaciones con AutoFact, el número máximo posible de unigenes del transcriptoma se reduce a 62 844 (tabla 11.7).

Para acercarse un poco más a la estimación real de genes que puede haber en el conjunto de 7929 unigenes desconocidos anotados con AutoFact se descartaron los EST con ortólogo repetidos y se redujo el número de secuencias basándose en su similitud con CD-HIT, lo que dejó 6480 unigenes, que junto con los 23 954 genes que tienen ortólogo o son posiblemente codificantes, permiten obtener una estimación del transcriptoma fiable cubierto. Esto arroja un total de 30 434 genes posibles en SPDB v1.2 (tabla 11.7). Con estos unigenes seguramente se cubrirá la mayoría de los genes del pino marítimo, aunque no se puede descartar que todavía falten algunos porque tengan niveles de expresión muy bajos o solo se expresen en condiciones muy especiales, o simplemente no se han secuenciado o se han podido perder en alguno de los pasos realizados desde la extracción de la muestra hasta la obtención de los unigenes en el ensamblaje.

Dado que no existe un genoma de referencia suficientemente cercano, la única estimación para saber si se tiene cubierta la mayoría del transcriptoma es basarse en las rutas metabólicas que se anotan con sus correspondientes códigos KEGG. Así se vio que

Tabla 11.7: *Resumen de los unigenes mínimos que representarán el transcriptoma de pino*

	Unigenes
Total	89 544
Con ortólogo	
todos	47 034
únicos	**16 879**
Codificantes	
todos	7829
sin repetidos	**7075**
anotados Autofact sin repetir	**3385**
Con ortólogo + codificantes	23 954
Transcriptoma mínimo[1]	**20 264**
Desconocidos	
Todos	34 629
anotados con Autofact	7929
anotados con Autofact únicos	**6480**
sin anotar	26 700
Total – desconocidos sin anotar	**62 844**
Transcriptoma máximo[2]	**30 434**

[1] Se calcula sumando a 16 879 los 3385 unigenes codificantes únicos anotados con AutoFact
[2] Se calcula sumando a 23 954 los 6480 unigenes únicos que se han anotado con AutoFact entre los desconocidos

estaban todas las enzimas de la ruta que conecta el metabolismo C1, la biosíntesis de monolignoles y la asimilación de amonio mediante el ciclo GS/GOGAT (apartado 11.2.1); se ha comprobado que en SPDB v1.2 siguen estando todas, pero además se recuperan todos los unigenes con la secuencia completa de la proteína, salvo la glutamato sintasa dependiente de NADH (NADH-GOGAT). Que ésta sea la única incompleta seguramente se debe su gran tamaño (2 208 aminoácidos). También se ha visto que están todos los genes del metabolismo del nitrógeno (figura 11.2). Por último, se ha mostrado que prácticamente todos los genes que tienen que ver con la fijación de carbono en organismos fotosintéticos (figura 11.6) están presentes en SPDB v1.2. Por tanto, no es aventurado pensar que los 30 434 unigenes propuestos como el mejor transcriptoma de *P. pinaster* obtenido hasta la fecha contengan la mayoría de los genes del pino.

Sin embargo, ésta no se puede considerar todavía una versión final del transcriptoma. A medida que se vayan incorporando más lecturas se completarán las secuencias de más genes, y se espera que vaya disminuyendo el número de unigenes desconocidos y se vaya confirmando el número de unigenes específicos de la especie. Pero no es esperable que se aumente significativamente el número total de unigenes. La futura incorporación de nuevas secuencias de genes y proteínas de coníferas o de gimnospermas en las bases de datos gracias a los proyectos genómicos de coníferas también ayudará a dar más fiabilidad a las futuras versiones del transcriptoma de pino marítimo, con mejores ensamblajes y anotaciones, y a descartar con aún mayor fiabilidad los unigenes sin información biológica.

Parte V

Conclusiones

Capítulo 12

1. Se ha comprobado que la combinación objetiva de dos o más métodos basados en distintos algoritmos para analizar datos (bien sean de micromatrices o de secuenciación) devuelve resultados más fiables que cuando se utiliza un único método.

2. MADE4-2C es el único programa de libre acceso para el análisis de micromatrices de dos colores con el que se puede obtener de forma automática (1) la calidad del experimento para asegurarse de que el resultado no se basa en variaciones técnicas, (2) el mejor método de normalización de los datos analizados, porque es uno de los factores que más influye en la detección de los GED, y (3) la identificación de los GED más fiables al basarse en la combinación de dos métodos diferentes. Gracias a esta información, en los experimentos con el transcriptoma de pino se han puesto de manifiesto errores experimentales, lo que ha permitido su corrección para obtener resultados más fiables.

3. Se han desarrollado los programas SeqTrim y SeqTrimNext para el preprocesamiento de secuencias de tipo Sanger y de NGS, respectivamente, y se ha demostrado que con ellos se obtienen transcriptomas mejor ensamblados y más fiables.

4. Los análisis con Full-LengtherNext no solamente sirven para detectar los unigenes que consiguen reconstruir proteínas completas, sino que son muy útiles para validar los ensamblajes *de novo* de cualquier especie que no tenga un genoma o transcriptoma de referencia; y además proporciona una anotación preliminar en muy poco tiempo.

5. Se ha aportado GENote v0.β1, una herramienta para ayudar a evaluar el ensamblaje de BAC con secuencias genómicas de organismos no modelo, y de gran utilidad para conocer los posibles genes que contengan basándose en la información transcriptómica.

6. Con las herramientas desarrolladas en este trabajo, junto con otras de dominio público, se ha propuesto un flujo de trabajo de preprocesamiento, ensamblaje, verificación y anotación diseñado para obtener el transcriptoma del pino, aunque igualmente útil para cualquier otro organismo eucariota.

7. EuroPineDB, la primera base de datos del transcriptoma de *Pinus pinaster*, contiene una representación bastante amplia del mismo, en la que están presentes casi todos los genes de las rutas metabólicas analizadas en este trabajo. Además, gracias a esta base de datos se pudo diseñar una nueva micromatriz de pino con 8208 sondas.

8. Se ofrece a la comunidad científica un modelo de base de datos para transcriptómica que es aplicable a cualquier organismo y que está diseñada para que, una vez que se tiene la información del transcriptoma, se pueda importar de un modo rápido y sencillo, sin preocuparse por crear su diseño ni su estructura. En esta nueva base de datos se ha incluido una nueva versión más completa del transcriptoma obtenida con el flujo propuesto en este trabajo.

9. En la versión más completa, el transcriptoma de *Pinus pinaster* se estima que contiene 16 879 genes bien caracterizados, además de 7075 genes nuevos posiblemente específicos de la especie, más otras 6480 secuencias que ya han aparecido en otros experimentos previos con EST. Esto arroja un transcriptoma mínimo de 20 264 y un transcriptoma máximo de 30 434 genes.

Parte VI

Bibliografía

Bibliografía

[1] F. Achard, United Nations Environment Programme, and GRID-Arendal. *Vital forest graphics*. UNEP, 2009.

[2] M.D. Adams, S.E. Celniker, R.A. Holt, C.A. Evans, J.D. Gocayne, P.G. Amanatides, S.E. Scherer, J.C. Venter, and et al. The genome sequence of drosophila melanogaster. *Science*, 287(5461):2185–2195, 2000.

[3] A Adomas, G Heller, A Olson, J Osborne, M Karlsson, J Nahalkova, L Van Zyl, R Sederoff, J Stenlid, R Finlay, and FQ Asiegbu. Comparative analysis of transcript abundance in pinus sylvestris after challenge with a saprotrophic, pathogenic or mutualistic fungus. *Tree Physiol*, 28(6):885–897, 2008.

[4] Jan Aerts and Andy Law. An introduction to scripting in ruby for biologists. *BMC Bioinformatics*, 10(1):221, 2009.

[5] A. Agah, M. Aghajan, F. Mashayekhi, S. Amini, R.W. Davis, J.D. Plummer, M. Ronaghi, and P.B. Griffin. A multi-enzyme model for pyrosequencing. *Nucleic Acids Res*, 32(21):e166, 2004.

[6] F. Al-Shahrour, L. Arbiza, H. Dopazo, J. Huerta-Cepas, P. Mínguez, D. Montaner, and J. Dopazo. From genes to functional classes in the study of biological systems. *BMC Bioinformatics*, 8:114, 2007.

[7] F. Alagna, N. DÁgostino, L. Torchia, M. Servili, R. Rao, M. Pietrella, G. Giuliano, M. Chiusano, L. Baldoni, and G. Perrotta. Comparative 454 pyrosequencing of transcripts from two olive genotypes during fruit development. *BMC Genomics*, 10(1):399, 2009.

[8] AA Alizadeh, MB Eisen, RE Davis, C Ma, IS Lossos, A Rosenwald, JC Boldrick, H Sabet, T Tran, and X Yu. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000.

[9] C. Alkan, S. Sajjadian, and E. E. Eichler. Limitations of next-generation genome sequence assembly. *Nat Methods*, 8(1):61–65, 2011.

[10] D.B. Allison, X. Cui, G.P. Page, and M. Sabripour. Microarray data analysis: from disarray to consolidation and consensus. *Nat Rev Genet*, 7(1):55–65, Jan 2006.

[11] Isabel Allona, Michelle Quinn, Elizabeth Shoop, Kristi Swope, Sheila St. Cyr, John Carlis, John Riedl, Ernest Retzel, Malcolm M. Campbell, Ronald Sederoff, and Ross W. Whetten. Analysis of xylem formation in pine by cdna sequencing. *Proceedings of the National Academy of Sciences*, 95(16):9693–9698, 1998.

[12] P. Alonso, M. Cortizo, F.R. Cantón, B. Fernández, A. Rodríguez, M.L. Centeno, F.M. Cánovas, and R.J. Ordás. Identification of genes differentially expressed during adventitious shoot induction in pinus pinea cotyledons by subtractive hybridization and quantitative pcr. *Tree Physiology*, 27(12):1721–1730, 2007.

[13] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

[14] F. Angeloni, C.A. Wagemaker, M.S. Jetten, H.J. Op den Camp, E.M. Janssen-Megens, K.J. Francoijs, H.G. Stunnenberg, and N.J. Ouborg. De novo transcriptome characterization and development of genomic tools for scabiosa columbaria l. using next-generation sequencing techniques. *Mol Ecol Resour.*, 11(4):662–674, July 2011.

[15] Rolf Apweiler, Amos Bairoch, Cathy H Wu, Winona C Barker, Brigitte Boeckmann, Serenella Ferro, Elisabeth Gasteiger, Hongzhan Huang, Rodrigo Lopez, Michele Magrane, Maria J Martin, Darren A Natale, Claire O'Donovan, Nicole Redaschi, and Lai-Su L Yeh. Uniprot: the universal protein knowledgebase. *Nucleic Acids Res*, 32(Database issue):D115–9, Jan 2004.

[16] The Arabidopsis and Genome Initiative. Analysis of the genome sequence of the flowering plant arabidopsis thaliana. *Nature*, 408(6814):796–815, 2000.

[17] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, G. Sherlock, and Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

[18] A. Barakat, D. DiLoreto, Y. Zhang, C. Smith, K. Baier, W. Powell, N. Wheeler, R. Sederoff, and J. Carlson. Comparison of the transcriptomes of american chestnut (castanea dentata) and chinese chestnut (castanea mollissima) in response to the chestnut blight infection. *BMC Plant Biology*, 9(1):51, 2009.

[19] William T. Barry, Andrew B. Nobel, and Fred A. Wright. Significance analysis of functional categories in gene expression studies: a structured permutation approach. *Bioinformatics*, 21(9):1943–1949.

[20] J. Batley and D. Edwards. Genome sequence data: management, storage, and visualization. *BioTechniques*, 46(333-336), 2009.

[21] Rocío Bautista, David Villalobos, Sara Díaz-Moreno, Francisco Cantón, Francisco Cánovas, and M. Claros. Toward a pinus pinaster bacterial artificial chromosome library. *Annals of Forest Science*, 64:855–864, 2007. 10.1051/forest:2007060.

[22] William R. Belknap, Yi Wang, Naxin Huo, Jiajie Wu, David R. Rockhold, Yong Q. Gu, and Ed Stover. Characterizing the citrus cultivar carrizo genome through 454 shotgun sequencing. *Genome*, 54(12):1005–1015, 2011.

[23] Y Benjamini and Y Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.

[24] Jeffrey L Bennetzen. Transposable elements, gene creation and genome rearrangement in flowering plants. *Current Opinion in Genetics Development*, 15(6):621–627, 2005.

[25] Jeffrey L. Bennetzen, Jianxin Ma, and Katrien M. Devos. Mechanisms of recent genome size variation in flowering plants. *Annals of Botany*, 95(1):127–132, 2005.

[26] M.S. Benson, D.A.and Boguski, D.J. Lipman, and J. Ostell. Genbank. *Nucleic Acids Research*, 25(1):1–6, 1997.

[27] J Martin Bland and Douglas G Altman. Multiple significance tests: the bonferroni method. *BMJ*, 310, 01 1995.

[28] B.M. Bolstad, R.A Irizarry, M. Åstrand, and T.P. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003.

[29] A. Bräutigam and U. Gowik. What can next generation sequencing do for you? next generation sequencing as a valuable tool in plant research. *Plant Biology*, 12(6):831–841, 2010.

[30] R. Breitling, P. Armengaud, A. Amtmann, and P. Herzyk. Rank products: a simple, yet powerful, new method to detect differentially regulated genes in replicated microarray experiments. *FEBS Lett*, 573(1-3):83–92, Aug 2004.

[31] Monika Brinker, Leonel van Zyl, Wenbin Liu, Deborah Craig, Ronald R. Sederoff, David H. Clapham, and Sara von Arnold. Microarray analyses of gene expression during adventitious root development in pinus contorta. *Plant Physiology*, 135(3):1526–1539, 2004.

[32] Carol J. Bult, Janan T. Eppig, James A. Kadin, Joel E. Richardson, Judith A. Blake, and the Mouse Genome Database Group. The mouse genome database (mgd): mouse biology and model systems. *Nucleic Acids Research*, 36(suppl 1):D724–D728, 2008.

[33] John Cairney, Li Zheng, Allison Cowels, Joseph Hsiao, Victoria Zismann, Jia Liu, Shu Ouyang, Francoise Thibaud-Nissen, John Hamilton, Kevin Childs, Gerald S. Pullman, Yiting Zhang, Thomas Oh, and C. Robin Buell. Expressed sequence tags from loblolly pine embryos reveal similarities with angiosperm embryogenesis. *Plant Molecular Biology*, 62:485–501, 2006. 10.1007/s11103-006-9035-9.

[34] C. Camacho, G. Coulouris, V. Avagyan, N. Ma, J. Papadopoulos, K. Bealer, and T.L. Madden. Blast+: architecture and applications. *BMC Bioinformatics*, 10:421, 2009.

[35] J. Canales, C. Avila, F.R. Cantón, D.P. Villalobos, S. Díaz-Moreno, D. Ariza, J.J. Molina-Rueda, R.M. Navarro-Cerrillo, M.G. Claros, and F.M. Cánovas. Gene expression profiling in the stem of young maritime pine trees: detection of ammonium stress-responsive genes in the apex. *Trees*, september 2011.

[36] J Canales, A Flores-Monterrosso, M Rueda-Lopez, C Avila, and FM Canovas. Identification of genes regulated by ammonium availability in the roots of maritime pine trees. *Amino Acids*, 39(4):991–1001, 2010.

[37] Brandi L. Cantarel, Ian Korf, Sofia M.C. Robb, Genis Parra, Eric Ross, Barry Moore, Carson Holt, Alejandro Sánchez Alvarado, and Mark Yandell. Maker: An easy-to-use annotation pipeline designed for emerging model organism genomes. *Genome Research*, 18(1):188–196, 2008.

[38] F. R. Cantón, G. Le Provost, V. García, A. Barré, J. M. Frigerio, J. Paiva, P. Fevereiro, C. Ávila, J. F. Mouret, A. De Daruvar, F. M. Canovas, and C. Plomion. *Transcriptome analysis of wood formation in maritime pine.*, pages 333–348. DFA-AFA Press, Vitoria-Gasteiz, 2004.

[39] F. R. Canton, M. F. Suarez, and F. M. Canovas. Molecular aspects of nitrogen mobilization and recycling in trees. *Photosynthesis Research*, 83(2):265–278, 2005.

[40] Mark J. Chaisson and Pavel A. Pevzner. Short read fragment assembly of bacterial genomes. *Genome Research*, 18(2):000, 2007.

[41] ET Chan, GT Quon, G Chua, T Babak, M Trochesset, RA Zirngibl, J Aubin, MJ Ratcliffe, A Wilde, M Brudno, QD Morris, and TR Hughes. Conservation of core gene expression in vertebrate tissues. *J Biol*, 8:33, 2009.

[42] C. Chang and E.M. Meyerowitz. Molecular cloning and dna sequence of the arabidopsis thaliana alcohol dehydrogenase gene. *Proc Natl Acad Sci USA*, 83:1408–1412, 1986.

[43] Foo Cheung, Brian Haas, Susanne Goldberg, Gregory May, Yongli Xiao, and Christopher Town. Sequencing medicago truncatula expressed sequenced tags using 454 life sciences technology. *BMC Genomics*, 7(1):272, 2006.

[44] B. Chevreux, T. Pfisterer, B. Drescher, A.J. Driesel, W.E.G. Müller, T. Wetter, and S. Suhai. Using the miraest assembler for reliable and automated mrna transcript assembly and snp detection in sequenced ests. *Genome Research*, 14:1147–1159, 2004.

[45] Monica Chiogna, Maria Sofia Massa, Davide Risso, and Chiara Romualdi. A comparison on effects of normalisations in the detection of differentially expressed genes. *BMC Bioinformatics*, 10(1):61, 2009.

[46] Peter J. A. Cock, Christopher J. Fields, Naohisa Goto, Michael L. Heuer, and Peter M. Rice. The sanger fastq file format for sequences with quality scores, and the solexa/illumina fastq variants. *Nucleic Acids Research*, 38(6):1767–1771, 2010.

[47] J. S. Coker and E. Davies. Identifying adaptor contamination when mining dna sequence data. *Biotechniques*, 37(2):194–198, 2004.

[48] L. Comai. The advantages and disadvantages of being polyploid. *Nat Rev Genet*, 6(11):836–46, 2005.

[49] A. Conesa, S. Gotz, J. M. Garcia-Gomez, J. Terol, M. Talon, and M. Robles. Blast2go: a universal tool for annotation, visualization and analysis in functional genomics research. *Bioinformatics*, 21(18):3674–3676, 2005.

[50] Ana Conesa, María José Nueda, Alberto Ferrer, and Manuel Talón. masigpro: a method to identify significantly differential expression profiles in time-course microarray experiments. *Bioinformatics*, 22(9):1096–1102, 2006.

[51] International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, February 2001.

[52] International Human Genome Sequencing Consortium. Finishing the euchromatic sequence of the human genome. *Nature*, 431:931–945, October 2004.

[53] The 1000 Genomes Project Consortium. A map of human genome variation from population-scale sequencing. *Nature*, 467:1061–1073, 2010.

[54] Liying Cui, P. Kerr Wall, James H. Leebens-Mack, Bruce G. Lindsay, Douglas E. Soltis, Jeff J. Doyle, Pamela S. Soltis, John E. Carlson, Kathiravetpilla Arumuganathan, Abdelali Barakat, Victor A. Albert, Hong Ma, and Claude W. dePamphilis. Widespread genome duplications throughout the history of flowering plants. *Genome Research*, 16(6):738–749, 2006.

[55] Stefanie De Bodt, Steven Maere, and Yves Van de Peer. Genome duplication and the origin of angiosperms. *Trends in Ecology & Evolution*, 20(11):591–597, 11 2005.

[56] M. de la Bastide and W. R. McCombie. Assembling genomic dna sequences with phrap. *Curr. Protoc. Bioinformatics*, Chapter 11:Unit 11.4, 2007.

[57] J. DeRisi, L. Penland, P. O. Brown, M.L. Bittner, P.S. Meltzer, M. Ray, Y. Chen, Y.A. Su, and J.M. Trent. Use of a cdna microarray to analyse gene expression patterns in human cancer. *Nat Genet*, 14(4):457–460, 12 1996.

[58] Sara Díaz-Moreno. *Identificación de genes determinantes de caracteres de la madera juvenil y madura mediante análisis transcriptómicos en pino.* 185 páginas. Dpto. Biología Molecular y Bioquímica., Facultad de Ciencias, Universidad de Málaga, Málaga., 2010.

[59] Elena V. Dolgosheina, Ryan D. Morin, Gozde Aksay, S. Cenk Sahinalp, Vincent Magrini, Elaine R. Mardis, Jim Mattsson, and Peter J. Unrau. Conifers have a unique small rna silencing signature. *RNA*, 14(8):1508–1515, 2008.

[60] Joaquín Dopazo. Functional interpretation of microarray experiments. *OMICS: A Journal of Integrative Biology.*, 10(3):398–410, 2006.

[61] R Drysdale and FlyBase Consortium. Flybase : a database for the drosophila research community. *Methods in molecular biology (Clifton, N.J.)*, 420, 2008.

[62] B.P. Durbin, J.S. Hardin, D.M. Hawkins, and D.M. Rocke. A variance-stabilizing transformation for gene-expression microarray data. *Bioinformatics*, 18(suppl 1):S105–S110, 2002.

[63] T. Durfee, R. Nelson, S. Baldwin, G. Plunkett, V. Burland, B. Mau, J.F. Petrosino, X. Qin, D.M. Muzny, M. Ayele, R.A. Gibbs, B. Csörgo, G. Pósfai, G.M. Weinstock, and F.R. Blattner. The complete genome sequence of escherichia coli dh10b: insights into the biology of a laboratory workhorse. *J Bacteriol*, 190(7):2597–606, Apr 2008.

[64] Jon Duvick, Ann Fu, Usha Muppirala, Mukul Sabharwal, Matthew D. Wilkerson, Carolyn J. Lawrence, Carol Lushbough, and Volker Brendel. Plantgdb: a resource for comparative plant genomics. *Nucleic Acids Research*, 36(suppl 1):D959–D965, 2008.

[65] AJ Eckert and BD Hall. Phylogeny, historical biogeography, and patterns of diversification for pinus (pinaceae): Phylogenetic tests of fossil-based hypotheses. *Mol Phylogenet Evol*, 40(1):166–182, 2006.

[66] Al Edwards and C. Thomas Caskey. Closure strategies for random dna sequencing. *Methods Comp. Methods Enzymol.*, 3:41–47, 1991.

[67] S.S. Epstein. *Uncultivated Microorganisms*. Microbiology Monographs. Springer, 2009.

[68] B. Ewing and P. Green. Base-calling of automated sequencer traces using phred. ii. error probabilities. *Genome Research*, 8(3):186–194, 1998.

[69] B. Ewing, L. Hillier, M. C. Wendl, and P. Green. Base-calling of automated sequencer traces using phred. i. accuracy assessment. *Genome Research*, 8(3):175–185, 1998.

[70] J. Falgueras, A.J. Lara, N. Fernández-Pozo, F.R. Cantón, G. Pérez-Trabado, and M.G. Claros. Seqtrim: a high-throughput pipeline for pre-processing any type of sequence read. *BMC Bioinformatics*, 11:38, 2010.

[71] FAO. State of the world's forests 2011. *FAO*, 2011.

[72] Paolo Fardin, Stefano Moretti, Barbara Biasotti, Annamaria Ricciardi, Stefano Bonassi, and Luigi Varesio. Normalization of low-density microarray using external spike-in controls: analysis of macrophage cell lines expression profile. *BMC Genomics*, 8(1):17, 2007.

[73] N. Fernández-Pozo, J. Canales, D. Guerrero-Fernández, D. P. Villalobos, S. M. Díaz-Moreno, R. Bautista, A. Flores-Monterroso, M.A. Guevara, P. Perdiguero, C. Collada, M.T. Cervera, A. Soto, R. Ordás, F. R. Cantón, C. Avila, F.M. Cánovas, and M.G. Claros. Europinedb: a high-coverage web database for maritime pine transcriptome. *BMC Genomics*, 12(366), july 2011.

[74] Catherine Feuillet, Jan E. Leach, Jane Rogers, Patrick S. Schnable, and Kellye Eversole. Crop genome sequencing: lessons and rationales. *Trends in plant science*, 16(2):77–88, 02 2011.

[75] J. W. Fickett. Recognition of protein coding regions in dna-sequences. *Nucleic Acids Research*, 10(17):5303–5318, 1982.

[76] Heike Fiegler, Philippa Carr, Eleanor J. Douglas, Deborah C. Burford, Sarah Hunt, James Smith, David Vetrie, Patricia Gorman, Ian P.M. Tomlinson, and Nigel P. Carter. Dna microarrays for comparative genomic hybridization based on dop-pcr amplification of bac and pac clones. *Genes, Chromosomes and Cancer*, 36(4):361–374, 2003.

[77] B. Flannigan and J.D. Miller. *Health implications of fungi in indoor environments - an overview. In Health Implications of Fungi in Indoor Environments*. Elsevier, Amsterdam, 1993.

[78] S. Franssen, R. Shrestha, A. Brautigam, E. Bornberg-Bauer, and A. Weber. Comprehensive transcriptome analysis of the highly complex pisum sativum genome using next generation sequencing. *BMC Genomics*, 12(1):227, 2011.

[79] M.R. García-Gil. Evolutionary aspects of functional and pseudogene members of the phytochrome gene family in scots pine. *J Mol Evol*, 67(2):222–32, Aug 2008.

[80] Brandon S. Gaut and Jeffrey Ross-Ibarra. Selection on major components of angiosperm genomes. *Science*, 320(5875):484–486, 2008.

[81] K. Community of Scientists Genome. Genome 10k: a proposal to obtain whole-genome sequence for 10,000 vertebrate species. *J Hered*, 100(6):659–674, 2009.

[82] Robert C Gentleman, Vincent J. Carey, Douglas M. Bates, et al. Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, 5:R80, 2004.

[83] R. A. George. Stackpack clustering system. *Briefings in Bioinformatics*, 2:394–396, 2001.

[84] Jeremy Goecks, Anton Nekrutenko, James Taylor, and The Galaxy Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology*, 11(8):R86, 2010.

[85] TR Golub, DK Slonim, P Tamayo, C Huard, M Gaasenbeek, JP Mesirov, H Coller, ML Loh, JR Downing, and MA Caligiuri. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[86] David M. Goodstein, Shengqiang Shu, Russell Howson, Rochak Neupane, Richard D. Hayes, Joni Fazo, Therese Mitros, William Dirks, Uffe Hellsten, Nicholas Putnam, and Daniel S. Rokhsar. Phytozome: a comparative platform for green plant genomics. *Nucleic Acids Research*, 40(D1):D1178–D1186, 2012.

[87] Naohisa Goto, Pjotr Prins, Mitsuteru Nakao, Raoul Bonnal, Jan Aerts, and Toshiaki Katayama. Bioruby: bioinformatics software for the ruby programming language. *Bioinformatics*, 26(20):2617–2619, 2010.

[88] Eric D. Green. Strategies for the systematic sequencing of complex genomes. *Nat Rev Genet*, 2(8):573–583, 08 2001.

[89] R. E. Green, J. Krause, A. W. Briggs, T. Maricic, U. Stenzel, M. Kircher, N. Patterson, H. Li, M. H. Zhai, W.and Fritz, N. F. Hansen, E. Y. Durand, A. Malaspinas, J. D. Jensen, T. Marques-Bonet, C. Alkan, K. Prüfer, M. Meyer, H. A. Burbano, J. M. Good, R. Schultz, M. Lachmann, D. Reich, S. Pääbo, et al. A draft sequence of the neandertal genome. *Science*, 328(5979):710–722, 2010.

[90] Andrew T. Groover. What genes make a tree a tree? *Trends in plant science*, 10(5):210–214, 05 2005.

[91] D. Guerrero, R. Bautista, D.P. Villalobos, F.R. Cantón, and M.G. Claros. Alignminer: a web-based tool for detection of divergent regions in multiple sequence alignments of conserved sequences. *Algorithms for Molecular Biology*, 5(24), 2010.

[92] Osman Gulsen and Ahmet Ceylan. Elucidating polyploidization of bermudagrasses as assessed by organelle and nuclear dna markers. *OMICS J. Integr. Biol.*, 15(12):in press, 2011.

[93] S. Guo, Y. Zheng, J.G. Joung, S. Liu, Z. Zhang, O. Crasta, B. Sobral, Y. Xu, S. Huang, and Z. Fei. Transcriptome sequencing and comparative analysis of cucumber flowers with different sex types. *BMC Genomics*, 11(1):384, 2010.

[94] Brian Haab, Maitreya Dunham, and Patrick Brown. Protein microarrays for highly parallel detection and quantitation of specific proteins and antibodies in complex solutions. *Genome Biology*, 2(2):research0004.1–research0004.13, 2001. A previous version of this manuscript was made available before peer review at http://genomebiology.com/2000/1/6/preprint/0001/.

[95] Florian Hahne, Wolfgang Huber, Robert Gentleman, Seth Falcon, W. Huber, D. Scholtens, F. Hahne, and A. Heydebreck. *Bioconductor Case Studies*. Use R! Springer New York, 2008.

[96] Bettina Harr and Christian Schlötterer. Comparison of algorithms for the analysis of affymetrix microarray data as evaluated by co-expression of genes in known operons. *Nucleic Acids Research*, 34(2):e8, 2006.

[97] Gregory Heller, Aleksandra Adomas, Guosheng Li, Jason Osborne, Len van Zyl, Ron Sederoff, Roger Finlay, Jan Stenlid, and Frederick Asiegbu. Transcriptional analysis of pinus sylvestris roots challenged with the ectomycorrhizal fungus laccaria bicolor. *BMC Plant Biology*, 8(1):19, 2008.

[98] Reinhard Hoffmann, Thomas Seidl, and Martin Dugas. Profound effect of normalization on detection of differentially expressed genes in oligonucleotide microarray data analysis. *Genome Biology*, 3(7):research0033.1–research0033.11, 2002.

[99] Jason A. Holliday, Steven G. Ralph, Richard White, Jörg Bohlmann, and Sally N. Aitken. Global monitoring of autumn gene expression within and among phenotypically divergent populations of sitka spruce (picea sitchensis). *New Phytologist*, 178(1):103–122, 2008.

[100] Sture Holm. A simple sequentially rejective multiple test procedure. *Scand J Statist*, 6:65–70, 1979.

[101] Carson Holt and Mark Yandell. Maker2: an annotation pipeline and genome-database management tool for second-generation genome projects. *BMC Bioinformatics*, 12(1):491, 2011.

[102] Fangxin Hong and Rainer Breitling. A comparison of meta-analysis methods for detecting differentially expressed genes in microarray experiments. *Bioinformatics*, 24(3):374–382, 2008.

[103] Eleanor Howe, Kristina Holton, Sarita Nair, Daniel Schlauch, Raktim Sinha, and John Quackenbush. Mev: Multiexperiment viewer. In Michael F. Ochs, John T. Casagrande, and Ramana V. Davuluri, editors, *Biomedical Informatics for Cancer Research*, pages 267–277. Springer US, 2010.

[104] E. Huala, A. Dickerman, M. Garcia-Hernandez, D. Weems, L. Reiser, F. LaFond, D. Hanley, D. Kiphart, J. Zhuang, W. Huang, L. Mueller, D. Bhattacharyya, D. Bhaya, B. Sobral, B. Beavis, C. Somerville, and S.Y. Rhee. The arabidopsis information resource (tair): A comprehensive database and web-based information retrieval, analysis, and visualization system for a model plant. *Nucleic Acids Research*, 29(1):102–105, 2001.

[105] X. Huang and A. Madan. Cap3: A dna sequence assembly program. *Genome Research*, 9:868–877, 1999.

[106] T. Hubbard, D. Barker, E. Birney, G. Cameron, Y. Chen, L. Clark, T. Cox, J. Cuff, V. Curwen, T. Down, R. Durbin, E. Eyras, J. Gilbert, M. Hammond, L. Huminiecki, A. Kasprzyk, H. Lehvaslaiho, P. Lijnzaad, C. Melsopp, E. Mongin, R. Pettett, M. Pocock, S. Potter, A. Rust, E. Schmidt, S. Searle, G. Slater, J. Smith, W. Spooner, A. Stabenau, J. Stalker, E. Stupka, A. Ureta-Vidal, I. Vastrik, and M. Clamp. The ensembl genome database project. *Nucleic Acids Research*, 30(1):38–41, 2002.

[107] S. Hunter, P. Jones, A. Mitchell, R. Apweiler, Teresa K. Attwood, A. Bateman, T. Bernard, D. Binns, P. Bork, S. Burge, E. de Castro, P. Coggill, M. Corbett, U. Das, L. Daugherty, L. Duquenne, Robert D. Finn, M. Fraser, J. Gough, D. Haft, N. Hulo, D. Kahn, E. Kelly, I. Letunic, D. Lonsdale, R. Lopez, M. Madera, J. Maslen, C. McAnulla, J. McDowall, C. McMenamin, H. Mi, P. Mutowo-Muellenet, N. Mulder, D. Natale, C. Orengo, S. Pesseat, M. Punta, Antony F. Quinn, C. Rivoire, A. Sangrador-Vegas, Jeremy D. Selengut, Christian J. A. Sigrist, M. Scheremetjew, J. Tate, M. Thimmajanarthanan, Paul D. Thomas, Cathy H. Wu, C. Yeats, and S. Yong. Interpro in 2011: new developments in the family and domain prediction database. *Nucleic Acids Research*, 40(D1):D306–D312, 2012.

[108] Massimo Iorizzo, Douglas Senalik, Dariusz Grzebelus, Megan Bowman, Pablo Cavagnaro, Marta Matvienko, Hamid Ashrafi, Allen Van Deynze, and Philipp Simon. De novo assembly and characterization of the carrot transcriptome reveals novel genes, new markers, and genetic diversity. *BMC Genomics*, 12(1):389, 2011.

[109] Rafael A. Irizarry, Bridget Hobbs, Francois Collin, Yasmin D. BeazerBarclay, Kristen J. Antonellis, Uwe Scherf, and Terence P. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, 2003.

[110] Y. Ishida, H. Saito, S. Ohta, Y. Hiei, T. Komari, and T. Kumashiro. High efficiency transformation of maize (zea mays l.) mediated by agrobacterium tumefaciens. *Nature Biotechnology*, 14:745–750, June 1996.

[111] S. Jackson, S. Rounsley, and M. Purugganan. Comparative sequencing of plant genomes: choices to make. *Plant Cell*, 18(5):1100–1104, 2006.

[112] Marten Jager, Claus-Eric Ott, Johannes Grunhagen, Jochen Hecht, Hanna Schell, Stefan Mundlos, Georg Duda, Peter Robinson, and Jasmin Lienau. Composite transcriptome assembly of rna-seq data in a sheep model for delayed bone healing. *BMC Genomics*, 12(1):158, 2011.

[113] Mukesh Jain. A next-generation approach to the characterization of a non-model plant transcriptome. *CURRENT SCIENCE*, 101(11):1435–1439, 2011.

[114] Manikandan Jayapal and Alirio J Melendez. Dna microarray technology for target identification and validation. *Clinical and Experimental Pharmacology and Physiology*, 33(5-6):496–503, 2006.

[115] T Kai, D Williams, and AC Spradling. The expression profile of purified drosophila germline stem cells. *Dev Biol*, 283:486–502, 2005.

[116] M. Kanehisa, S. Goto, Y. Sato, M. Furumichi, and M. Tanabe. Kegg for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Research*, 40(D1):D109–D114, 2012.

[117] C. Kanz, P. Aldebert, N. Althorpe, W. Baker, A. Baldwin, K. Bates, P. Browne, A. van den Broek, M. Castro, G. Cochrane, and et al. The embl nucleotide sequence database. *Nucleic Acids Research*, 33:D29–D33, 2005.

[118] Ilene Karsch-Mizrachi, Yasukazu Nakamura, and Guy Cochrane. The international nucleotide sequence database collaboration. *Nucleic Acids Research*, 40(D1):D33–D37, 2012.

[119] C. Kendziorski, R. A. Irizarry, K.-S. Chen, J. D. Haag, and M. N. Gould. On the utility of pooling biological samples in microarray experiments. *Proceedings of the National Academy of Sciences of the United States of America*, 102(12):4252–4257, 2005.

[120] Purvesh Khatri and Sorin Drăghici. Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, 21(18):3587–3595, 2005.

[121] Matias Kirst, Arthur F. Johnson, Christie Baucom, Erin Ulrich, Kristy Hubbard, Rod Staggs, Charles Paule, Ernest Retzel, Ross Whetten, and Ronald Sederoff. Apparent homology of expressed genes from wood-forming tissues of loblolly pine (pinus taeda l.) with arabidopsis thaliana. *Proceedings of the National Academy of Sciences*, 100(12):7383–7388, 2003.

[122] Dries Knapen, Lucia Vergauwen, Kris Laukens, and Ronny Blust. Best practices for hybridization design in two-colour microarray analysis. *Trends in Biotechnology*, 27(7):406 – 414, 2009.

[123] Yuichi Kodama, Martin Shumway, and Rasko Leinonen. The sequence read archive: explosive growth of sequencing data. *Nucleic Acids Research*, 40(D1):D54–D56, 2012.

[124] R. Kolpakov, G. Bana, and G. Kucherov. mreps: efficient and flexible detection of tandem repeats in dna. *Nucleic Acids Research*, 31(13):3672–3678, april 2003.

[125] Juha Kononen, Lukas Bubendorf, Anne Kallionimeni, Maarit Barlund, Peter Schraml, Stephen Leighton, Joachim Torhorst, Michael J Mihatsch, Guido Sauter, and Olli-P. Kallionimeni. Tissue microarrays for high-throughput molecular profiling of tumor specimens. *Nat Med*, 4(7):844–847, 07 1998.

[126] L.B. Koski, M.W. Gray, B.F. Lang, and G. Burger. Autofact: an automatic functional annotation and classification tool. *BMC Bioinformatics*, 6:151, 2005.

[127] A. Kovach, JL. Wegrzyn, G. Parra, C. Holt, G.E. Bruening, C.A. Loopstra, J. Hartigan, M. Yandell, C.H. Langley, I. Korf, and DB. Neale. The pinus taeda genome is characterized by diverse and highly diverged repetitive sequences. *BMC Genomics*, 11(420), 2010.

[128] S. Kumar and M. Blaxter. Comparing de novo assemblers for 454 transcriptome data. *BMC Genomics*, 11(1):571, 2010.

[129] Ben Langmead, Cole Trapnell, Mihai Pop, and Steven Salzberg. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biology*, 10(3):R25, 2009.

[130] A. Lara, G. Pérez-Trabado, D. Villalobos, S. Díaz-Moreno, F. Cantón, and M. G. Claros. *A Web Tool to Discover Full-Length Sequences: Full-Lengther*, pages 361–368. Springer, 2007.

[131] Byungwook Lee, Taehui Hong, Sang Jin Byun, Taeha Woo, and Yoon Jeong Choi. Estpass: a web-based server for processing and annotating expressed sequence tag (est) sequences. *Nucleic Acids Research*, 35(suppl 2):W159–W162, 2007.

[132] Rasko Leinonen, Ruth Akhtar, Ewan Birney, Lawrence Bower, Ana Cerdeno-Tárraga, Ying Cheng, Iain Cleland, Nadeem Faruque, Neil Goodgame, Richard Gibson, Gemma Hoad, Mikyung Jang, Nima Pakseresht, Sheila Plaister, Rajesh Radhakrishnan, Kethi Reddy, Siamak Sobhany, Petra Ten Hoopen, Robert Vaughan, Vadim Zalunin, and Guy Cochrane. The european nucleotide archive. *Nucleic Acids Research*, 39(suppl 1):D28–D31, 2011.

[133] Rasko Leinonen, Hideaki Sugawara, and Martin Shumway. The sequence read archive. *Nucleic Acids Research*, 39(suppl 1):D19–D21, 2011.

[134] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, and 1000 Genome Project Data Processing Subgroup. The sequence alignment/map (sam) format and samtools. *Bioinformatics*, 2009.

[135] P. Li, E. Peatman, S. Wang, J. Feng, C. He, P. Baoprasertkul, P. Xu, H. Kucuktas, S. Nandi, B. Somridhivej, J. Serapion, M. Simmons, C. Turan, L. Liu, W. Muir, R. Dunham, Y. Brady, J. Grizzle, and Z. Liu. Towards the ictalurid catfish transcriptome: generation and analysis of 31,215 catfish ests. *BMC Genomics*, 8(1):177, 2007.

[136] R. Li, C. Yu, Y. Li, T. W. Lam, S. M. Yiu, K. Kristiansen, and J. Wang. Soap2: an improved ultrafast tool for short read alignment. *Bioinformatics*, 25(15):1966–1967, 2009.

[137] S. Li and H. H. Chou. Lucy 2: an interactive dna sequence quality trimming and vector removal tool. *Bioinformatics*, 20(16):2865–2866, 2004.

[138] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–9, Jul 2006.

[139] C. Liang, G. Wang, L. Liu, G. L. Ji, L. Fang, Y. S. Liu, K. Carter, J. S. Webb, and J. F. D. Dean. Coniferest: an integrated bioinformatics system for data reprocessing and mining of conifer expressed sequence tags (ests). *Bmc Genomics*, 8:134–144, 2007.

[140] F. Liang, I. Holt, G. Pertea, S. Karamycheva, S. L. Salzberg, and J. Quackenbush. An optimized protocol for analysis of est sequences. *Nucleic Acids Research*, 28(18):3657–3665, 2000.

[141] W. Walter Lorenz, Feng Sun, Chun Liang, Dmitri Kolychev, Haiming Wang, Xin Zhao, Marie-Michele Cordonnier-Pratt, Lee H. Pratt, and Jeffrey F. D. Dean. Water stress-responsive genes in loblolly pine (pinus taeda) roots identified by analyses of expressed sequence tag libraries. *Tree Physiology*, 26(1):1–16, 2006.

[142] WW Lorenz, Y-S Yu, M Simoes, and JFD Dean. Processing the loblolly pine ptgen2 cdna microarray. *J Vis Exp*, 25:1182, 2009.

[143] H.M. Ma, S. Schulze, S. Lee, M. Yang, E. Mirkov, J. Irvine, P. Moore, and A. Paterson. An est survey of the sugarcane transcriptome. *TAG Theoretical and Applied Genetics*, 108:851–863, 2004. 10.1007/s00122-003-1510-y.

[144] J.J. MacKay and J. F. D. Dean. *Transcriptomics.*, chapter Genetics, Genomics and Breeding of Conifers. Edenbridge Science Publishers and CRC Press, New York (in press)., 2011.

[145] Steven Maere, Stefanie De Bodt, Jeroen Raes, Tineke Casneuf, Marc Van Montagu, Martin Kuiper, and Yves Van de Peer. Modeling gene and genome duplications in eukaryotes. *Proceedings of the National Academy of Sciences of the United States of America*, 102(15):5454–5459, 2005.

[146] J.H. Malone and B. Oliver. Microarrays, deep sequencing and the true measure of the transcriptome. *BMC Biology*, 9(34), 2011.

[147] Marcel Margulies, Michael Egholm, William E. Altman, Said Attiya, Joel S. Bader, Lisa A. Bemben, Jan Berka, Michael S. Braverman, Yi-Ju Chen, Zhoutao Chen, Scott B. Dewell, Lei Du, Joseph M. Fierro, Xavier V. Gomes, Brian C. Godwin, Wen He, Scott Helgesen, Chun He Ho, Gerard P. Irzyk, Szilveszter C. Jando, Maria L. I. Alenquer, Thomas P. Jarvie, Kshama B. Jirage, Jong-Bum Kim, James R. Knight, Janna R. Lanza, John H. Leamon, Steven M. Lefkowitz, Ming Lei, Jing Li, Kenton L. Lohman, Hong Lu, Vinod B. Makhijani, Keith E. McDade, Michael P. McKenna, Eugene W.

Myers, Elizabeth Nickerson, John R. Nobile, Ramona Plant, Bernard P. Puc, Michael T. Ronan, George T. Roth, Gary J. Sarkis, Jan Fredrik Simons, John W. Simpson, Maithreyan Srinivasan, Karrie R. Tartaro, Alexander Tomasz, Kari A. Vogt, Greg A. Volkmer, Shally H. Wang, Yong Wang, Michael P. Weiner, Pengguang Yu, Richard F. Begley, and Jonathan M. Rothberg. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, 09 2005.

[148] Jeffrey Martin, Vincent Bruno, Zhide Fang, Xiandong Meng, Matthew Blow, Tao Zhang, Gavin Sherlock, Michael Snyder, and Zhong Wang. Rnnotator: an automated de novo transcriptome assembly pipeline from stranded rna-seq reads. *BMC Genomics*, 11(1):663, 2010.

[149] Victoria Martin-Requena, Antonio Munoz-Merida, M Gonzalo Claros, and Oswaldo Trelles. Prep+07: improvements of a user friendly tool to preprocess and analyse microarray data. *BMC Bioinformatics*, 10(1):16, 2009.

[150] A. Masoudi-Nejad, K. Tonomura, S. Kawashima, Y. Moriya, M. Suzuki, M. Itoh, M. Kanehisa, T. Endo, and S. Goto. Egassembler: online bioinformatics service for large-scale processing, clustering and assembling ests and genomic dna fragments. *Nucleic Acids Research*, 34:W459–W462, 2006.

[151] A.M. Maxam and W. Gilbert. A new method for sequencing dna. *PNAS*, 74(2):560–564, Feb 1977.

[152] Emma Meaburn, Lee M. Butcher, Leonard C. Schalkwyk, and Robert Plomin. Genotyping pooled dna using 100k snp microarrays: a step towards genomewide association scans. *Nucleic Acids Research*, 34(4):e28.

[153] Antonio M. Mérida, Gonzalo M. Claros, Oswaldo Trelles, and Antonio J. Perez. Sma3s: a 3 stages software for sequences make sense. In *XI Jornadas de Bioinformática*, 2012.

[154] Michael L. Metzker. Sequencing technologies [mdash] the next generation. *Nat Rev Genet*, 11(1):31–46, 01 2010.

[155] J. R. Miller, A. L. Delcher, S. Koren, E. Venter, B. P. Walenz, A. Brownley, J. Johnson, K. Li, C. Mobarry, and G. Sutton. Aggressive assembly of pyrosequencing reads with mates. *Bioinformatics*, 24(24):2818–2824, 2008.

[156] J. R. Miller, S. Koren, and G. Sutton. Assembly algorithms for next-generation sequencing data. *Genomics*, 95(6):315–327, 2010.

[157] David Montaner, Joaquín Tárraga, Jaime Huerta-Cepas, Jordi Burguet, Juan M. Vaquerizas, Lucía Conde, Pablo Minguez, Javier Vera, Sach Mukherjee, Joan Valls, Miguel A. G. Pujana, Eva Alloza, Javier Herrero, Fátima Al-Shahrour, and Joaquín Dopazo. Next station in microarray data analysis: Gepas. *Nucleic Acids Research*, 34(suppl 2):W486–W491, 2006.

[158] Vamsi K Mootha, Cecilia M Lindgren, Karl-Fredrik Eriksson, Aravind Subramanian, Smita Sihag, Joseph Lehar, Pere Puigserver, Emma Carlsson, Martin Ridderstrale, Esa Laurila, Nicholas Houstis, Mark J Daly, Nick Patterson, Jill P Mesirov, Todd R Golub, Pablo Tamayo, Bruce Spiegelman, Eric S Lander, Joel N Hirschhorn, David Altshuler, and Leif C Groop. Pgc-1[alpha]-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes. *Nat Genet*, 34(3):267–273, 07 2003.

[159] A.M. Morse, D.G. Peterson, M.N. Islam-Faridi, K.E. Smith, Z. Magbanua, S.A. Garcia, T.L. Kubisiak, H.V. Amerson, J.E. Carlson, C.D. Nelson, and J.M. Davis. Evolution of genome size and complexity in pinus. *PLoS One*, 4(2):e4332, 2009.

[160] Ali Mortazavi, Brian A Williams, Kenneth McCue, Lorian Schaeffer, and Barbara Wold. Mapping and quantifying mammalian transcriptomes by rna-seq. *Nat Meth*, 5(7):621–628, 07 2008.

[161] Marvin Mundry, Erich Bornberg-Bauer, Michael Sammeth, and Philine G. D. Feulner. Evaluating characteristics of de novo assembly software on 454 transcriptome data: A simulation approach. *PLoS ONE*, 7(2):e31410, 02 2012.

[162] Dougu Nam and Seon-Young Kim. Gene-set approach for expression pattern analysis. *Briefings in Bioinformatics*, 9(3):189–197, 2008.

[163] D.B. Neale and A. Kremer. Forest tree genomics: growing resources and applications. *Nat Rev Genet*, 12(2):111–22, Feb 2011.

[164] NobelPrize. The nobel prize in chemistry 1980.

[165] Ruben Nogales-Cadenas, Pedro Carmona-Saez, Miguel Vazquez, Cesar Vicente, Xiaoyuan Yang, Francisco Tirado, Jose María Carazo, and Alberto Pascual-Montano. Genecodis: interpreting gene lists through enrichment analysis and integration of diverse biological information. *Nucleic Acids Research*, 37(suppl 2):W317–W322, 2009.

[166] E. Novaes, D. Drost, W. Farmerie, G. Pappas, D. Grattapaglia, R. Sederoff, and M. Kirst. High-throughput gene and snp discovery in eucalyptus grandis, an uncharacterized genome. *BMC Genomics*, 9(1):312, 2008.

[167] Alicia Oshlack, Dianne Emslie, Lynn Corcoran, and Gordon Smyth. Normalization of boutique two-color microarrays with a high proportion of differentially expressed probes. *Genome Biology*, 8(1):R2, 2007.

[168] S. Ossowski, K. Schneeberger, R. M. Clark, C. Lanz, N. Warthmann, and D. Weigel. Sequencing of natural strains of arabidopsis thaliana with short reads. *Genome Research*, 18(12):2024–2033, 2008.

[169] David P. Villalobos, Sara Díaz-Moreno, El-Sayed S. Said, Rafael A. Cañas, Daniel Osuna, Sonia HE Van Kerckhoven, Rocío Bautista, M. Gonzalo Claros, Francisco M. Cánovas, and Francisco R. Cantón. Reprogramming of gene expression during compression wood formation in pine: coordinated modulation of s-adenosylmethionine, lignin and lignan related genes. *BMC Plant Biology*, In Press., 2012.

[170] JAP Paiva, PH Garnier-Gere, JC Rodrigues, A Alves, S Santos, J Graca, G Le Provost, G Chaumeil, D Da Silva-Perez, A Bosc, P Fevereiro, and C Plomion. Plasticity of maritime pine (pinus pinaster) wood-forming tissues during a growing season. *New Phytol*, 179(4):1080–1094, 2008.

[171] Andrew H Paterson, Michael Freeling, Haibao Tang, and Xiyin Wang. Insights from the comparison of plant genome sequences. *Annu Rev Plant Biol*, 61:349–72, 2010.

[172] Tucker A Patterson, Edward K Lobenhofer, Stephanie B Fulmer-Smentek, Patrick J Collins, Tzu-Ming Chu, Wenjun Bao, Hong Fang, Ernest S Kawasaki, Janet Hager, Irina R Tikhonova, Stephen J Walker, Liang Zhang, Patrick Hurban, Francoise de Longueville, James C Fuscoe, Weida Tong, Leming Shi, and Russell D Wolfinger. Performance comparison of one-color and two-color platforms within the microarray quality control (maqc) project. *Nat Biotech*, 24(9):1140–1150, 09 2006.

[173] Bernardo Peixoto, Ricardo Vencio, Camila Egidio, Luisa Mota-Vieira, Sergio Verjovski-Almeida, and Eduardo Reis. Evaluation of reference-based two-color methods for measurement of gene expression ratios using spotted cdna microarrays. *BMC Genomics*, 7(1):35, 2006.

[174] E. Pennisi. Keeping genome databases clean and up to date. *Science*, 286(5439):447–450, 1999.

[175] P.A. Pevzner, H. Tang, and M.S. Waterman. An eulerian path approach to dna fragment assembly. *PNAS*, 98(17):9748–9753, August 2001.

[176] Daniel Pinkel, Richard Segraves, Damir Sudar, Steven Clark, Ian Poole, David Kowbel, Colin Collins, Wen-Lin Kuo, Chira Chen, Ye Zhai, Shanaz H. Dairkee, Britt-marie Ljung, Joe W. Gray, and Donna G. Albertson. High resolution analysis of dna copy number variation using comparative genomic hybridization to microarrays. *Nat Genet*, 20(2):207–211, 10 1998.

[177] M. Pop and S.L. Salzberg. Bioinformatics challenges of new sequencing technology. *Trends in genetics : TIG*, 24(3):142–149, 03 2008.

[178] Consortium Potato Genome Sequencing, X. Xu, S. Pan, S. Cheng, B. Zhang, D. Mu, P. Ni, G. Zhang, S. Yang, R. Li, J. Wang, G. Orjeda, F. Guzman, M. Torres, R. Lozano, O. Ponce, D. Martinez, G. De la Cruz, S. K. Chakrabarti, V. U. Patil, K. G. Skryabin, B. B. Kuznetsov, N. V. Ravin, T. V. Kolganova, A. V. Beletsky, A. V. Mardanov, A. Di Genova, D. M. Bolser, D. M. Martin, G. Li,

Y. Yang, H. Kuang, Q. Hu, X. Xiong, G. J. Bishop, B. Sagredo, N. Mejia, W. Zagorski, R. Gromadka, J. Gawor, P. Szczesny, S. Huang, Z. Zhang, C. Liang, J. He, Y. Li, Y. He, J. Xu, Y. Zhang, B. Xie, Y. Du, D. Qu, M. Bonierbale, M. Ghislain, R. Herrera Mdel, G. Giuliano, M. Pietrella, G. Perrotta, P. Facella, K. O'Brien, S. E. Feingold, L. E. Barreiro, G. A. Massa, L. Diambra, B. R. Whitty, B. Vaillancourt, H. Lin, A. N. Massa, M. Geoffroy, S. Lundback, D. DellaPenna, C. R. Buell, S. K. Sharma, D. F. Marshall, R. Waugh, G. J. Bryan, M. Destefanis, I. Nagy, D. Milbourne, S. J. Thomson, M. Fiers, J. M. Jacobs, K. L. Nielsen, M. Sonderkaer, M. Iovene, G. A. Torres, J. Jiang, R. E. Veilleux, C. W. Bachem, J. de Boer, T. Borm, B. Kloosterman, H. van Eck, E. Datema, B. L. Hekkert, A. Goverse, R. C. van Ham, and R. G. Visser. Genome sequence and analysis of the tuber crop potato. *Nature*, 475(7355):189–195, 2011.

[179] Li-Xuan Qin, Kathleen F. Kerr, and Contributing Members of the Toxicogenomics Research Consortium. Empirical evaluation of data transformations and ranking statistics for microarray analysis. *Nucleic Acids Research*, 32(18):5471–5479, 2004.

[180] John Quackenbush, Feng Liang, Ingeborg Holt, Geo Pertea, and Jonathan Upton. The tigr gene indices: reconstruction and representation of expressed gene sequences. *Nucleic Acids Research*, 28(1):141–145, 2000.

[181] Steven Ralph, Hye Chun, Natalia Kolosova, Dawn Cooper, Claire Oddy, Carol Ritland, Robert Kirkpatrick, Richard Moore, Sarah Barber, Robert Holt, Steven Jones, Marco Marra, Carl Douglas, Kermit Ritland, and Jorg Bohlmann. A conifer genomics resource of 200,000 spruce (picea spp.) ests and 6,464 high-quality, sequence-finished full-length cdnas for sitka spruce (picea sitchensis). *BMC Genomics*, 9(1):484, 2008.

[182] Steven G. Ralph, Hesther Yueh, Michael Friedmann, Dana Aeschliman, Jeffrey A. Zeznik, Colleen C. Nelson, Yaron S. N. Butterfield, Robert Kirkpatrick, Jerry Liu, Steven J. M. Jones, Marco A. Marra, Carl J. Douglas, Kermit Ritland, and Jörg Bohlmann. Conifer defence against insects: microarray gene expression profiling of sitka spruce (picea sitchensis) induced by mechanical wounding or feeding by spruce budworms (choristoneura occidentalis) or white pine weevils (pissodes strobi) reveals large-scale changes of the host transcriptome. *Plant, Cell Environment*, 29(8):1545–1570, 2006.

[183] C.W. Riggins, Y. Peng, C.N. Stewart, and P.J. Tranel. Characterization of de novo transcriptome for waterhemp (amaranthus tuberculatus) using gs-flx 454 pyrosequencing and its application for studies of herbicide target-site genes. *Pest Management Science*, 66(10):1042–1052, 2010.

[184] M.E. Ritchie, J. Silver, A. Oshlack, M. Holmes, D. Diyagama, A. Holloway, and G.K. Smyth. A comparison of background correction methods for two-colour microarrays. *Bioinformatics*, 23(20):2700–7, Oct 2007.

[185] David M. Rocke and Blythe Durbin. Approximate variance-stabilizing transformations for gene-expression microarray data. *Bioinformatics*, 19(8):966–972, 2003.

[186] Gar W. Rothwell, Heather Sanders, Sarah E. Wyatt, and Simcha Lev-Yadun. A fossil record for growth regulation: The role of auxin in wood evolution1. *Annals of the Missouri Botanical Garden*, 95(1):121–134, 2012/03/27 2008.

[187] A Rotter, M Hren, S Baebler, A Blejec, and K Gruden. Finding differentially expressed genes in two-channel dna microarray datasets: how to increase reliability of data preprocessing. *Omics : a journal of integrative biology*, 12(3), 09 2008.

[188] Matthew Ruffalo, Thomas LaFramboise, and Mehmet Koyutürk. Comparative analysis of algorithms for next-generation sequencing read alignment. *Bioinformatics*, 2011.

[189] AI. Saeed, V. Sharov, J. White, J. Li, W. Liang, N. Bhagabati, J. Braisted, M. Klapa, T. Currier, M. Thiagarajan, A. Sturn, M. Snuffin, A. Rezantsev, D. Popov, A. Ryltsov, E. Kostukovich, I. Borisovsky, Z. Liu, A. Vinsavich, V. Trush, and J. Quackenbush. Tm4: a free, open-source system for microarray data management and analysis. *Biotechniques*, 34(2):374–378, 2003.

[190] F. Sanger and A.R. Coulson. A rapid method for determining sequences in dna by primed synthesis with dna polymerase. *Journal of Molecular Biology*, 94(3):441–448, 1975.

[191] F. Sanger, S. Nicklen, and A.R. Coulson. Dna sequencing with chain-terminating inhibitors. *Proc.Natl.Acad.Sci.*, 74(12):5463–5467, December 1977.

[192] T. E. Scheetz, N. Trivedi, C. A. Roberts, T. Kucaba, B. Berger, N. L. Robinson, C. L. Birkett, A. J. Gavin, B. O'Leary, T. A. Braun, M. F. Bonaldo, J. P. Robinson, V. C. Sheffield, M. B. Soares, and T. L. Casavant. Estprep: preprocessing cdna sequence reads. *Bioinformatics*, 19(11):1318–1324, 2003.

[193] M Schena, D Shalon, R Heller, A Chai, P O Brown, and R W Davis. Parallel human genome analysis: microarray-based expression monitoring of 1000 genes. *Proceedings of the National Academy of Sciences*, 93(20):10614–10619, 1996.

[194] Peer M. Schenk, Kemal Kazan, Iain Wilson, Jonathan P. Anderson, Todd Richmond, Shauna C. Somerville, and John M. Manners. Coordinated plant defense responses in arabidopsis revealed by microarray analysis. *Proceedings of the National Academy of Sciences*, 97(21):11655–11660, 2000.

[195] T. Schmidt and J.S. Heslop-Harrison. Genomes, genes and junk: the large- scale organization of plant chromosomes. *Trends Plant Sci.*, 3(5):195–198, 1998.

[196] R. Schmieder and R. Edwards. Fast identification and removal of sequence contamination from genomic and metagenomic datasets. *PLoS ONE*, 6(3):e17288, 03 2011.

[197] R. Schmieder, Y. Lim, F. Rohwer, and R. Edwards. Tagcleaner: Identification and removal of tag sequences from genomic and metagenomic datasets. *BMC Bioinformatics*, 11(1):341, 2010.

[198] A. Schulze and J. Downward. Navigating gene expression using microarrays — a technology review. *NATURE CELL BIOLOGY*, 3:190–195, August 2001.

[199] Motoaki Seki, Mari Narusaka, Junko Ishida, Tokihiko Nanjo, Miki Fujita, Youko Oono, Asako Kamiya, Maiko Nakajima, Akiko Enju, Tetsuya Sakurai, Masakazu Satou, Kenji Akiyama, Teruaki Taji, Kazuko Yamaguchi-Shinozaki, Piero Carninci, Jun Kawai, Yoshihide Hayashizaki, and Kazuo Shinozaki. Monitoring the expression profiles of 7000 arabidopsis genes under drought, cold and high-salinity stresses using a full-length cdna microarray. *The Plant Journal*, 31(3):279–292, 2002.

[200] D Shalon, S J Smith, and P O Brown. A dna microarray system for analyzing complex dna samples using two-color fluorescent probe hybridization. *Genome Research*, 6(7):639–645, 1996.

[201] J. Shendure and H. Ji. Next-generation dna sequencing. *nature biotechnology*, 26(10):1135–1145, Oct 2008.

[202] Kazuhiro Shibata, Masayoshi Itoh, Katsunori Aizawa, Sumiharu Nagaoka, Nobuya Sasaki, Piero Carninci, Hideaki Konno, Junichi Akiyama, Katsuo Nishi, Tokuji Kitsunai, Hideo Tashiro, Mari Itoh, Noriko Sumi, Yoshiyuki Ishii, Shin Nakamura, Makoto Hazama, Tsutomu Nishine, Akira Harada, Rintaro Yamamoto, Hiroyuki Matsumoto, Masami Muramatsu, Yorinao Inoue, Akira Kira, Yoshihide Hayashizaki, et al. Riken integrated sequence analysis (risa) system—384-format sequencing pipeline with 384 multicapillary sequencer. *Genome Research*, 10(11):1757–1771, 2000.

[203] A. F. Siegel, G. van den Engh, L. Hood, B. Trask, and J. C. Roach. Modeling the feasibility of whole genome shotgun sequencing using a pairwise end strategy. *Genomics*, 68(3):237–246, 2000.

[204] J. T. Simpson, K. Wong, S. D. Jackman, J. E. Schein, S. J. Jones, and I. Birol. Abyss: a parallel assembler for short read sequence data. *Genome Res*, 19(6):1117–1123, 2009.

[205] G. K. Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions using R and Bioconductor*, pages 397–420. Springer, New York,, 2005.

[206] Gordon K. Smyth, Matthew Ritchie, Natalie Thorne, and James Wettenhall. *Linear Models for Microarray Data User's Guide*. The Walter and Eliza Hall Institute of Medical Research Melbourne, Australia, April 2007.

[207] Gordon K Smyth and Terry Speed. Normalization of cdna microarray data. *Methods*, 31(4):265 – 273, 2003. <ce:title>Candidate Genes from DNA Array Screens: application to neuroscience</ce:title>.

[208] Chris Somerville and Jeff Dangl. Plant biology in 2010. *Science*, 290(5499):2077–2078, 2000.

[209] PT Spellman, G Sherlock, MQ Zhang, VR Iyer, K Anders, MB Eisen, PO Brown, D Botstein, and B Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell*, 9:3273–3297, 1998.

[210] Judy Sprague, Leyla Bayraktaroglu, Dave Clements, Tom Conlin, David Fashena, Ken Frazer, Melissa Haendel, Douglas G. Howe, Prita Mani, Sridhar Ramachandran, Kevin Schaper, Erik Segerdell, Peiran Song, Brock Sprunger, Sierra Taylor, Ceri E. Van Slyke, and Monte Westerfield. The zebrafish information network: the zebrafish model organism database. *Nucleic Acids Research*, 34(suppl 1):D581–D585.

[211] L. Sterck, S. Rombauts, K. Vandepoele, P. Rouzé, and Y. Van de Peer. How many genes are there in plants (... and why are they there)? *Curr Opin Plant Biol*, 10(2):199–203, Apr 2007.

[212] R.L. Strausberg, E.A. Feingold, R.D. Klausner, and F.S. Collins. The mammalian gene collection. *Science*, 286(5439):455–457, 1999.

[213] S. R Strickler, A. Bombarely, and L.A. Mueller. Designing a transcriptome next-generation sequencing project for a nonmodel plant species1. *Am J Bot*, Jan 2012.

[214] Aravind Subramanian, Pablo Tamayo, Vamsi K. Mootha, Sayan Mukherjee, Benjamin L. Ebert, Michael A. Gillette, Amanda Paulovich, Scott L. Pomeroy, Todd R. Golub, Eric S. Lander, and Jill P. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, 2005.

[215] S.M. Swarbreck, E.A. Lindquist, D.D. Ackerly, and G.L. Andersen. Analysis of leaf and root transcriptomes of soil-grown avena barbata plants. *Plant and Cell Physiology*, 52(2):317–332, 2011.

[216] M. A. Tanner, B. M. Goebel, M. A. Dojka, and N. R. Pace. Specific ribosomal dna sequences from diverse environmental settings correlate with experimental contaminants. *Applied and Enviromental Microbiology*, 64(8):3110–3113, Aug 1998.

[217] Yoshio Tateno and Takashi Gojobori. Dna data bank of japan in the age of information biology. *Nucleic Acids Research*, 25(1):14–17, 1997.

[218] Y. Taya, R. Devos, J. Tavernier, H. Cheroutre, G. Engler, and W. Fiers. Cloning and structure of the human immune interferon-gamma chromosomal gene. *EMBO J.*, 1:953–958, 1982.

[219] John Travis. Sweden bets on new lab to spruce up its bioscience future. *Science*, 328(5980):805, 2010.

[220] Leonel van Zyl, Sara von Arnold, Peter Bozhkov, Yongzhong Chen, Ulrika Egertsdotter, John MacKay, Ronald R. Sederoff, Jing Shen, Lyubov Zelena, and David H. Clapham. Heterologous array analysis in pinaceae: hybridization of pinus taeda cdna arrays with cdna from needles and embryogenic cultures of p. taeda, p. sylvestris or picea abies. *Comparative and Functional Genomics*, 3(4):306–318, 2002.

[221] R. K. Varshney, S. N. Nayak, G. D. May, and S. A. Jackson. Next-generation sequencing technologies and their implications for crop genetics and breeding. *Trends Biotechnol*, 27(9):522–530, 2009.

[222] Riccardo Velasco, Andrey Zharkikh, Jason Affourtit, Amit Dhingra, Alessandro Cestaro, Ananth Kalyanaraman, Paolo Fontana, Satish K Bhatnagar, Michela Troggio, Dmitry Pruss, Silvio Salvi, Massimo Pindo, Paolo Baldi, Sara Castelletti, Marina Cavaiuolo, Giuseppina Coppola, Fabrizio Costa, Valentina Cova, Antonio Dal Ri, Vadim Goremykin, Matteo Komjanc, Sara Longhi, Pierluigi Magnago, Giulia Malacarne, Mickael Malnoy, Diego Micheletti, Marco Moretto, Michele Perazzolli, Azeddine Si-Ammour, Silvia Vezzulli, Elena Zini, Glenn Eldredge, Lisa M Fitzgerald, Natalia Gutin,

Jerry Lanchbury, Teresita Macalma, Jeff T Mitchell, Michael Egholm, Yves Van de Peer, Francesco Salamini, Roberto Viola, et al. The genome of the domesticated apple (malus [times] domestica borkh.). *Nat Genet*, 42(10):833–839, 10 2010.

[223] Riccardo Velasco, Andrey Zharkikh, Michela Troggio, Dustin A. Cartwright, Alessandro Cestaro, Dmitry Pruss, Massimo Pindo, Lisa M. FitzGerald, Silvia Vezzulli, Julia Reid, Giulia Malacarne, Diana Iliev, Giuseppina Coppola, Bryan Wardell, Diego Micheletti, Teresita Macalma, Marco Facci, Jeff T. Mitchell, Michele Perazzolli, Glenn Eldredge, Pamela Gatto, Alexander Gutin, Yves Van de Peer, Francesco Salamini, Roberto Viola, et al. A high quality draft consensus sequence of the genome of a heterozygous grapevine variety. *PLoS ONE*, 2(12):e1326, 12 2007.

[224] J C Venter, M D Adams, E W Myers, P W Li, R J Mural, G G Sutton, H O Smith, M Yandell, C A Evans, R A Holt, J D Gocayne, P Amanatides, R M Ballew, D H Huson, J R Wortman, Q Zhang, C D Kodira, X H Zheng, L Chen, M Skupski, G Subramanian, P D Thomas, J Zhang, G L Gabor Miklos, C Nelson, and et al. The sequence of the human genome. *Science*, 291(5507):1304–51, Feb 2001.

[225] J. Craig Venter, Samuel Levy, Tim Stockwell, Karin Remington, and Aaron Halpern. Massive parallelism, randomness and genomic advances. *Nat Genet*, 33:219–227, 2003.

[226] David Pacheco Villalobos. *Aproximación genómica al estudio de la formación de la madera en los pinos*. 193 páginas. Dpto. Biología Molecular y Bioquímica., Facultad de Ciencias, Universidad de Málaga, Málaga., 2008.

[227] Jinrong Wan, Mark Dunning, and Andrew Bent. Probing plant-pathogen interactions and downstream defense signaling using dna microarrays. *Functional Integrative Genomics*, 2:259–273, 2002. 10.1007/s10142-002-0080-4.

[228] Yonghong Wang, Ze-Hong Miao, Yves Pommier, Ernest S. Kawasaki, and Audrey Player. Characterization of mismatch and high-signal intensity probes associated with affymetrix genechips. *Bioinformatics*, 23(16):2088–2095, 2007.

[229] JL Wegrzyn, JM Lee, BR Tearse, and DB Neale. Treegenes: A forest tree genome database. *International J Plant Genomes 2008*, 2008. Article ID 412875.

[230] Ross Whetten, Ying-Hsuan Sun, Yi Zhang, and Ron Sederoff. Functional genomics and cell wall biosynthesis in loblolly pine. *Plant Molecular Biology*, 47:275–291, 2001. 10.1023/A:1010652003395.

[231] Thomas Wicker, Edith Schlagenhauf, Andreas Graner, Timothy Close, Beat Keller, and Nils Stein. 454 sequencing put to the test using the complex genome of barley. *BMC Genomics*, 7(1):275, 2006.

[232] Melissa Wong, Charles Cannon, and Ratnam Wickneswari. Identification of lignin genes and regulatory sequences involved in secondary cell wall formation in acacia auriculiformis and acacia mangium via de novo transcriptome sequencing. *BMC Genomics*, 12(1):342, 2011.

[233] Huiling Xiong, Dapeng Zhang, Christopher Martyniuk, Vance Trudeau, and Xuhua Xia. Using generalized procrustes analysis (gpa) for normalization of cdna microarray data. *BMC Bioinformatics*, 9(1):25, 2008.

[234] Suk-Hwan Yang and Carol A. Loopstra. Seasonal variation in gene expression for loblolly pines (pinus taeda) from different geographical regions. *Tree Physiology*, 25(8):1063–1073, 2005.

[235] Suk-Hwan Yang, Leonel van Zyl, Eun-Gyu No, and Carol A. Loopstra. Microarray analysis of genes preferentially expressed in differentiating xylem of loblolly pine (pinus taeda). *Plant Science*, 166(5):1185 – 1195, 2004.

[236] Barry Zeeberg, Joseph Riss, David Kane, Kimberly Bussey, Edward Uchio, W Marston Linehan, J Carl Barrett, and John Weinstein. Mistaken identifiers: Gene name errors can be introduced inadvertently when using excel in bioinformatics. *BMC Bioinformatics*, 5(1):80, 2004.

[237] D. R. Zerbino and E. Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome Res*, 18(5):821–829, 2008.

[238] Yi Zheng, Liangjun Zhao, Junping Gao, and Zhangjun Fei. iassembler: a package for de novo assembly of roche-454/sanger transcriptome sequences. *BMC Bioinformatics*, 12(1):453, 2011.

Parte VII

Apéndices

Apéndice A

Script para dividir un fichero fasta

`split_fasta.rb` es un *script* para dividir ficheros en formato fasta con muchas secuencias, en otros ficheros que contendrán tantas secuencias como se haya indicado. Por ejemplo, con el comando

```
split_fasta.rb input.fasta 500
```

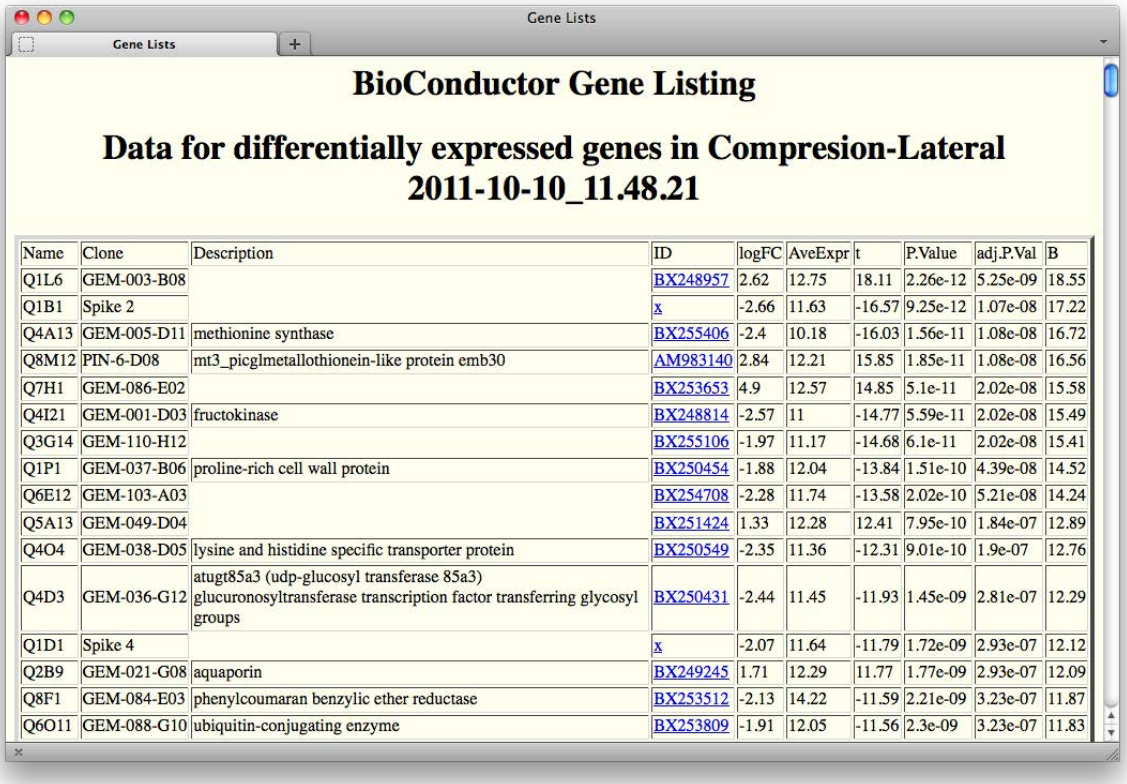
si el fichero input.fasta contiene 10 000 secuencias, al ejecutar el *script* se obtendrán 20 ficheros con 500 secuencias cada uno.

```
1  #!/usr/bin/env ruby
2  # Noe Fernandez Pozo 2010-04-17. script for spliting a fasta file by number of sequences.
3
4  if ARGV.size != 2
5    puts "incorrect number of arguments, you need a fasta file and a number of sequences per
         file; ruby split_fasta.rb input.fasta number_of_seqs"
6    Process.exit(-1);
7  end
8
9  (file,seq_number)=ARGV
10
11 num_seqs = 0
12 num_files = 1
13 output_name = file.sub(/\.fasta/,'')
14 output=File.open("#{output_name}_part#{num_files}.fasta", 'w')
15
16 File.open(file).each do |line|
17   if line =~/^>/
18     num_seqs += 1
19   end
20   if (num_seqs == seq_number.to_i + 1)
21     num_files += 1
22     output.close
23     output=File.open("#{output_name}_part#{num_files}.fasta", 'w')
24     num_seqs = 1
25   end
26   output.puts line
27 end
28
29 output.close
```

Apéndice B

Ejemplo de informe de MADE4-2C

El programa MADE4-2C devuelve varios ficheros útiles para continuar analizando tus datos y para acceder a la información obtenida. En este anexo se muestran como ejemplo una captura de pantalla del fichero en formato html que resume los datos de los genes expresados diferencialmente (figura B.1) y el informe generado por el programa para que el usuario pueda evaluar fácilmente la calidad y limpieza de sus datos.



Name	Clone	Description	ID	logFC	AveExpr	t	P.Value	adj.P.Val	B
Q1L6	GEM-003-B08		BX248957	2.62	12.75	18.11	2.26e-12	5.25e-09	18.55
Q1B1	Spike 2		x	-2.66	11.63	-16.57	9.25e-12	1.07e-08	17.22
Q4A13	GEM-005-D11	methionine synthase	BX255406	-2.4	10.18	-16.03	1.56e-11	1.08e-08	16.72
Q8M12	PIN-6-D08	mt3_picglmetallothionein-like protein emb30	AM983140	2.84	12.21	15.85	1.85e-11	1.08e-08	16.56
Q7H1	GEM-086-E02		BX253653	4.9	12.57	14.85	5.1e-11	2.02e-08	15.58
Q4I21	GEM-001-D03	fructokinase	BX248814	-2.57	11	-14.77	5.59e-11	2.02e-08	15.49
Q3G14	GEM-110-H12		BX255106	-1.97	11.17	-14.68	6.1e-11	2.02e-08	15.41
Q1P1	GEM-037-B06	proline-rich cell wall protein	BX250454	-1.88	12.04	-13.84	1.51e-10	4.39e-08	14.52
Q6E12	GEM-103-A03		BX254708	-2.28	11.74	-13.58	2.02e-10	5.21e-08	14.24
Q5A13	GEM-049-D04		BX251424	1.33	12.28	12.41	7.95e-10	1.84e-07	12.89
Q4O4	GEM-038-D05	lysine and histidine specific transporter protein	BX250549	-2.35	11.36	-12.31	9.01e-10	1.9e-07	12.76
Q4D3	GEM-036-G12	atugt85a3 (udp-glucosyl transferase 85a3) glucuronosyltransferase transcription factor transferring glycosyl groups	BX250431	-2.44	11.45	-11.93	1.45e-09	2.81e-07	12.29
Q1D1	Spike 4		x	-2.07	11.64	-11.79	1.72e-09	2.93e-07	12.12
Q2B9	GEM-021-G08	aquaporin	BX249245	1.71	12.29	11.77	1.77e-09	2.93e-07	12.09
Q8F1	GEM-084-E03	phenylcoumaran benzylic ether reductase	BX253512	-2.13	14.22	-11.59	2.21e-09	3.23e-07	11.87
Q6O11	GEM-088-G10	ubiquitin-conjugating enzyme	BX253809	-1.91	12.05	-11.56	2.3e-09	3.23e-07	11.83

Figura B.1: *captura de pantalla con un ejemplo del fichero html creado por MADE-4-2Colors con la información de los genes expresados diferencialmente*

A continuación se muestra un ejemplo del informe que se genera automáticamente MADE4-2C.

MADE-4-2C, MicroArrays differential expression for two colors report

User Name: Noé Fernández-Pozo
User Project: Ejemplo Tesis

Information about the project: Ejemplo para anexo de mi tesis

Plataforma Andaluza de Bioinformática
Universidad de Málaga

April 25, 2012

Contents

Chapter 1

Experiment description

Cy5 => RED
Cy3 => GREEN

Targets file name: **targets_CL2009.txt**

Table 1.1: Experimental design defined in your target file

Label	SlideNumber	FileName	Cy3	Cy5	repBiol
51-2-A	51-2-A	datos/Pinarray 51 15-12-2007 2-A.gpr	COM	LAT	1
51-2-Z	51-2-Z	datos/Pinarray 51 15-12-2007 2-Z.gpr	COM	LAT	1
52-2-A	52-2-A	datos/Pinarray 52 15-12-2007 2-A.gpr	COM	LAT	2
52-2-Z	52-2-Z	datos/Pinarray 52 15-12-2007 2-Z.gpr	COM	LAT	2
53-2-A	53-2-A	datos/Pinarray 53 15-12-2007 2-A.gpr	LAT	COM	3
53-2-Z	53-2-Z	datos/Pinarray 53 15-12-2007 2-Z.gpr	LAT	COM	3
54-2-A	54-2-A	datos/Pinarray 54 15-12-2007 2-A.gpr	LAT	COM	4
54-2-Z	54-2-Z	datos/Pinarray 54 15-12-2007 2-Z.gpr	LAT	COM	4

Number of chips: **8**
Number of spots: **3584**
Number of empty spots by design: **128**
Number of absences permitted in your analysis: **0**

SpotTypes file name: **SpotTypes.txt**

Table 1.2: Content of the spot type file

SpotType	ID	Name	Color
cDNA	*	*	black
vacío	empty	*	green

Fold change = **1.5**
P-value = **0.05**
Multiple testing correction method of P-values: **BH**

Chapter 2

Assessing quality

2.1 Background noise

Background noise box-plots in Fig. 2.1 must show equivalent distributions for each slide of your experiment. That means that the slides with highest and lowest background noise must be below the cyan dashed line.

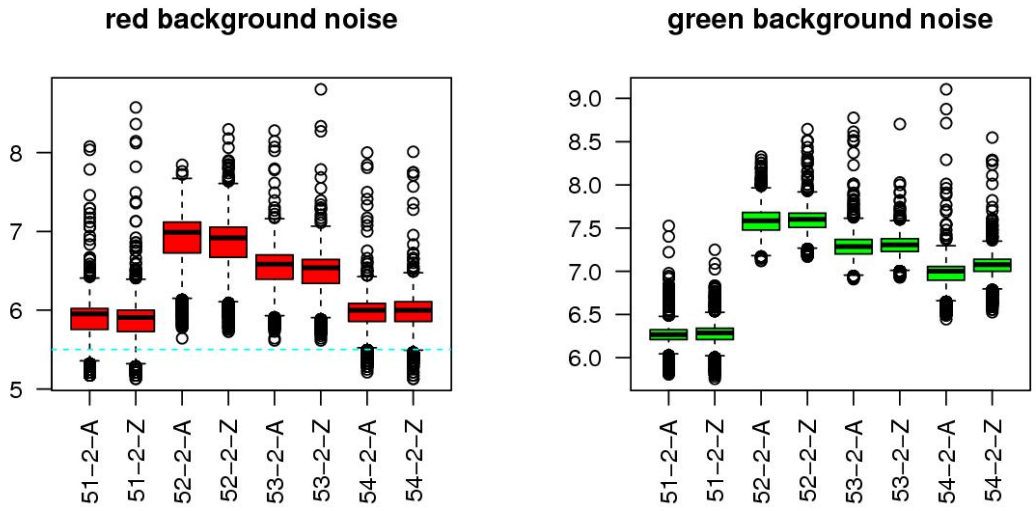


Figure 2.1: Box-plot of background noise in red (left) and green (right) channels of your original data [image `Background_BoxPlot01.jpg`]

2.2 Aspect of raw data

Background after hybridization must be homogeneous in both channels and in each block of every slide. So, ideal results should not have traces of a scratch nor stripes nor show any kind of gradient. In order to see make evident any unwanted pattern, results have been ranked. Presence of any kind of pattern may affect negatively the downstream analysis.

Please, verify if background aspect in figures 2.2, 2.3 and 2.4 show any spatial artifact. If they are evident, you have to consider repeating your experiment or reviewing your microarrayer.

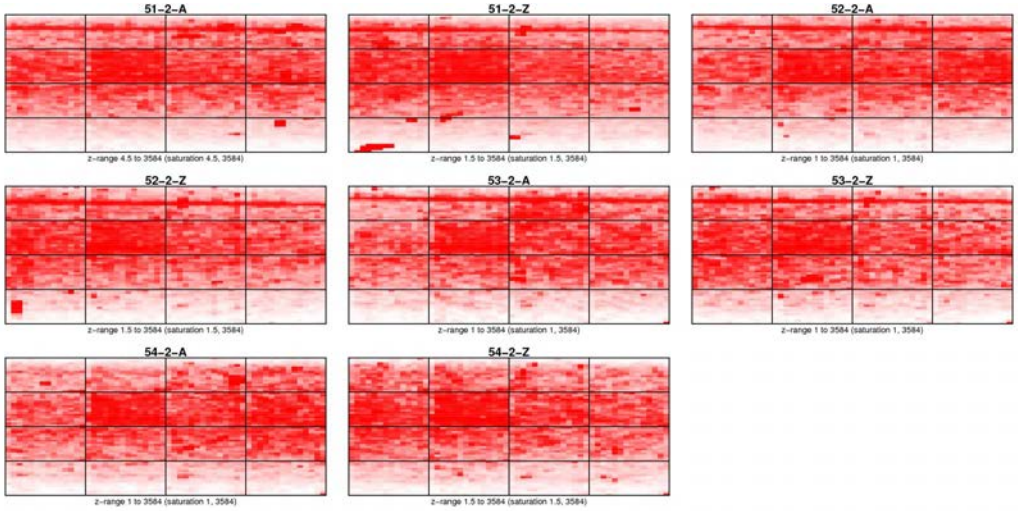


Figure 2.2: Red background noise with a rank color scale. Please, note that this is a false color image for highlighting spatial patterns [image `Background_Image01_RED01.jpg`]

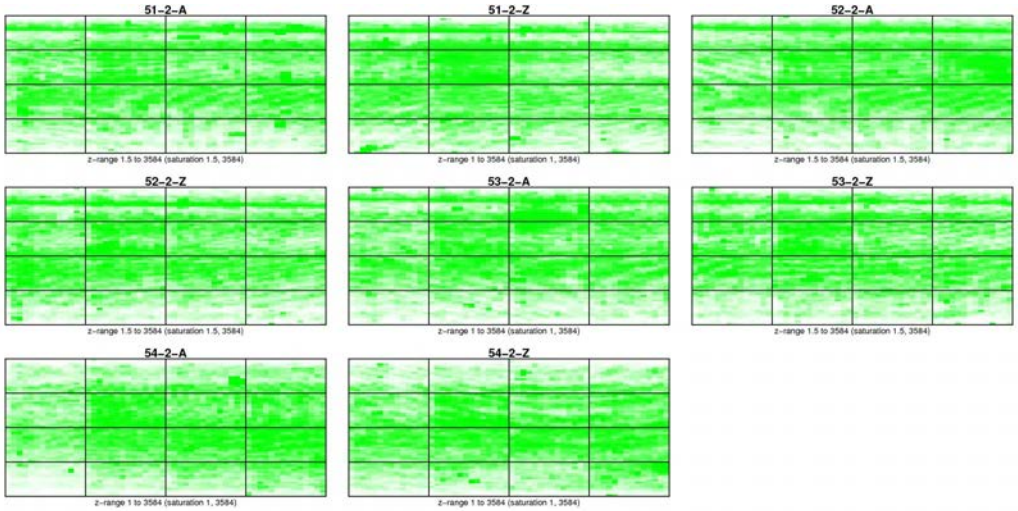


Figure 2.3: Green background noise with a rank color scale. Please, note that this is a false color image for highlighting spatial patterns [image `Background_Image_GREEN01.jpg`]

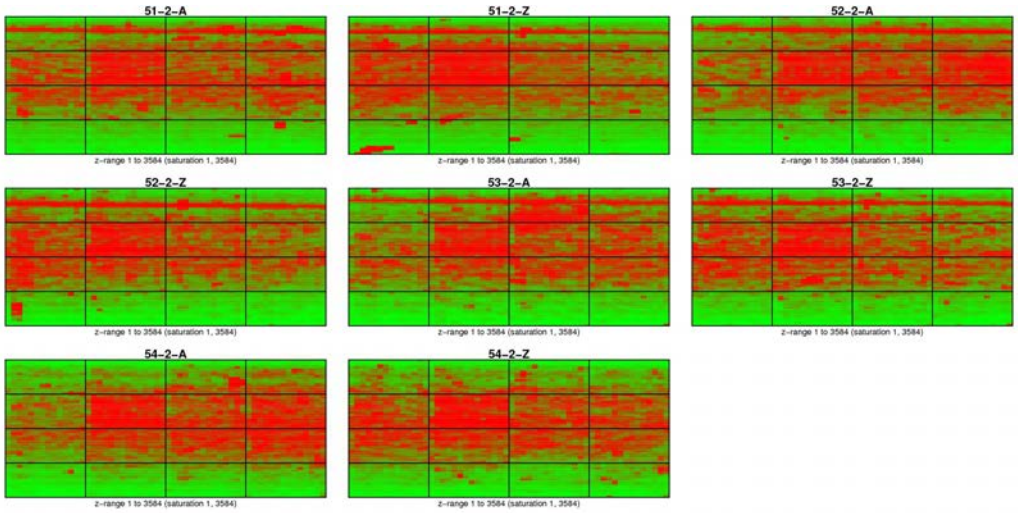


Figure 2.4: Red and green foreground signal in ranks. Ideal image should not present any scratch, strip, gradient or bias. If a clear bias is present, please, refer to normalized data below to see if it disappears with normalization [image `Background_Image_RedGreen01.jpg`]

The **signal-to-noise ratio** (SNR) is a good indicator for dye problems. Good data should have a SNR over 3 and with close values with both dyes in order to get reliable intensity signals. The red and green peaks of your signals are shown in figure 2.5. Peaks should be greater than the cyan reference line.

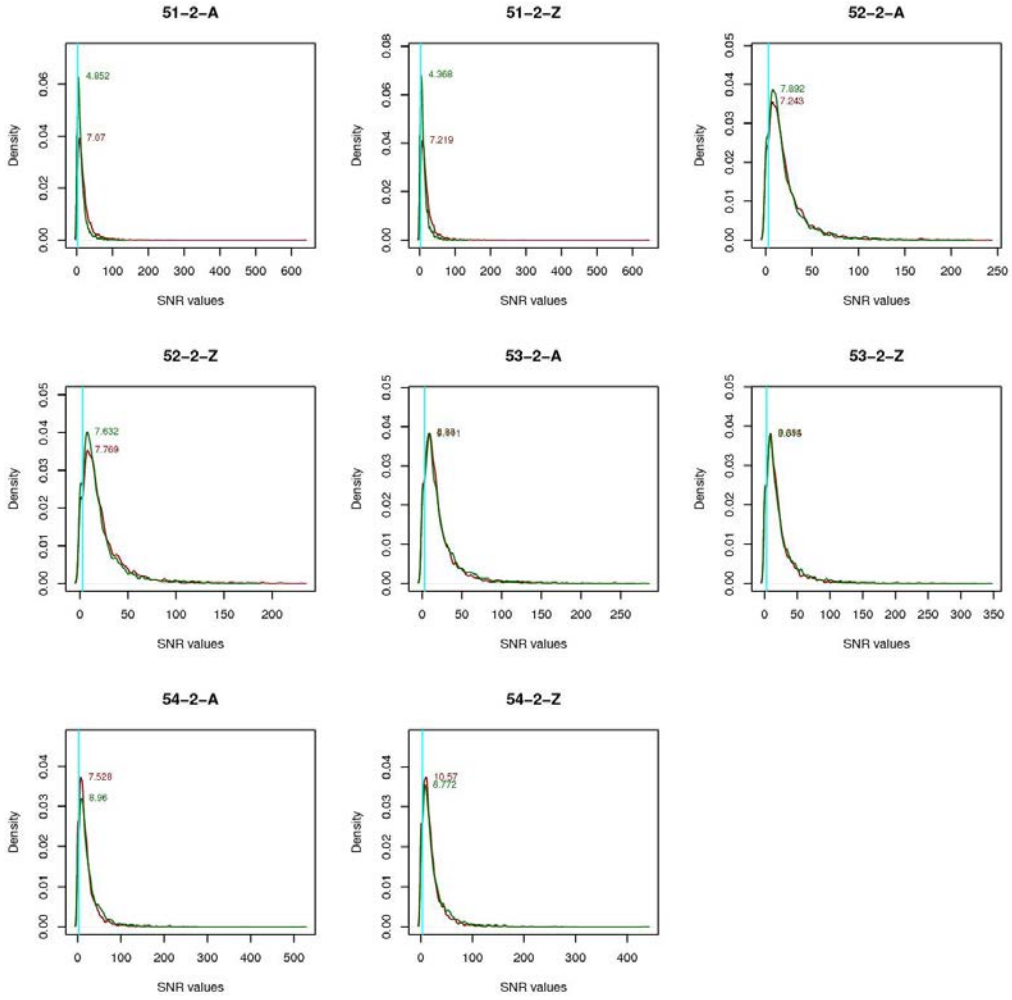


Figure 2.5: Density plots of your SNR for Cy5 and Cy3 channels in order to see if there is enough SNR or your spots for a reliable analysis

Reliable signal in slides:

51-2-A (G = 4.852, R = 7.07), 51-2-Z (G = 4.368, R = 7.219), 52-2-A (G = 7.892, R = 7.243), 52-2-Z (G = 7.632, R = 7.769), 53-2-A (G = 9.111, R = 8.88), 53-2-Z (G = 8.613, R = 9.394), 54-2-A (G = 8.96, R = 7.528), 54-2-Z (G = 8.772, R = 10.57)

Let's consider your raw data as correct and see their look converting the initial red and green data to M and A data. **M** is the log-ratio of signal intensities defined as $M = log_2(Red/Green)$ and can be considered the logarithmic ratio of gene expression. **A** is a measure of the overall intensity $A = log_2\sqrt{(Red \times Green)}$. This is another way to visualize uneven hybridization and missing spots.

Your M values after background correction are then presented in figure 2.6. Slide images must be homogeneous and devoid of any kind of gradient or bias. If a clear uneven hybridization is present, please, refer to normalized data below to see if it disappears with normalization.

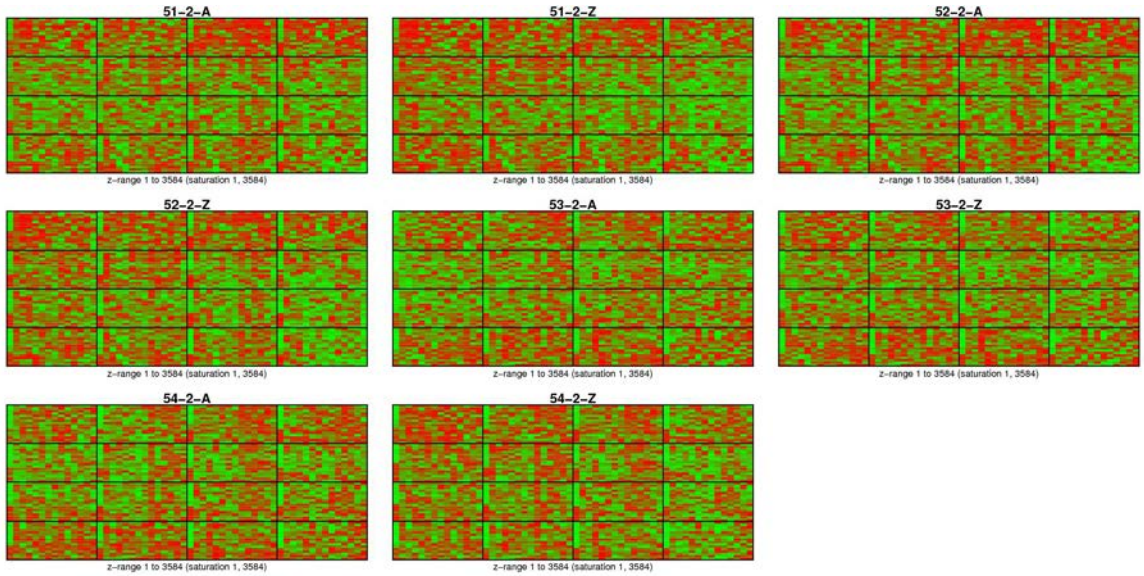


Figure 2.6: Hybridization signal of raw data presented as ranked values of **red signal divided by green signal (M)** in order to highlighting spatial patterns. [image `RawMAImageM01.jpg`]

Intensities (A) of your raw data presented in figure 2.7 must result in homogeneous images devoid of any kind of gradient or bias. Intensity is supposed to lack any uneven signal since it will not be corrected in normalization. Hence, if you find a clear bias, please consider to discard your hybridizations and repeat your experiment.

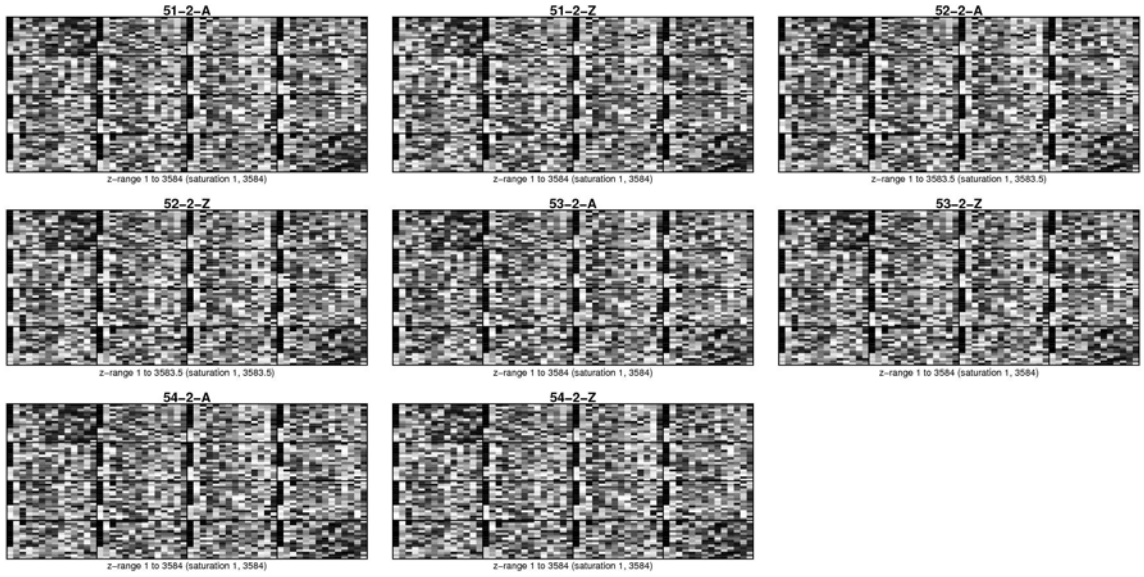


Figure 2.7: Signals of your hybridization presented as ranked raw values of **intensity (A)** in order to highlighting spatial patterns. [image `RawMAImageA01.jpg`]

M and A data derived of your raw data can be represented as MA-plots (Fig. 2.8). Please, note that if you have dye-swapped experiments, they should appear as specular images as A values are maintained while M values change their sign. Most spots should appear in range $10 < A < 16$. If there is no trend, this

cloud of spots must be around the M = 0 value. If you observe a downward of upward curvature of the points distribution, there is an imbalance between red and green channels. Please, note that if this trend does not disappear after normalization, repeating hybridization should be considered.

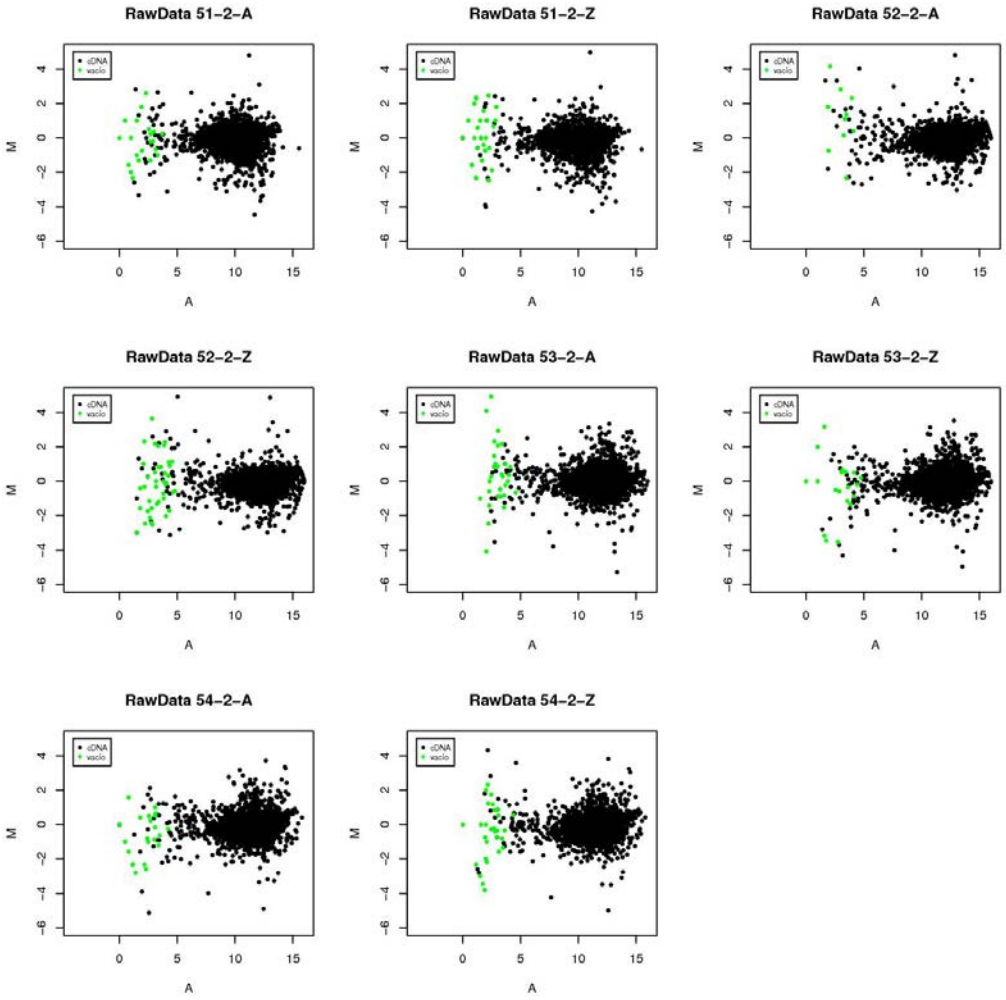


Figure 2.8: MA-Plot of your raw data. If you defined spot types, they are marked on the image [image `RawDataMAPlot01.jpg`]

MA-plots can also include the representation of the loess adjustment of data (Fig. 2.9). Each line curves for each print-tip group. Raw data and background adjusted data use to deploy a mesh of adjustment lines. This should be corrected with normalization. Things to look for in this MA-plot are saturation of spots and the trend of loess curves, which is an indicator of the amount of normalization to be performed.

Another important aspect of next plot (Fig. 2.9) is that you will see which spots are considered unuseful in every slide.

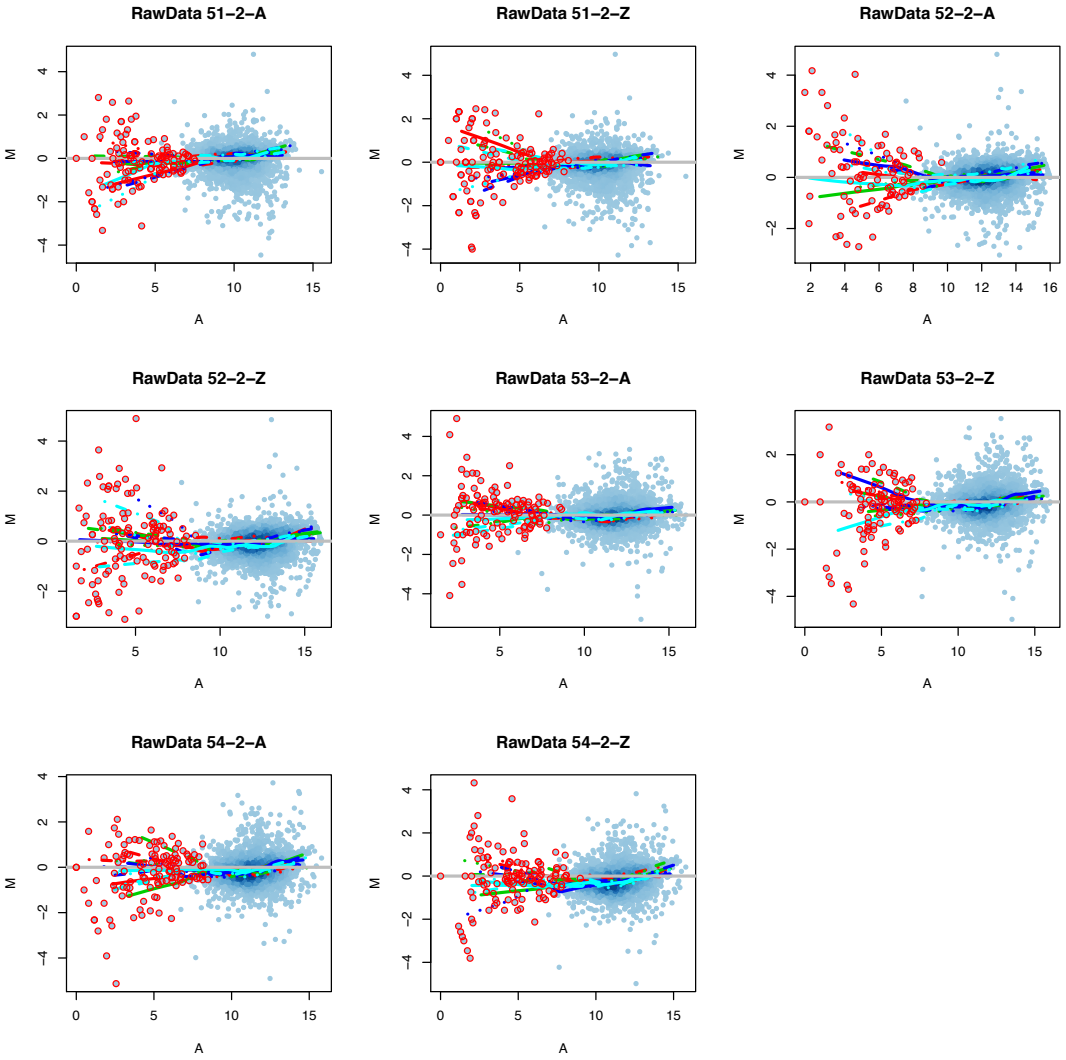


Figure 2.9: MA-Plot of your raw data with lowess adjustment for every block. The points cloud has been colored in blue levels to illustrate where most of points are. Spots that will not be analyzed by low quality are circled in red. [image `RawDataMAPlotLwAdj01.jpg`]

2.3 Background correction

Background correction of signals is essential to obtain good sensitivity in the analysis. There are some authors that recommend to avoid background correction, but it has been proven that, although it do not improve the detection of differentially expressed genes, no background correction underestimates systematically fold-chages in expression [8].

There are several methods to correct the background, being `normexp` (normal-exponential) with an offset of 50 the best as published in [9] and [8]. This method is derived from the RMA method developed for Affymetrix chips. `normexp` adjusts the foreground adaptively for the background intensities that guarantees a positive value of intensity for all spots while decreasing the variability at low intensities. It uses an offset to damp the variation of the log-ratios for very low intensity spots towards zero. Although this method

renders more biased data than no correction, the improvement in precission outweighs the increase in biass when detecting differential expression [9]. Other method that provide good results are `morph` and `vsn`, and even no background subtraction, but background subtraction will provide the worst differentially expressed gene list later.

Images of red and green background and foreground signal do no change appreciably with this mathematical treatment. So, they will not be presented.

However, background correction with `normexp` produces MA-plots that can differ significantly in low intensity points. Compare the raw data in figure 2.8 with the new representation in figure 2.10. Note that your identified spots (if they exist) should have the same distribution.

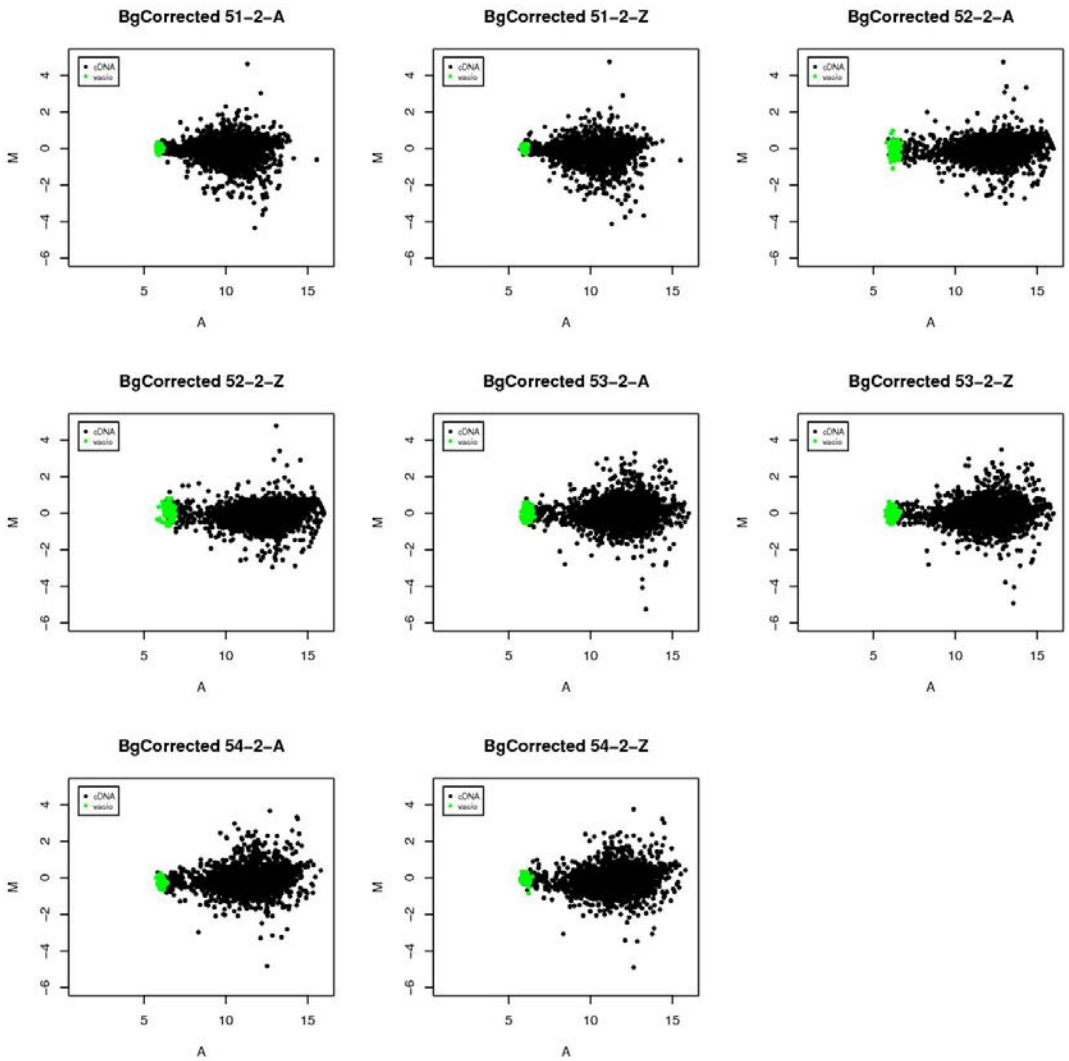


Figure 2.10: Background corrected MA-Plot [image `BgCorrectedMAPlot01.jpg`]

MA-plots can also include the representation of the loess adjustment of data (Fig. 2.11). Each line curves for each print-tip group. Raw data and background adjusted data use to deploy a mesh of adjustment lines. This should be corrected with normalization. Things to look for in this MA-plot are saturation of spots and the trend of loess curves, which is an indicator of the amount of normalization to be performed. In this plot, bad spots are the same for all slides in order to illustrate where are these unusable features.

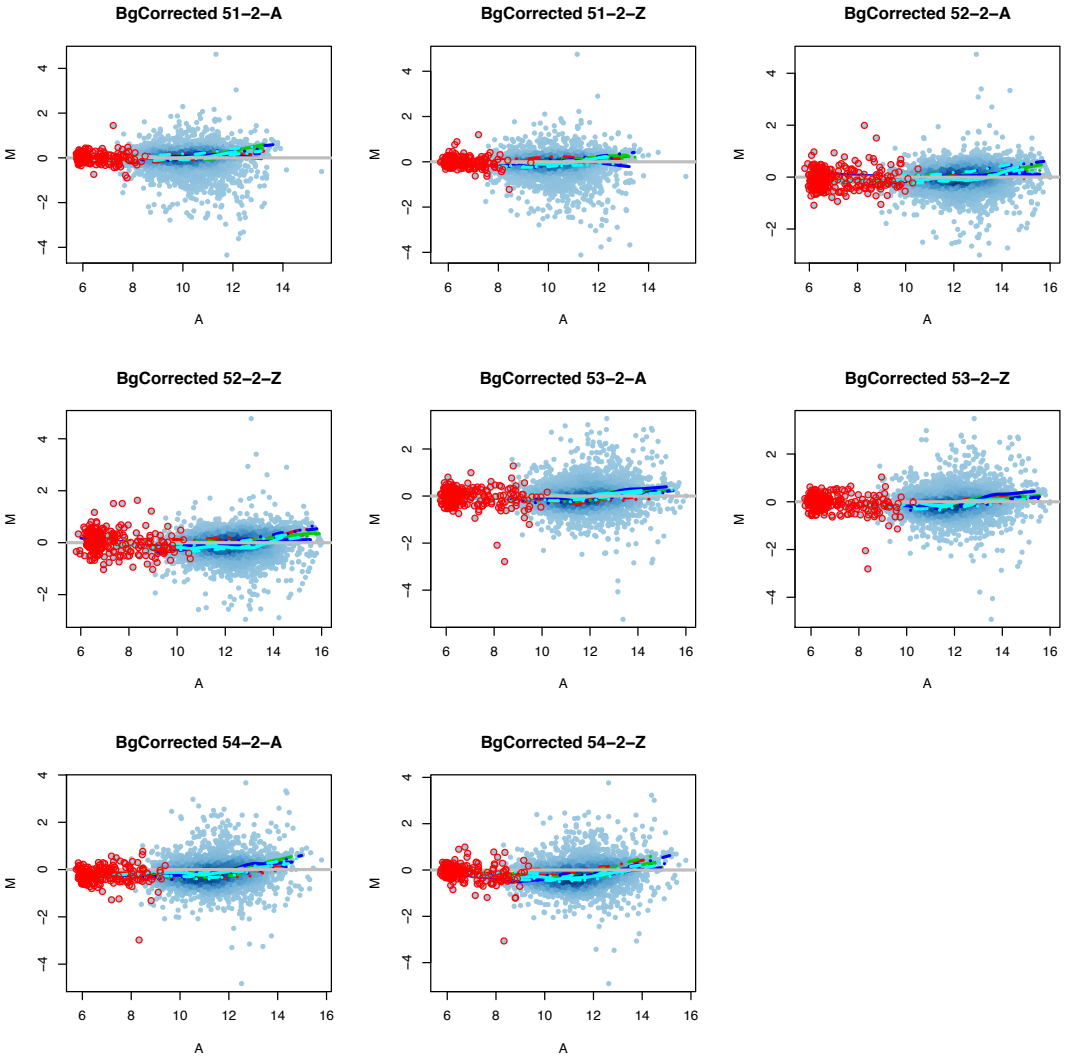


Figure 2.11: Background corrected MA-Plot with lowess adjustment for every block. The points cloud has been colored in blue levels to illustrate where most of points are. Spots that will not be analyzed by low quality are circled in red. [image `BgCorrectedMAPlotLwAdj01.jpg`]

2.4 Normalization

Before analyzing if data are appropriate for biological interpretation, background-corrected data will be normalized. The interest of microarray normalization is to extract quantities of interest from the data while correcting for systematic variations, correcting for different types of dye biases (e.g. intensity, spatial, plate biases), and controlling the stochastic variability. Normalization is needed to ensure that observed differences in intensities are indeed due to differential expression and not experimental artifacts. However, normalization clearly impacts on identification of differentially expressed genes and introduces new artifacts or trends in your data. Hence, no perfect or ideal method exists.

First, only reliable spots will be included in normalization procedures. Figure 2.12 is a theorical image of one chip where spots circled in black will not be considered in the analysis. Empty features (unprinted

spots) are not considered to calculate the final yield of usable spots. It is supposed that loss of spots due to experimental errors or background correction method (different than `normexp`) should be under 15 % [9].

You can recover all normalized data as a `maNorm` object in your `Results` folder with names starting by the normalization method and ending by `_maNorm.data`.

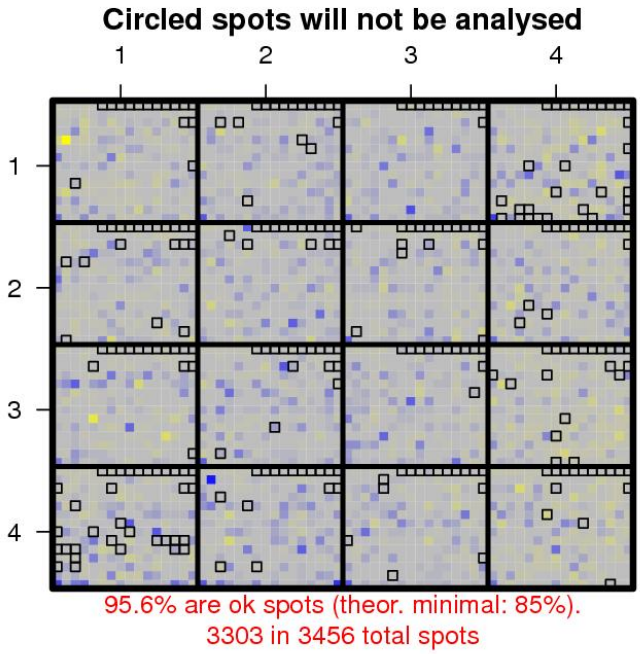


Figure 2.12: One chip displaying in black squares the spots that will not be analyzed in this project. Note that you must have >85% of useful spots for a reliable experiment [image `BadSpots01.jpg`]

Spots that are circled in black in figure 2.12 will not influence normalization of other spots. There have been developed several methods to deal with this. Normalization methods tested here are:

Loess `Loess` is a new polynomial function based on `Lowess` with different defaults. It is more robust than `Lowess` since it guards against deviant points distorting the smoothed points. `Loess` creates a smoothing local regression for summarizing multivariate data using general curves and surfaces, capturing the non–linear dependence of M values on the overall intensity A, while ensuring that the computed normalization values are not driven by a small number of differentially expressed genes with extreme log–ratios. This normalization changes M values while leaving A values intact. The best way to correct spatial bias consist of adjusting a loess curve for every block (corresponding to a single print-tip: `printTipLoess`), provided that there are at least 150 spots per block. If this condition is not fulfilled, `global loess` is used. `Loess` also assumes that the bulk of the probes on the array are not differentially expressed, that is, are around $M = 0$.

Loess + scale `Loess` normalization does not guarantees data comparability among arrays. This inter-array normalization is performed by the *scale* method based on a robust estimate such as the median absolute deviation (MAD). It has been proposed several times that scaling destroys biological information

[7].

Loess + quantile `Quantile` is a non parametric normalization based upon quantiles to correct probe level intensities among arrays, that is R and G channels, to obtain the same density profile (see figure 2.23 below). This method assumes that the distribution of gene abundances is nearly the same in all samples. Unfortunately, this normalization does not make dissapear spatial trends and produces a very aggresive adjustment. Although described that combination of `loess + quantile` do not provide reliable results [2], it is included since both are very popular. In any case, if your data seems to be comparable only with `Loess` normalization, it is preferable to use only-`loess` normalized data than data with the additional `quantile` transformation.

VSN `VSN` (**V**ariance **S**tabilization **N**ormalization) produces an affine transformation whose aim is to calibrate systematic experimental factor such as labelling efficiency or detector sensitivity. A global logarithmic transformation is performed for variance stabilization. There may be other factors influencing the variance, such as gene-inherent properties or changes of the tightness of transcriptional control in different conditions, but are addressed by other methods beyond `VSN`. `VSN` does not deal with spatial biases [1].

VSN + Loess The combination of these methods in this order has been described to provide the most consistent differential expression data [2].

Once these methods have been applied, normalized data will be compared with raw data in next sections in order to determine if data have enough quality for analysis, and to determine which normalization method is the most suitable for your data.

2.5 Hybridization quality

2.5.1 Taking into account hibridization signals

In section 2.2 you have obtained a first approach to the aspect (quality) of your data. Now you can obtain more information about it and decide if results obtained are reliable or not. Figure 2.13 shows a weighting measure of every hybridization slide for all kind of data generated for your microarrays.

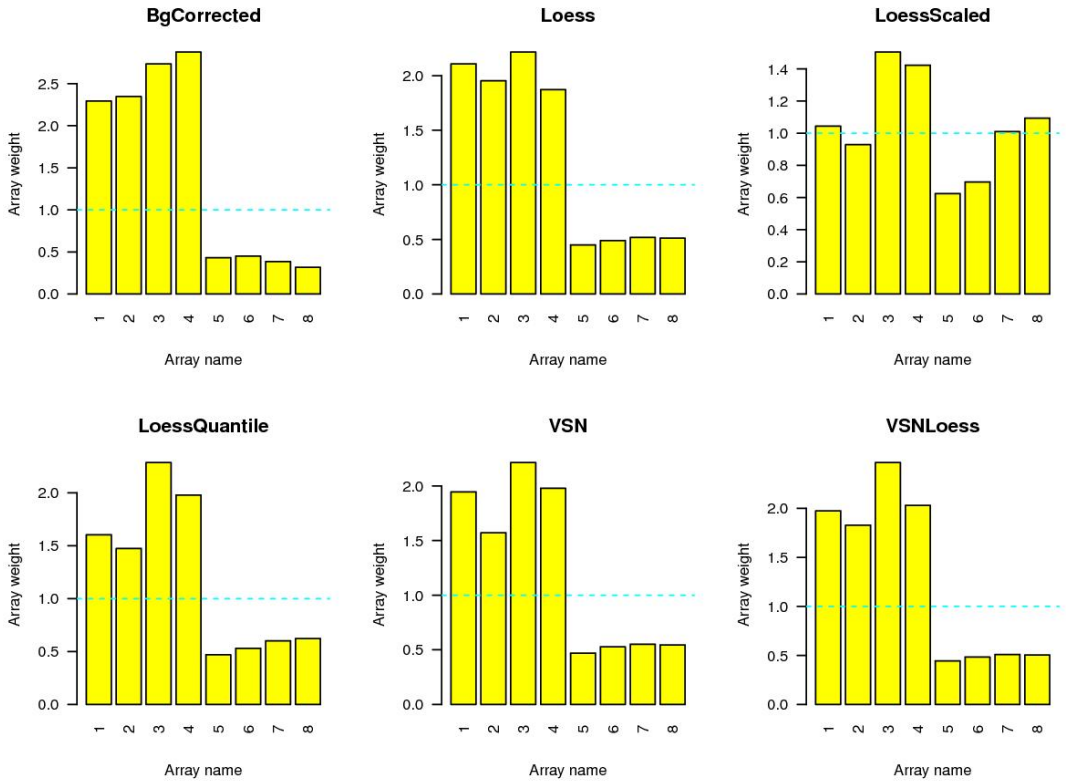


Figure 2.13: Quality weights of slides. It should be preferable that yellow bars were homogeneous around 1.0. If maximum and minimum weights are too far away, you can remove the ones with the lowest weight for a new analysis, or preferably repeat your hybridization. [image `ArrayQualityWeights01.jpg`]

Interpretation of this results is:

Best methods for SLIDE HIBRIDIZATION WEIGHTING are: Loess, LoessScaled, LoessQuantile, VSN, VSNLoess

The figure 2.14 reflects the quality of data from every blocks before and after normalization. White blocks are considered of very low quality, and dark blue blocks are considered of high quality. The better is to have all blocks in nearly the same blue intensity. If very different blue intensities are shown, you should consider to repeat your hybridization.

Figure 2.14: Quality weights by blocks [image `GridWeights01.jpg`]

2.5.2 Correlation of experimental data

Dendrograms of the experimental condition are now analyzed in order to see if slides cluster by experimental condition or cluster by any other reason.

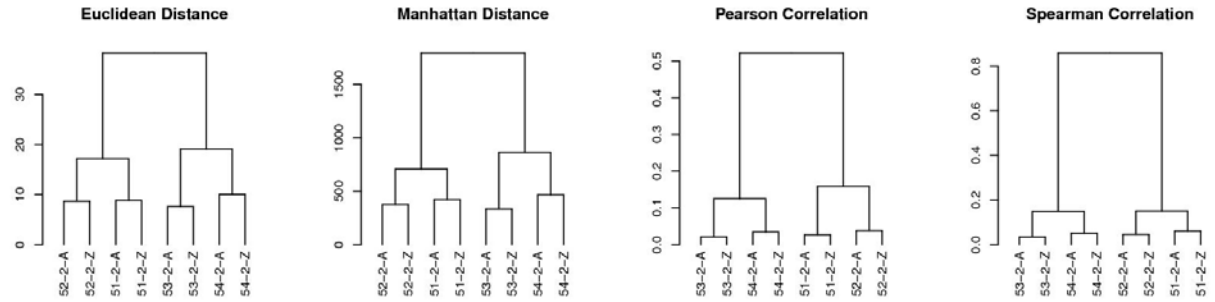


Figure 2.15: Raw data [image `ARawDendrograms01.jpg`]

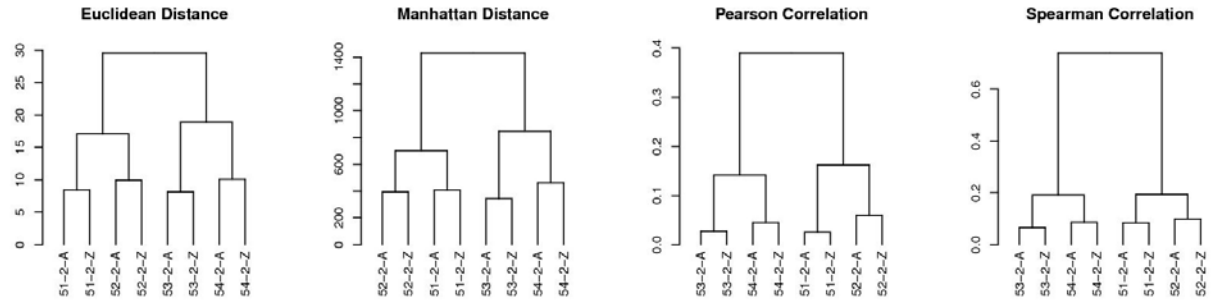


Figure 2.16: Background corrected data [image `BgCorrectedDendrograms01.jpg`]

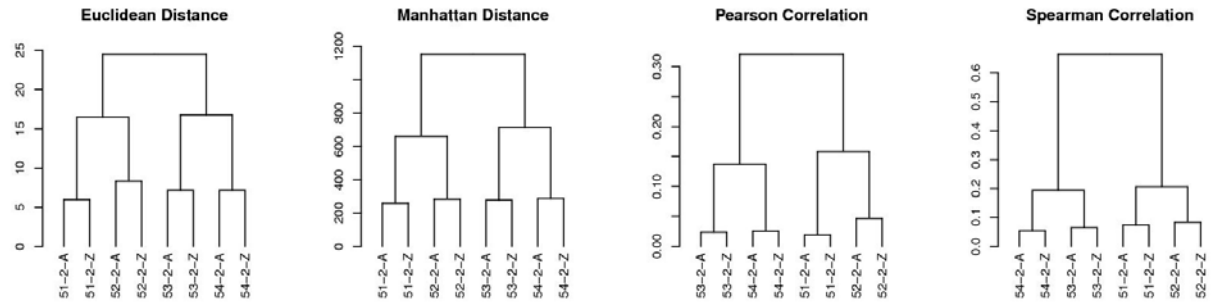


Figure 2.17: Loess-normalized data [image `LoessDendrograms01.jpg`]

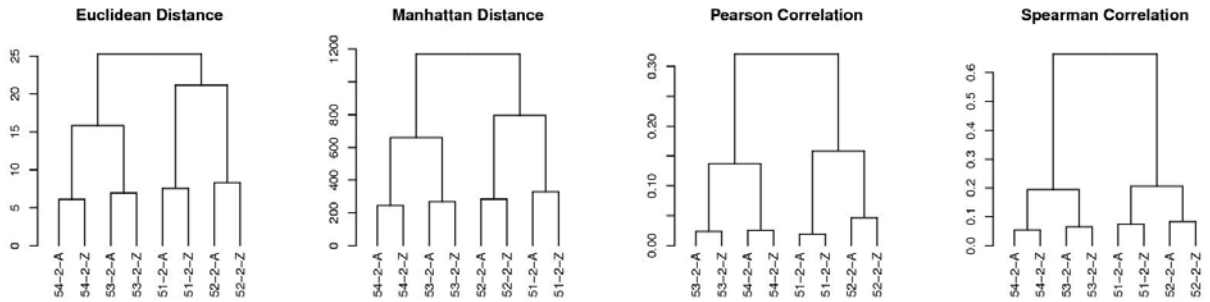


Figure 2.18: Loess- plus scale-normlized data [image LoessScaled_Dendrograms01.jpg]

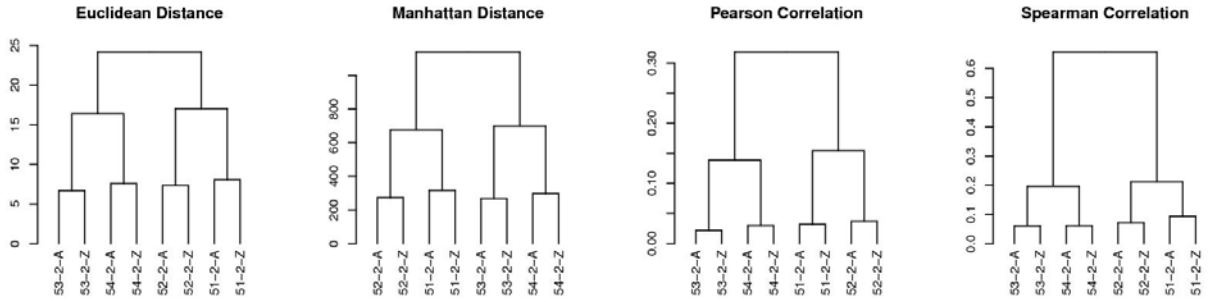


Figure 2.19: Loess- plus quantile-normalized data [image LoessQuantileDendrograms01.jpg]

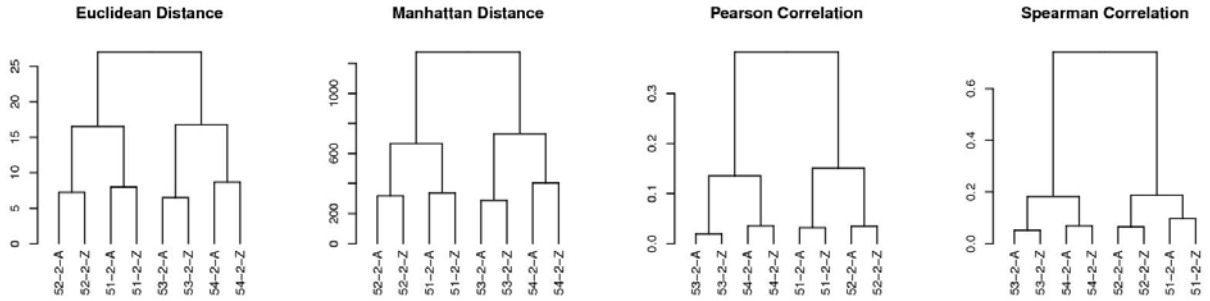


Figure 2.20: VSN-normalized data [image VSNDendrograms01.jpg]

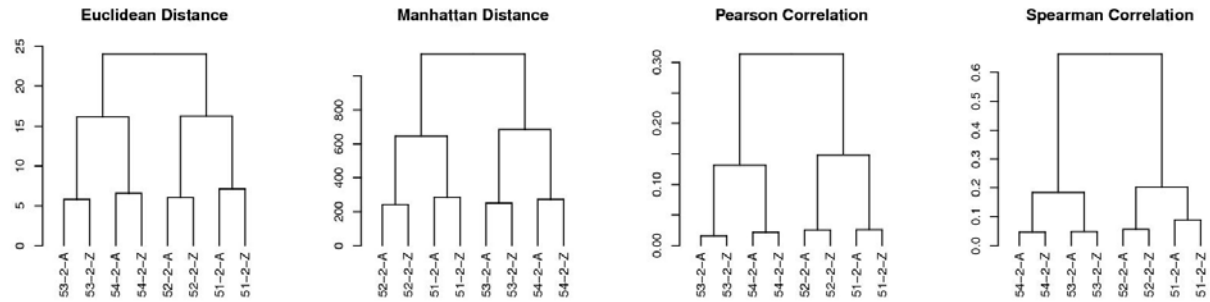


Figure 2.21: VSN- plus loess-normalized data [image VSNLoessDendrograms01.jpg]

2.5.3 Consistence after normalization

The figure 2.22 is a box-plot comparison of data normalized in different ways (see 2.6). After normalization, box-plots must be aligned by their median value (black line inside the red box) in a value of M = 0. The size of boxes and the range of dashed lines must be similar if normalization has done its work.

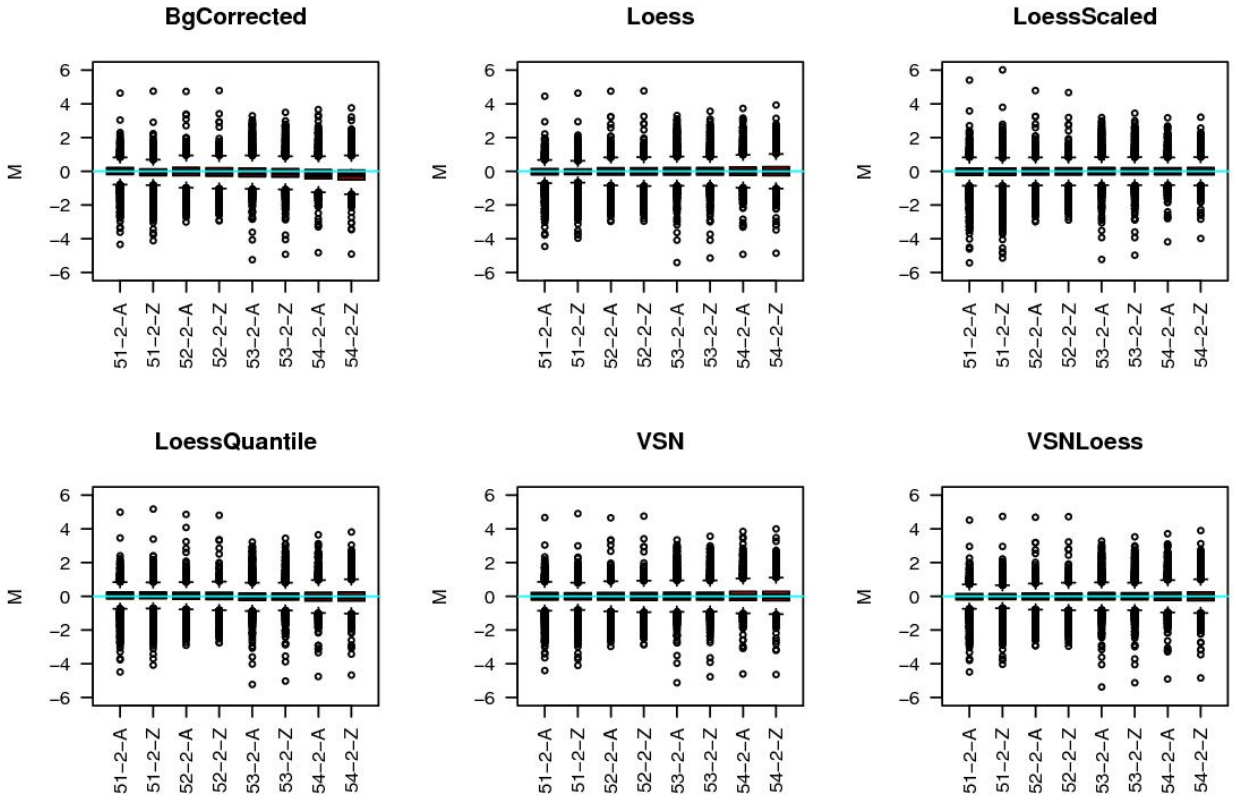


Figure 2.22: Comparison of normalization methods vs raw data through box plots [image BPNormalizeData01.jpg]

Best methods for BOXPLOT HOMOGENEITY CRITERIA are: LoessScaled

Density distribution of spots intensities must be similar in R and G channels at least after normalization. Maximum of curves must be on the same intensity value. Different normalization methods carry out

different density adjustments. In figure 2.23 you can compare the resulting distribution of your data according to normalization methods. Small shoulders after or before the main peak correspond to apparent saturation and quantization, respectively. Both effects are undesirable, although quantization shoulder is less detrimental. A large fraction of data must be concentrated at high intensities; if they were at low ones, most spots (features, genes) are very dim and analysis will lack of power.

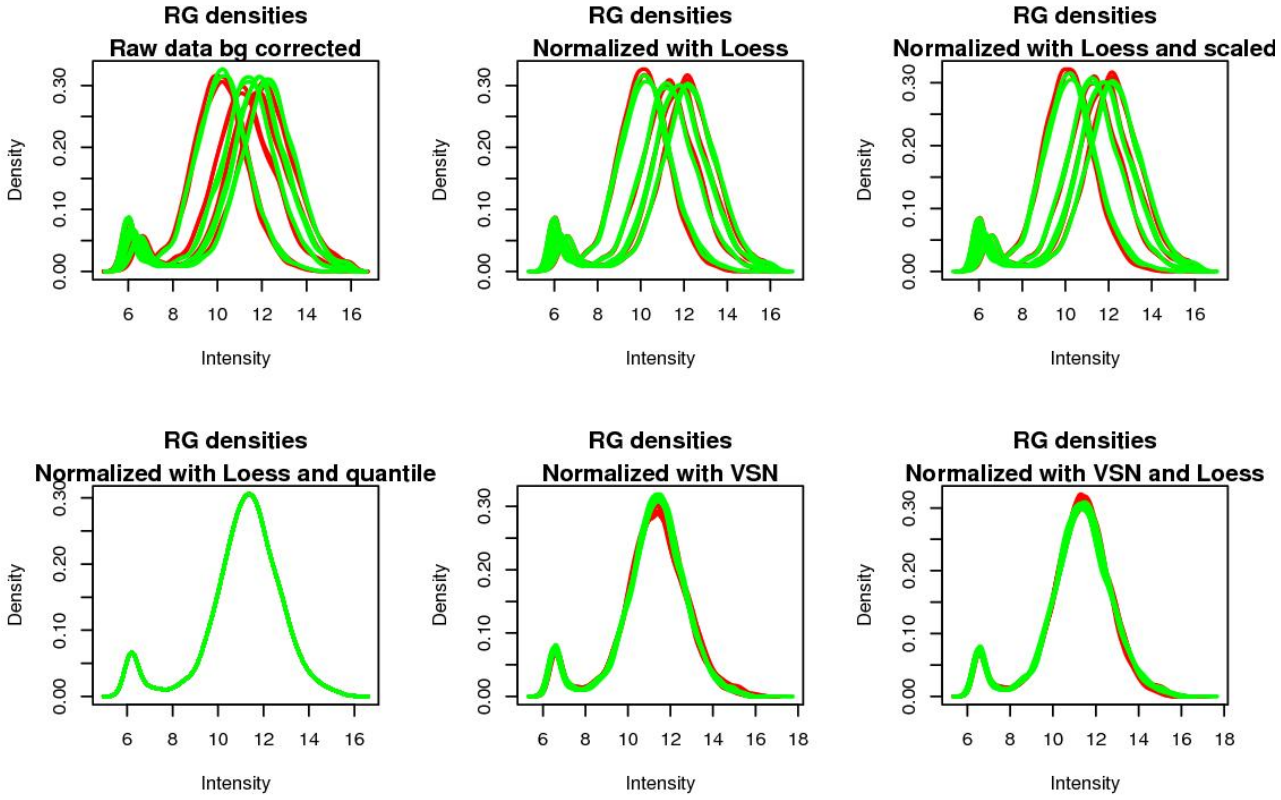


Figure 2.23: Distribution of intensity values of raw and normalized data [image `DensityPlots01.jpg`]

QQplots serve to have an approximation to the gaussian distribution of values. Usually, raw data are not clearly adjusted to the quartile line while normalized ones are. Tails detaching from the quartile correspond to differentially expressed genes or outliers. It is expected that most point match with the quantile line. Figure 2.24 contains the QQplot for raw and normalized data.

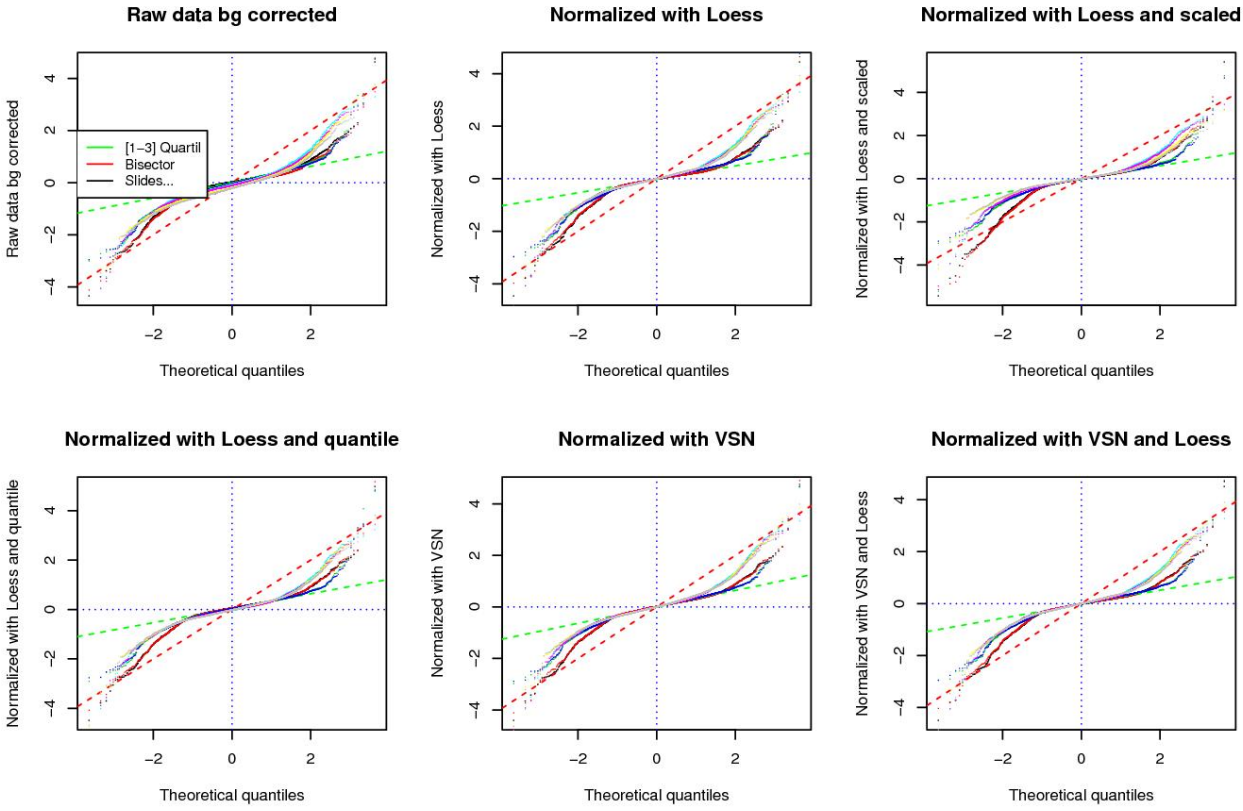


Figure 2.24: QQplots of raw and normalized data [image `QQPNormalizeData01.jpg`]

You must be sure that experimental variability (measured by the log ratio values [M]) is not dominated by the intensity expression level (A). The scatterplot of M versus the rank of mean intensity together with the linear adjustment of points (red line) helps to identify any possible dependence. If there is no variance-mean dependence, then the red line should be approximately horizontal. This serves to discard normalization methods that increase the slope of the raw or background-corrected data.

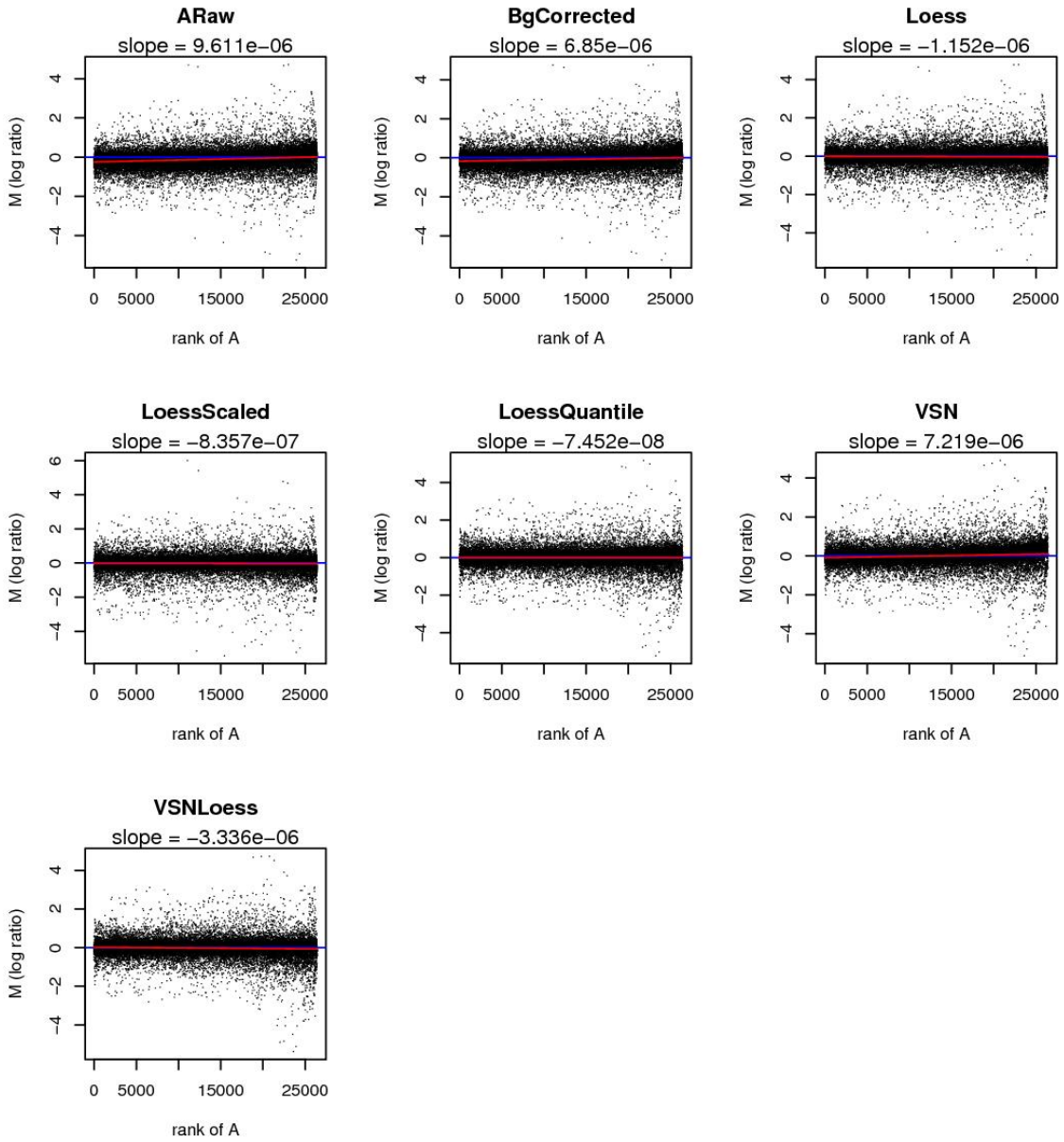


Figure 2.25: Scatterplot of M values of raw and normalized data on the *y*-axis versus the rank of the mean intensity (A) on the *x*-axis. The red illustrates the trend of M vs. A. It should be approximately horizontal when no substantial dependence exists. The rank scale is used for the *x*-axis in order to distribute the data evenly along the *x*-dimension and allows a better visual assessment of the log ratios M as a function of the mean. [image `MvsRankedA-plot01.jpg`]

Best methods for VARIABILITY AND INTENSITY INDEPENDENCE are: Loess, LoessScaled, LoessQuantile, VSNLoess

Complementary to the previous figure (Fig 2.25), you can see if experimental variability (measured by the standard deviation) is not dominated by any dependence on the mean expression level (A). Their scatterplot allows to identify visually any possible dependence. The red dots depict the running median estimator (window-width 10%). If there is no variance-mean dependence, then the line formed by the red dots should be approximately horizontal (Figure 2.26).

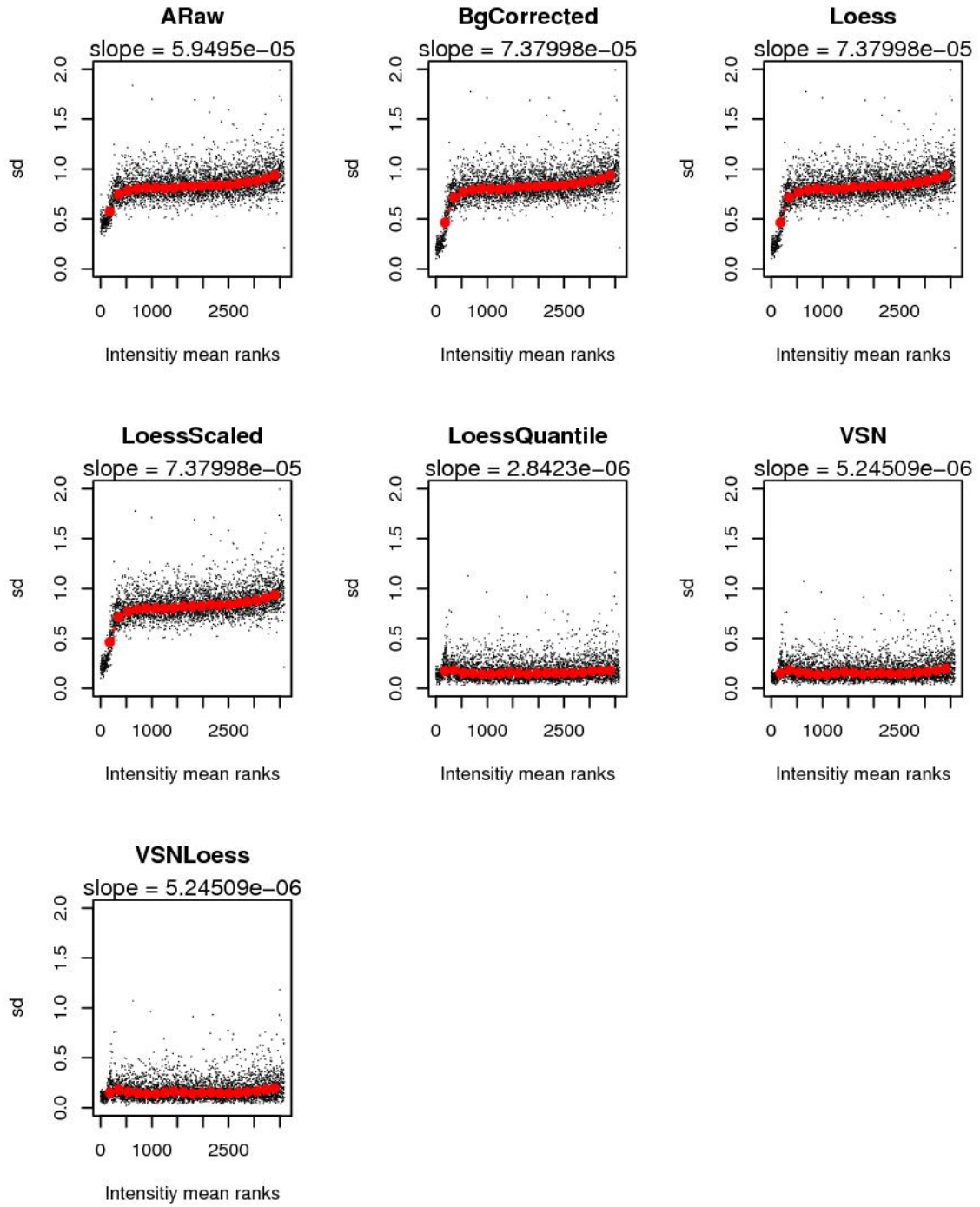


Figure 2.26: Empirical standard deviation of raw and normalized data on the *y*-axis versus the rank of the mean on the *x*-axis. The red dots show the running median of the standard deviation. Rationale of A rank is as in Fig 2.25. [image `meanSdPlots01.jpg`]

2.5.4 Correlation of intensity distribution

It is supposed that the expression level of a gene should ideally remain the same across multiple replicated slides. The **variability** of M values for each gene can therefore be used to compare normalization methods. The standard deviation of the log ratio intensity (M) is calculated for each gene over the slides. A smaller standard deviation is indicative of a more effective normalization procedure [10]. You can see the variability of your technical replicates in Fig. 2.27

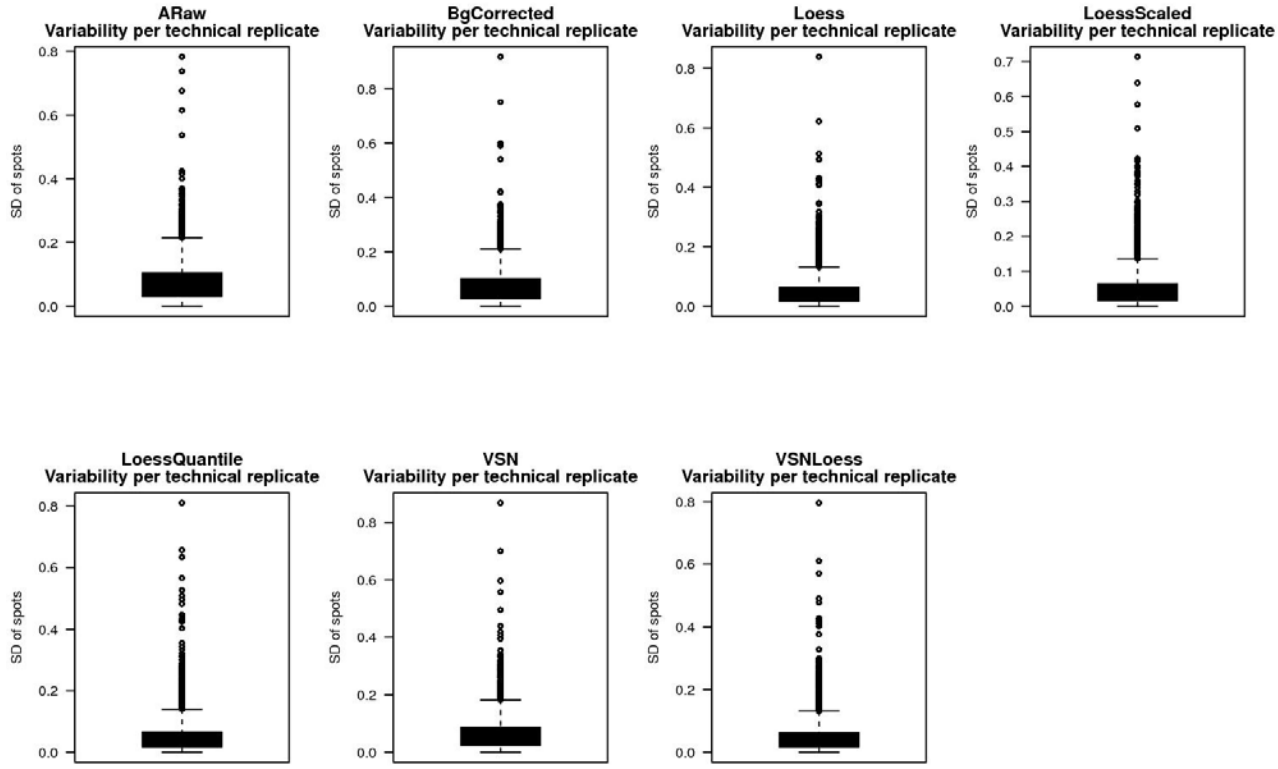


Figure 2.27: Box plots of variability among your technical replicates for every normalization method. The ideal case is when variability of every technical replicate is the same within a normalization method. [image `Variab_TechReplicates01.jpg`]

The comparison of data variability before and after normalizations is observed in Fig 2.28

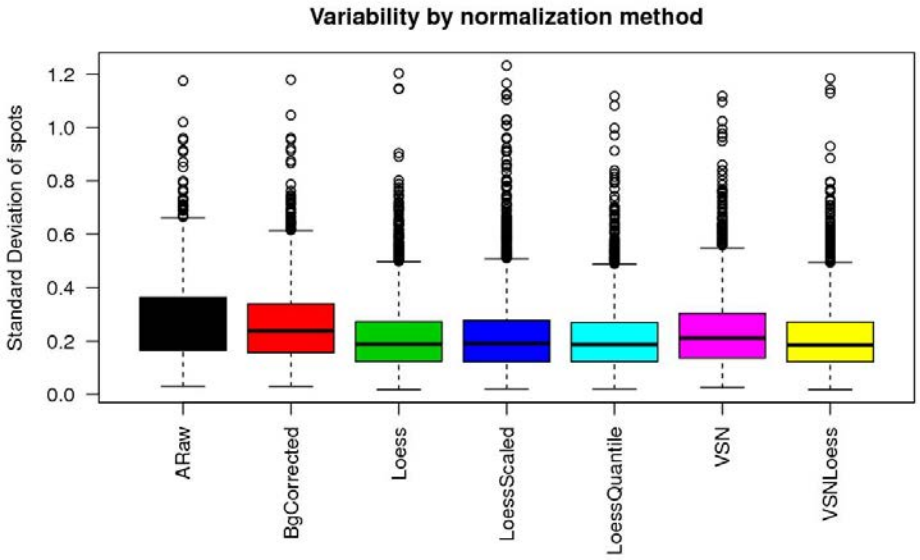


Figure 2.28: Bar plot of the mean replicate variability for your data set. Larger values indicate a higher variability across slides. It is not unexpected that `VSN` performs the best since this method specifically aims to stabilize the variance across the replicated arrays. [image `VariabilityByNormMethod01.jpg`]

Table 2.1: Variability of biological replicates between normalization methods. Lower values indicate low variability, that is, high correlation.

	ARaw	BgCorrected	Loess	LoessScaled	LoessQuantile	VSN	VSNLoess
Mean of variability	0.07293	0.07069	0.04544	0.0464	0.04747	0.06053	0.04515
SD of variability	0.276	0.26	0.211	0.2184	0.2099	0.2352	0.2097

Methods improving spot correlation are: LoessQuantile, VSNLoess, LoessScaled, Loess, VSN

Kolmogorov-Smirnov test (KS) is a goodness-of-fit test of two continuos distributions. Normalization methods can be evaluated with it based on the rationale that an effective normalization procedure sould result in two similar (ideally identical) distributions with a small, ideally zero-valued, KS statistic, while different distributions will generate large KS statistic [6]. It is not unexpected that `quantile` normalization performs the best since this method forces the empirical distributions in different slides to be identical, as can be seen in figure 2.23 where red and green densities are overlapping.

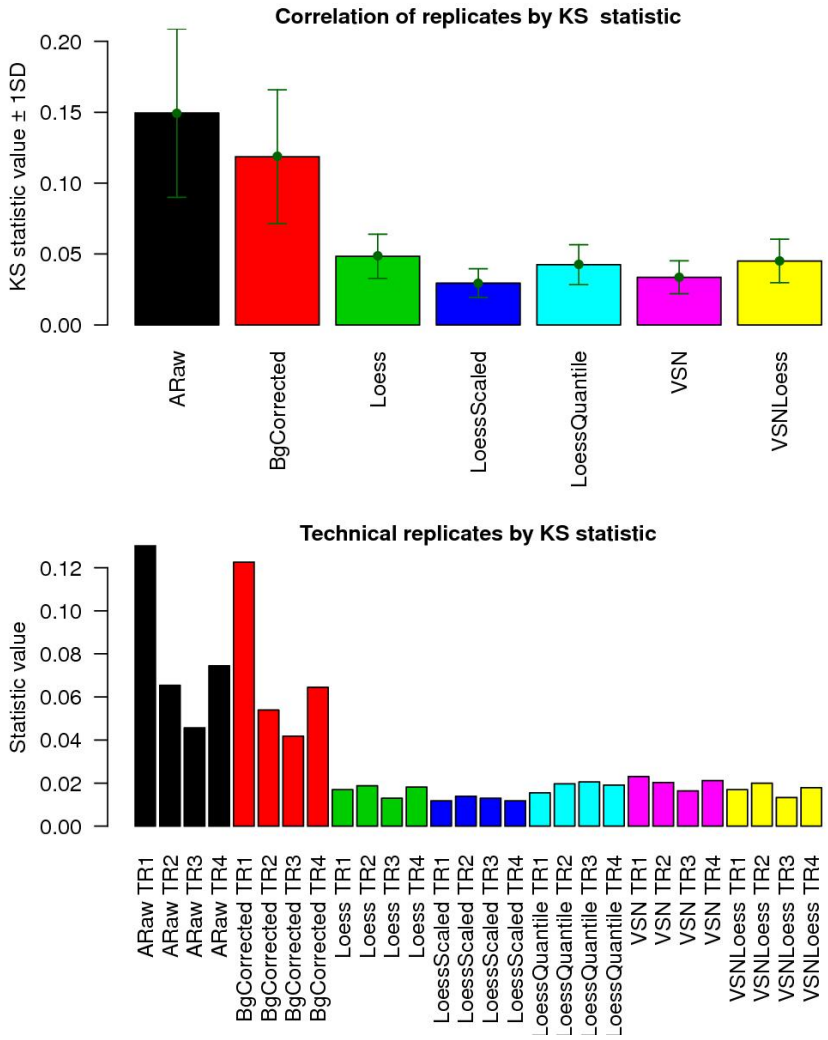


Figure 2.29: **Upper**: Bar plot of the mean of KS statistic between pairs of slides for your data set. **Lower**: Bar plot of KS statistic for every technical replicate. Larger values indicate low correlation across slides. [images `EvalNormByKS01.jpg` and `TrechRepCorrByKS01.jpg`]

Another way to study goodness-of-normalization is regarding the correlation between replicates of normalized data. This correlation will be measured with two different methods: Spearman correlation and Pearson correlation. The **Pearson product-moment correlation coefficient** r is a measure of the correlation (linear dependence) between two replicates. But r is neither distributionally robust, nor outlier resistant, so its value can be misleading if outliers are present. **Spearman's** ρ is a non-parametric product moment correlation coefficient based on ranks rather than raw expression levels without making any other assumptions about the particular nature of the relationship between the variables. This makes it less sensitive to extreme values in the data, has been widely used for comparing different normalization methods [3] and

seems to provide more robust measure of reproducibility than Pearson's *r* correlation coefficient [5].

If you have different technical and biological replicates, technical replicates do have a high correlation coefficient while the whole experiment replicate display a much lower correlation (usually under 0.5). In cases where any **correlation coefficient has a negative value**, that means that your data are inversely correlated, which should not be the case for technical nor biological replicates. You should then reconsider to repeat your hibridization.

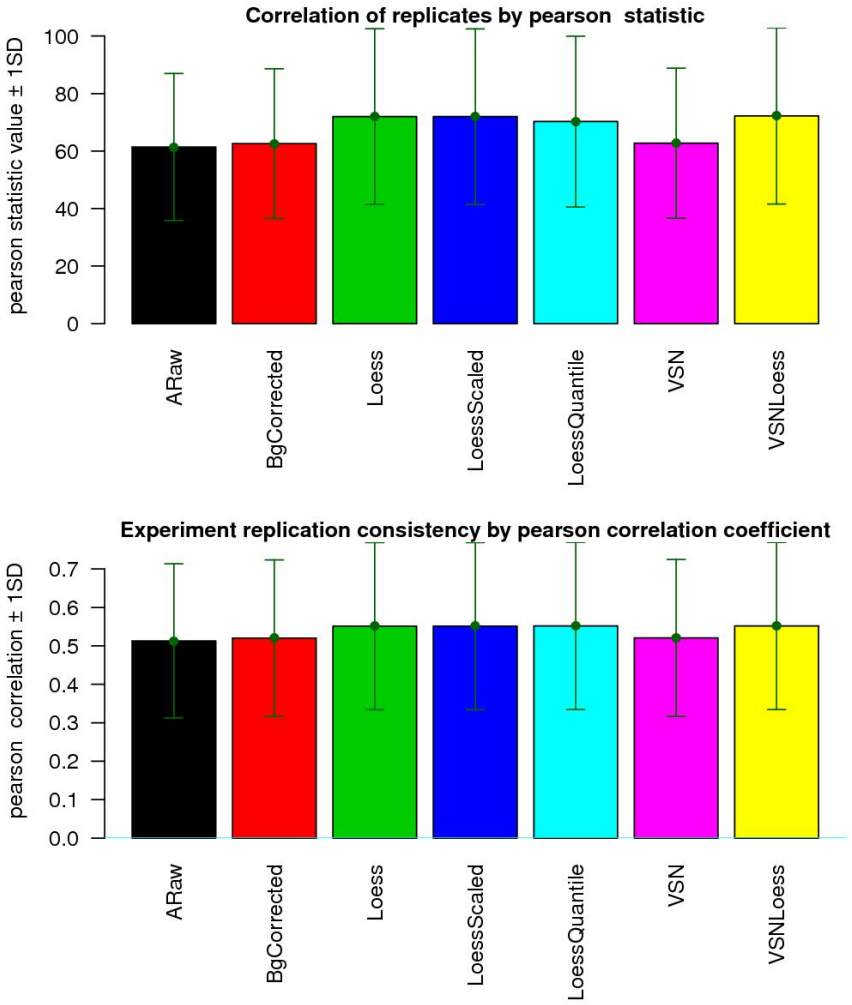


Figure 2.30: Bar plot of the mean Pearson statistic (top) and Pearson *r* correlation (bottom) between pairs of slides for your data set. Larger values indicate high correlation across slides. Top and bottom images are equivalent, but differences between methods are more clear with the Pearson statistic. [image `EvalNormBypearson01.jpg`]

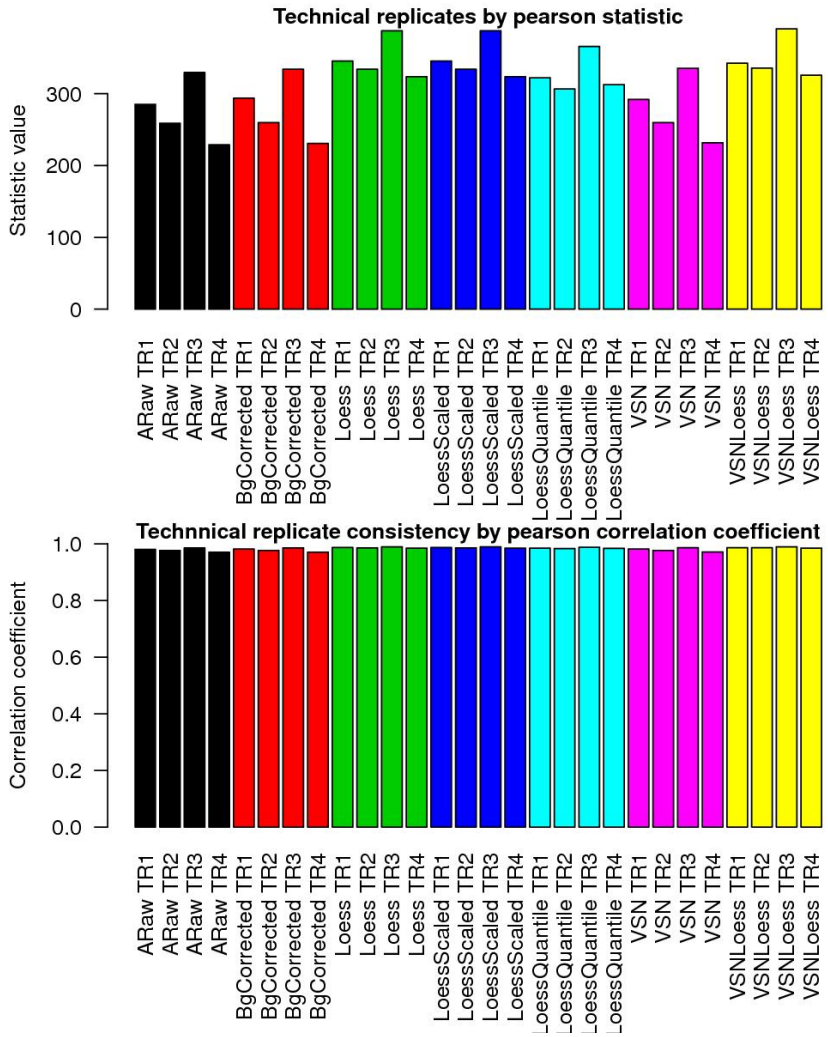


Figure 2.31: Bar plot of the mean Pearson statistic (top) and Pearson r correlation (bottom) of the technical replicates for your data set. Larger values indicate high correlation across slides. Top and bottom images are equivalent, but differences between methods are more clear with the Pearson statistic. [image `TrechRepCorrBypearson01.jpg`]

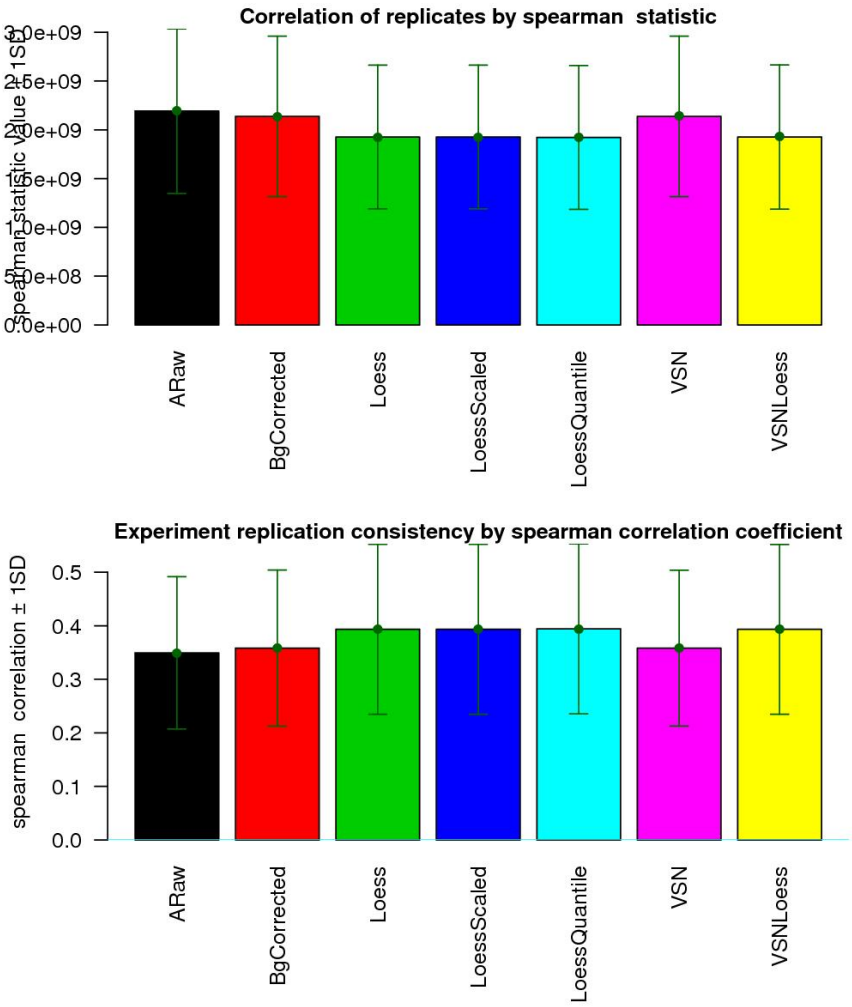


Figure 2.32: Bar plot of the mean Spearman statistic (top) and Spearman ρ correlation (bottom) between pairs of slides for your data set. Larger values of ρ indicate high correlation across slides, as well as lower values of the Spearman statistic. Top and bottom images are equivalent and specular, but differences between methods are clearer with the Spearman statistic. [image `EvalNormBypearson01.jpg`]

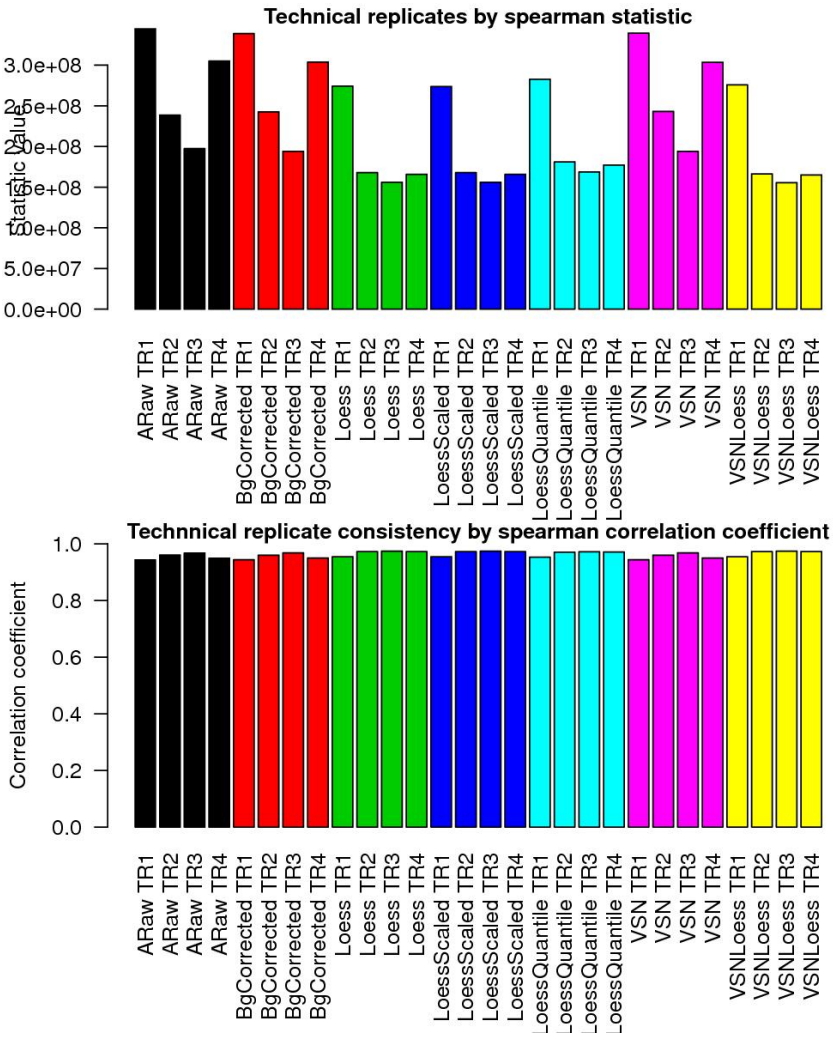


Figure 2.33: Bar plot of the mean Spearman statistic (top) and Spearman ρ correlation (bottom) of the technical replicates for your data set. Larger values of ρ indicate high correlation across slides, as well as lower values of the Spearman statistic. Top and bottom images are equivalent and specular, but differences between technical replicates are clearer with the Spearman statistic. [image `TrechRepCorrBypearson01.jpg`]

Taking into account variability, Kolmogorov-Smirnov test, and Spearman and Pearson correlations, normalization methods that perform best are in all analyses are extracted.

Best methods after all filtering criteria are: LoessScaled

2.6 Best normalization methods

Let's see the main characteristics of normalization methods that will be used for differentially expression analysis. Please, note that it is preferable to use only one normalization adjustment when possible to avoid the loss of biological information. Moreover, hereinafter, duplicate or replicate spots are averaged as a single one.

The methods to characterize are:

Normalisation method: **LoessScaled**

You have 3214 unique spots out of 3584 (total number of features) for every method.

Finally 2321 spots were selected for DEG finding.

2.6.1 Ranked images of microarrays

Let's see the M and A images of data normalized with the previous methods in order to see if spatial artifacts have disappeared. Take into account that

- **M values** are presented as **ranked** values of red signal divided by the green signal in order to highlighting spatial patterns. Images will appear in red and green
- **A values** are also **ranked** also in order to highlighting spatial patterns. White means high expression and black low expression. You should realize that `loess` normalization methods do not change A values and images should be identical to the one observed in figure 2.7. Only `VSN`-normalized data can display a different pattern for A values

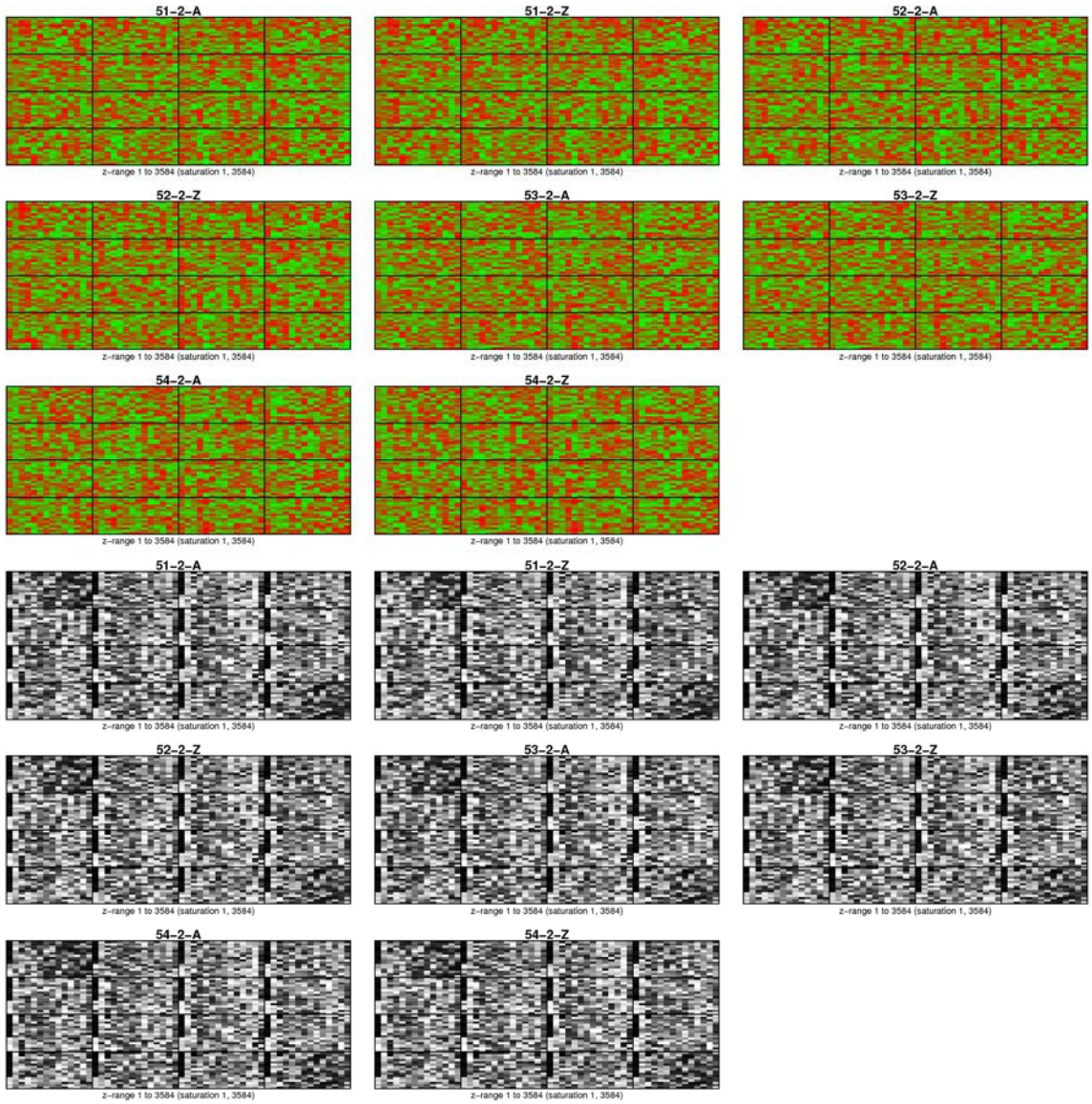


Figure 2.34: Ranked M (top) and A (bottom) images for `LoessScaled`-normalized data [images `LoessScaledMAImageM01.jpg` and `LoessScaledMAImageA01.jpg`]

2.6.2 MA-plots

Normalized data in MA-plots in following figures must be centered around M = 0 without any trend. You can also appreciate if your identified spots have significantly changed their position with respect of figure 2.10.

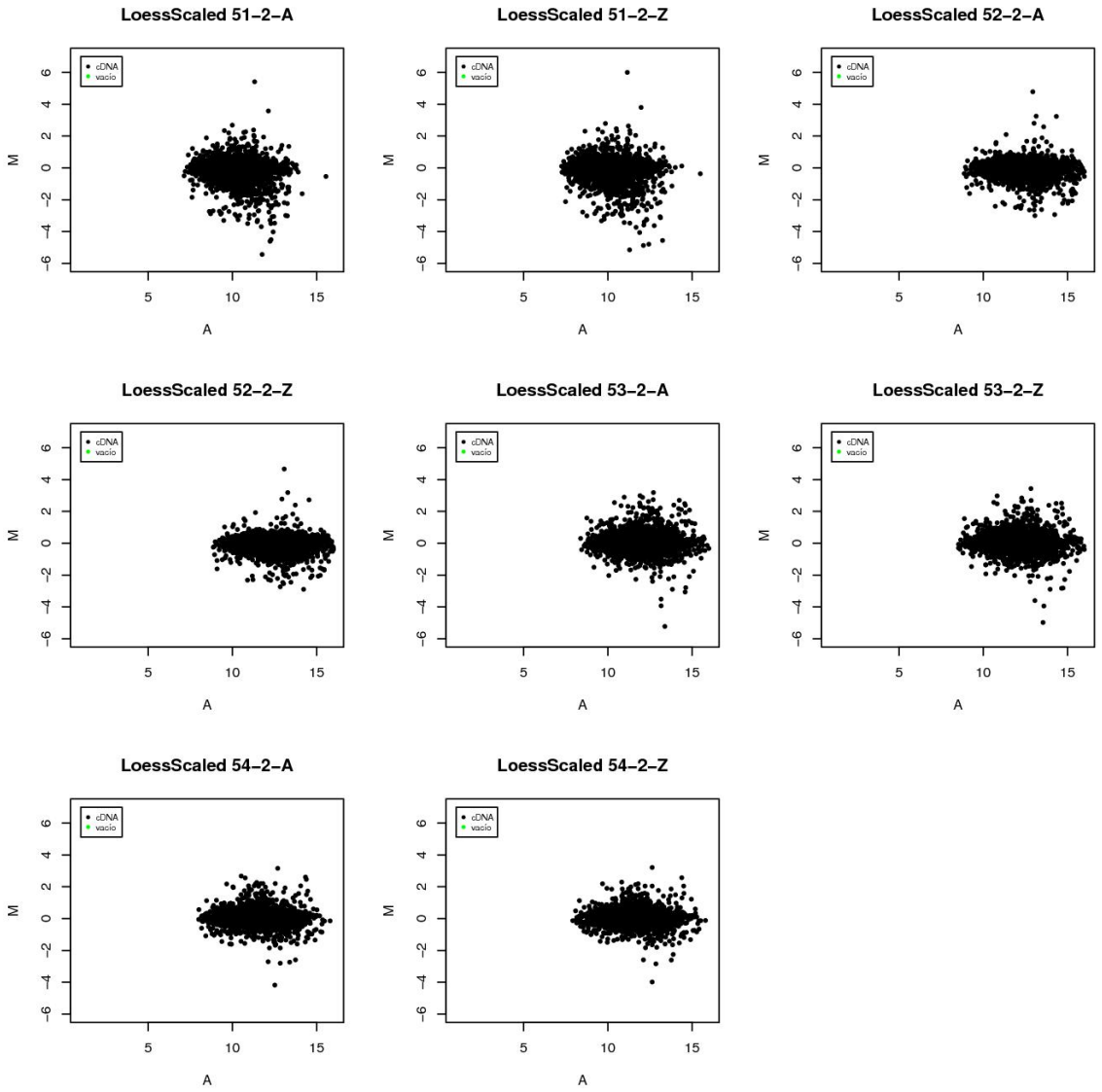


Figure 2.35: MA-plot of `LoessScaled`-normalized data. Bad spots are not plotted. [image `LoessScaledMAPlot01.jpg`]

When the `lowess` adjustment is displayed on MA-plots as those shown before, all of them must coincide with M = 0, otherwise the normalization has not corrected spatial and dye artifacts. Note that in contrast to previous ones, the following spots contain the microarray bad spots.

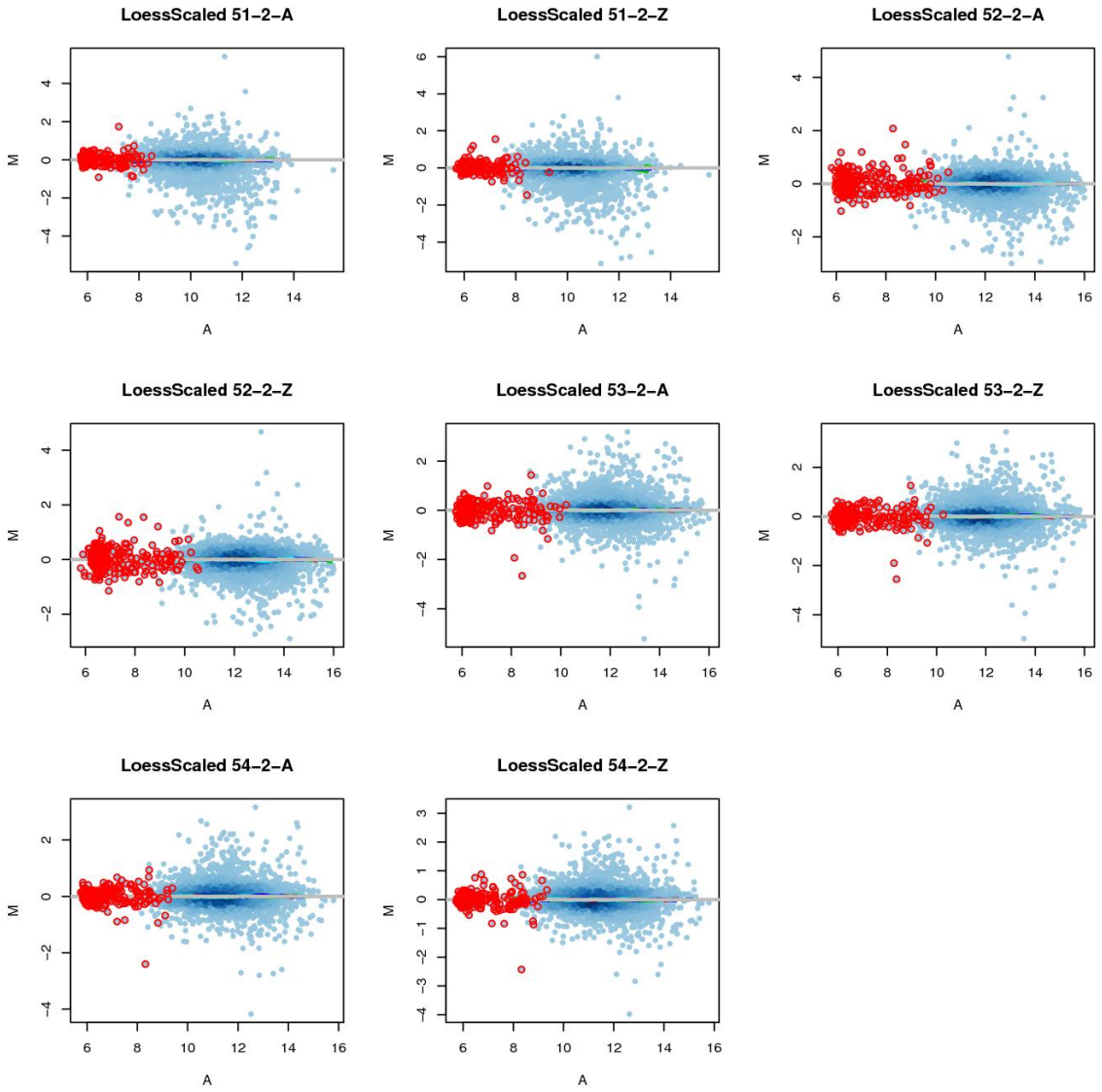


Figure 2.36: MA-plot of `LoessScaled`-normalized data and the textttlowess adjustment of every block in the chip. The points have been colored to illustrate where most of of them are concentrated. Bad spots are plotted and circled in red. [image `LoessScaledMAPlotLwAdj01.jpg`]

Chapter 3

Differential expression with moderated t-test

Differentially expressed genes (DEG) will be identified by a combination of linear models, moderated t-test and empirical bayesian adjustment with the `limma` package using the normalized data that were shown in section 2.6, that is:

Normalisation method: **LoessScaled**

You have 3214 unique spots out of 3584 (total number of features) for every method.

Finally 2321 spots were selected for DEG finding.

Moderated *t*-test instead is preferred instead of a Student *t*-test when the sample size is small, as use to occur in microarray experiments. The empirical Bayes moderation is quite useful in cases with fewer replicates (less than 3) where a regular *t*-test will spread *P*-values too much.

3.1 P-value distribution

The shape of *P*-value frequency for your genes, by means of an histogram, can be used to asses the quality and realiability of your differentially expressed genes in relation to the rest of the genes:

- If you find a peak at $P < 0.001$ and the remainder *P*-values have a low and constant frequency, your differentially expressed genes are reliable
- If you do not have a peak at $P < 0.001$, your analysis will lack of power to detect differentially expressed genes
- If distribution of *P* values when $P > 0.1$ is not fairly uniform, this indicates a overdispersion of data and/or a strong interfering effect of another variable that is not your experimental condition.
- Adjusted *P* values must also present a peak at $P < 0.01$ to assure the power of your analysis.

Distribution of *P* values and adjusted *P* values are shown in the following figures for the normalization methods considered in this analysis.

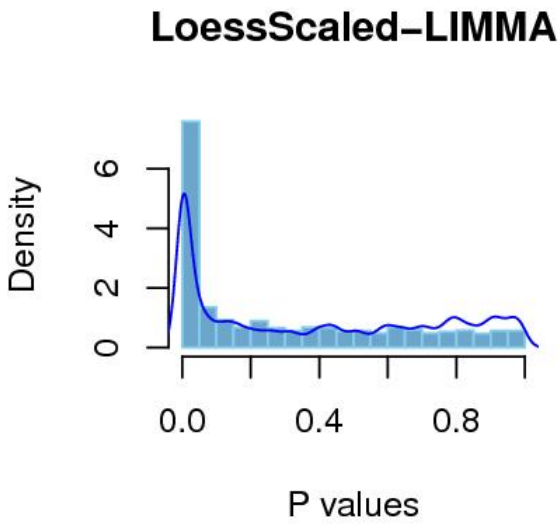


Figure 3.1: Distribution of P-values (histogram) and adjusted P-values (blue line) of your data with the different candidate normalization methods [image `PvaluesDistribution-LIMMA01.jpg`]

3.2 QQ, vulcano and MA plots of DEGs

The following figure allows you to evaluate the normality and distribution of microarray spots, highlighting in red the differentially expressed genes.

QQ plots Representation of theoretical quantiles versus Student's t values (first QQ-plot) or the M values (second QQ-plot) serve to see that this statistic can be preferably used as a correct normalization parameter of spots since red and green lines are almost overlapping.

MA plot MA plot of the normalized averaged values of M and A after analysis is showing that data are correctly normalized. Cyan lines show the cutoffs for considering differentially expressed genes only by the M (logFC) values. This criterium is not accurate since you can see blue points among the red points.

Vulcano Vertical cyan lines are cutoffs for log ratios as in MA plots. An additional cutoff is included considering the P value of each gene, presented as a horizontal red line for adjusted P-value cutoff, and a horizontal gray line for the unadjusted P-value selected by user.

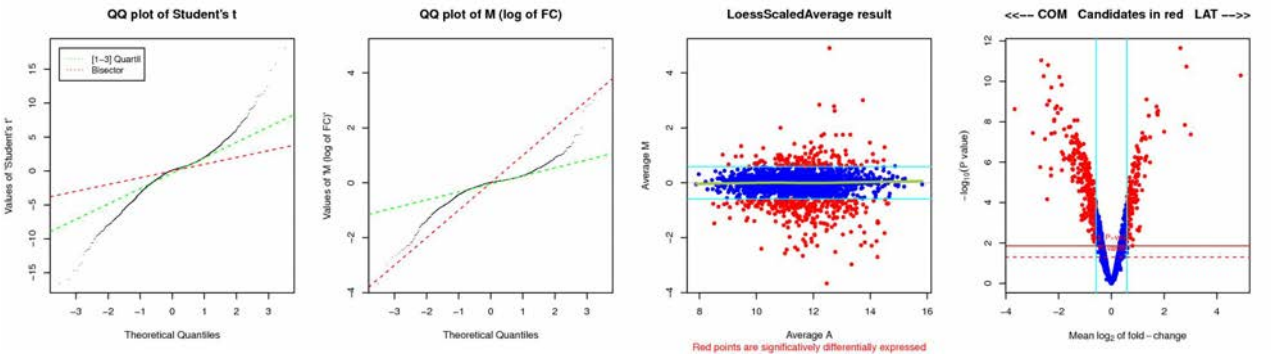


Figure 3.2: Plots of differential expression results from `LoessScaled`-normalized data. [image `LoessScaled-LIMMA-QQMAVolc01.jpg`]

Now, you will see your control spots and differentially expressed genes in MA- and vulcano-plots for every normalization method. You should verify that your controls are in the right side (plots on the left) and if controls are candidate genes (plots on the right). You will also observe the difference between considering the original or adjusted *P*-values.

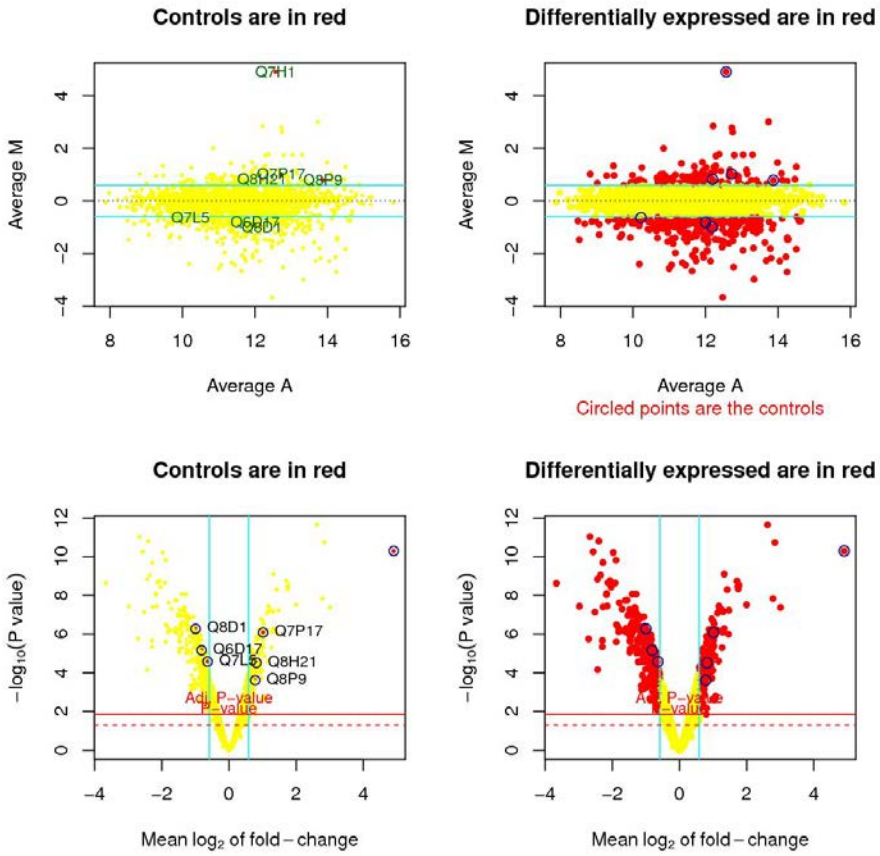


Figure 3.3: Plots of differential expression results from `LoessScaled`-normalized data [images `LoessScaledMAPlotControls01.jpg` and `LoessScaledVulcanoControls01.jpg`]

3.3 Differentially expressed gene lists

Now you have the summary of the differentially expressed genes (DEGs) considering every normalization method by means of Venn diagrams:

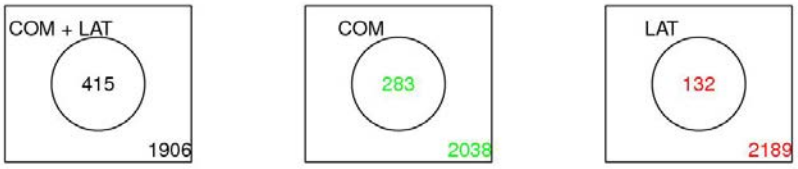


Figure 3.4: DEG for `LoessScaled`-normalized data. [image `LoessScaledVenn01.jpg`]

DEGs are saved into the `Results` folder of your project directory. There is a collection of files by each normalization method. File names start by the normalization method and continue with an explicative suffix whose meaning is the following:

- **_LIMMA_All_genes_data.html**: A HTML file containing a table with data for all genes
- **_LIMMA_All_genes_data.txt**: A tab-delimited file containing a table with data for all genes
- **_LIMMA_DiffExpressed_genes_data.html**: A HTML file containing a table with data for your differentially expressed genes
- **_LIMMA_DiffExpressed_genes_data.txt**: A tab-delimited file containing a table with data for your differentially expressed genes
- **_LIMMA_≪CTRL≫-down.txt**: A list of gene (spot) names of the genes that are over-expressed in the control condition (≪CTRL≫) of your experiment
- **_LIMMA_≪TREAT≫-up.txt**: A list of gene (spot) names of the genes that are over-expressed in the treatment condition (≪TREAT≫) of your experiment

HTML files should be opened with any browser, but tab-delimited files can be opened with a text editor or a spreadsheet provided that it recognizes dots (.) as decimal indicator.

Other saved files

- **_maNorm.data**: The normalized `marrayNorm` object that can be read with `dget()` for further processing
- **_Normalized_data.txt**: A tab-delimited file containing a table with normalized data from all genes of your experiment
- **_t_ord.txt**: A tab-delimited file containing a table with the name of all genes ordered by the *t* statistic. It is useful for further processing with FatiScan together with your annotation file.

Chapter 4

Differential expression with ranks

It has been recently shown that rank product method for detecting DEG, in general, has higher sensitivity and selectivity than the t-based method in both individual and meta-analysis, especially in the setting of small sample size and/or large between-study variation [4]. Rank products are more robust in gene ranking, which leads to a much higher reproducibility among independent studies. Though t-based meta-analysis greatly improves over the individual analysis, it suffers from a potentially large amount of false positives when P-values serve as threshold.

Plots of ranked differential expression for normalized data are presented in the following figure:

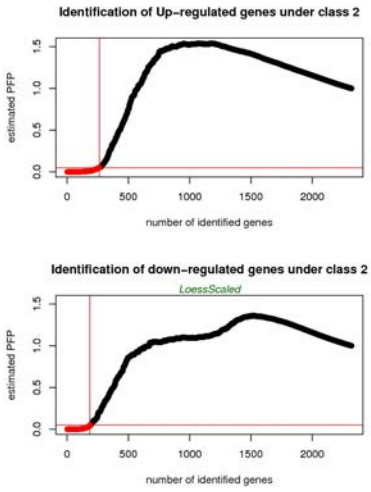


Figure 4.1: Plots of differential expression results with rank products. [images ≪NormMethod≫-RankProduct-plot01.jpg]

4.1 P-value distribution

The shape of *P*-value frequency for your genes, by means of an histogram, can be used to asses the quality and reliability of your differentially expressed genes in relation to the rest of the genes:

- If you find a peak at $P < 0.001$ and the remainder *P*-values have a low and constant frequency, your differentially expressed genes are reliable

- If you do not have a peak at $P < 0.001$, your analysis will lack of power to detect differentially expressed genes
- If distribution of *P* values when $P > 0.1$ is not fairly uniform, this indicates a overdispersion of data and/or a strong interfering effect of another variable that is not your experimental condition.
- Adjusted *P* values must also present a peak at $P < 0.01$ to assure the power of your analysis.

Distribution of *P* values and adjusted *P* values are shown in the following figures for the normalization methods considered in this analysis.

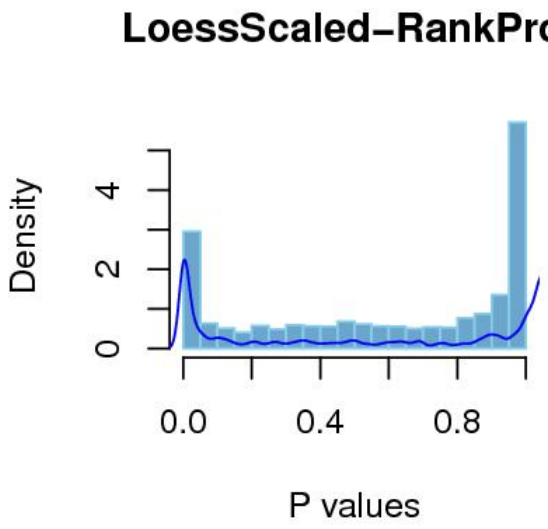


Figure 4.2: Distribution of P-values (histogram) and adjusted P-values (blue line) of your data with the different candidate normalization methods [image `PvaluesDistribution-RANKS01.jpg`]

4.2 Plots of differentially expressed genes

The following figures allows you to evaluate the normality and distribution of microarray spots, highlighting in red the differentially expressed genes. Please, note that the amplitude of M values is larger with these ranked data than with limma analyzed data.

QQ plots Representation of theoretical quantiles versus the M values (log ratios) serve to see that this parameter is normalizated, but since red and green lines are not nearly overlapping, it is not a good estimator for further analysis.

Vulcano Vertical cyan lines are cutoffs for log ratios as in MA plots. An additional cutoff is included considering the *P* value of each gene, presented as a horizontal red line for adjusted P-value cutoff, and a horizontal gray line for the unadjusted P-value selected by user.

- Another vulcano plot can be seen only if you have identified some control spots.

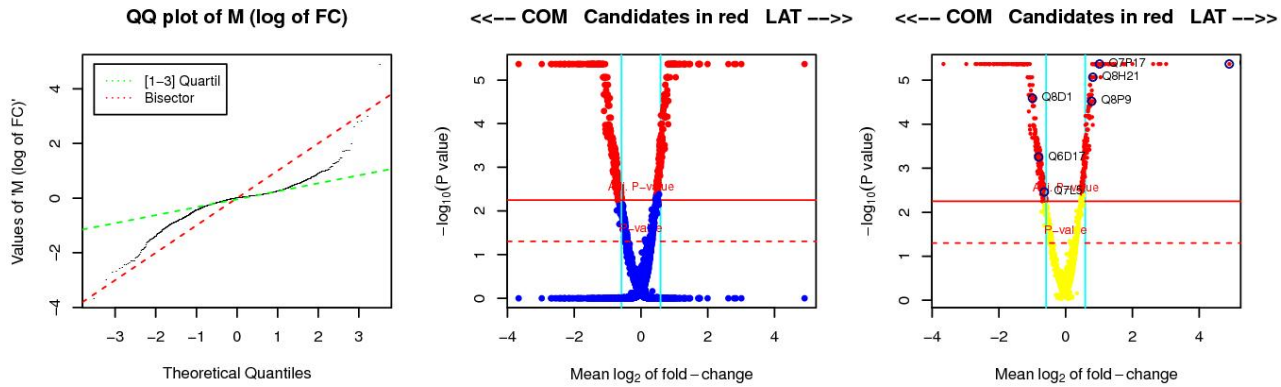


Figure 4.3: Plots of differential expression results from `LoessScaled`-normalized data. [image `LoessScaled-RankProduct-QQvolc01.jpg`]

4.3 Differentially expressed gene lists

DEGs by ranks are saved into the `Results` folder of your project directory. There is a collection of files by each normalization method. File names start by the normalization method and continue with an explicative suffix whose meaning is the following:

- **-RankProduct-≪CTRL≫-down.txt**: A list of gene (spot) names of the genes that are over-expressed in the control condition (≪CTRL≫) of your experiment
- **-RankProduct-≪TREAT≫-up.txt**: A list of gene (spot) names of the genes that are over-expressed in the treatment condition (≪TREAT≫) of your experiment
- **-RankProduct-≪CTRL≫-down_All_Data.txt**: A tab delimited file containing a table with data for your differentially expressed genes in the control condition (≪CTRL≫) of your experiment
- **-RankProduct-≪TREAT≫-up_All_Data.txt**: A tab delimited file containing a table with data for your differentially expressed genes in the control condition (≪TREAT≫) of your experiment

Chapter 5

Consensus of differentially expressed genes

5.1 Lists of consensus DEGs

Here you will find the genes that are expressed considering at the same time the normalization methods and the method for detecting differential expression.

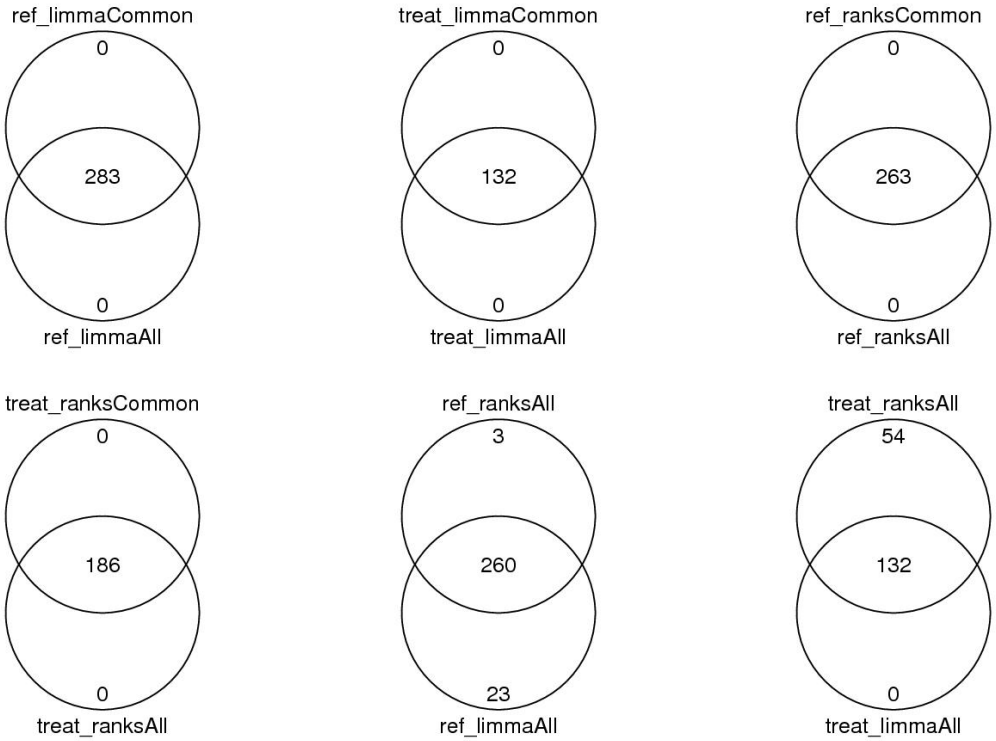


Figure 5.1: Venn diagrams of DEG coincidence between limma and rank product methods for detecting differentially expressed genes. [images `VennResults-plotXX.jpg`]

Differentially expressed genes with moderated t-test (`limma`)

COM (reference) in at least one normalization method:

Q7L17, Q8D1, Q7L5, Q6D17, Q5D17, Q5L21, Q5H1, Q5H5, Q5H13, Q4D17, Q4H5, Q2D9, Q2L9, ...

COM (reference) in **ALL** normalization methods:

Q7L17, Q8D1, Q7L5, Q6D17, Q5D17, Q5L21, Q5H1, Q5H5, Q5H13, Q4D17, Q4H5, Q2D9, Q2L9, Q1L17, Q1D21, Q2L1, Q1D1, Q1P1, ...

LAT (treatment) in at least one normalization method:

Q8P9, Q8H21, Q7P17, Q7D1, Q7H1, Q5P21, Q6H1, Q3P21, Q4P1, Q3H1, Q2H1, Q1L13, Q7K17, Q7K21, Q8O1, Q6K1, Q5C1, Q4C1, ...

LAT (treatment) in **ALL** normalization methods:

Q8P9, Q8H21, Q7P17, Q7D1, Q7H1, Q5P21, Q6H1, Q3P21, Q4P1, Q3H1, Q2H1, Q1L13, Q7K17, Q7K21, Q8O1, Q6K1, Q5C1, Q4C1, Q2G5,...

Differentially expressed genes with rank products

COM (reference) in at least one normalization method:

Q9G4, Q1O16, Q1B1, Q4I21, Q1A1, Q1N21, Q4D3, Q6E12, Q6D4, Q4A13, Q4O4, Q1C1, Q5P12, Q8G19, Q7L3, Q5N23, Q1B2, Q8F1, Q1D1, ...

COM (reference) in **ALL** normalization methods:

Q9G4, Q1O16, Q1B1, Q4I21, Q1A1, Q1N21, Q4D3, Q6E12, Q6D4, Q4A13, Q4O4, Q1C1, Q5P12, Q8G19, Q7L3, Q5N23, Q1B2, Q8F1, Q1D1, ...

LAT (treatment) in at least one normalization method:

Q7H1, Q4D15, Q8M12, Q1L6, Q5B6, Q4O12, Q5M2, Q3H1, Q2B9, Q1M2, Q7J5, Q2O10, Q6H10, Q5I15, Q5A13, Q6O20, Q4F13, Q5A19, Q2H1, ...

LAT (treatment) in **ALL** normalization methods:

Q7H1, Q4D15, Q8M12, Q1L6, Q5B6, Q4O12, Q5M2, Q3H1, Q2B9, Q1M2, Q7J5, Q2O10, Q6H10, Q5I15, Q5A13, Q6O20, Q4F13, Q5A19, Q2H1, ...

Differentially expressed genes appearing in both moderated t-test and rank products

COM (reference) appearing at least once:

Q7L17, Q8D1, Q7L5, Q6D17, Q5D17, Q5L21, Q5H1, Q5H5, Q5H13, Q4D17, Q4H5, Q2D9, Q2L9, Q1L17, Q1D21, Q2L1, Q1D1, Q1P1, Q1D13, ...

COM (reference) appearing in **ALL** methods:

Q7L17, Q8D1, Q7L5, Q6D17, Q5D17, Q5L21, Q5H1, Q5H5, Q5H13, Q4H5, Q2L9, Q1L17, Q1D21, Q2L1, Q1D1, Q1P1, Q1D13, Q7G17, Q8K5, ...

LAT (treatment) appearing at least once:

Q8P9, Q8H21, Q7P17, Q7D1, Q7H1, Q5P21, Q6H1, Q3P21, Q4P1, Q3H1, Q2H1, Q1L13, Q7K17, Q7K21, Q8O1, Q6K1, Q5C1, Q4C1, Q2G5, Q1G5, ...

LAT (treatment) appearing in **ALL** methods:

Q8P9, Q8H21, Q7P17, Q7D1, Q7H1, Q5P21, Q6H1, Q3P21, Q4P1, Q3H1, Q2H1, Q1L13, Q7K17, Q7K21, Q8O1, Q6K1, Q5C1, Q4C1, Q2G5, Q1G5, ...

5.2 Heat maps of DEGs

You can also see a heat map of DEGs of every normalization method:

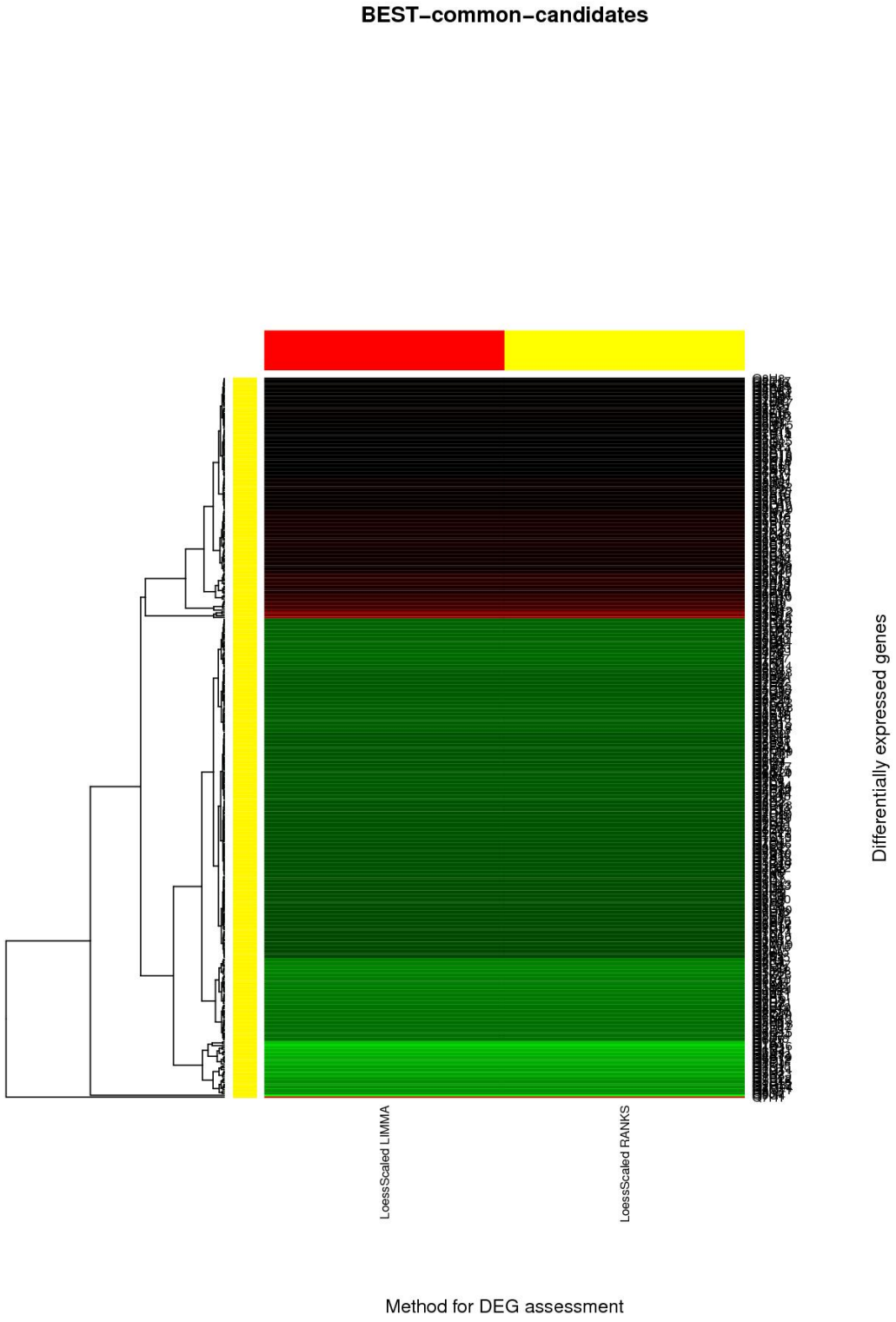


Figure 5.2: HeatMap of **BEST-common-candidates** of differentially expressed genes (DEGs) between all normalized method and both moderated t-test and rank products. [image `HeatMap-DEG-BEST-common-candidates01.jpg`]

Bibliography

[1] Bolstad et al. A comparison of normalization methods for high density oligonucleotide array data based on variance and biass. *Bioinformatics* (2003) vol. 19 pp. 185-193

[2] Chiogna et al. A comparison on effects of normalisations in the detection of differentially expressed genes. *BMC Bioinformatics* (2009) vol. 10 pp. 61

[3] Harr & Schlötterer. Comparison of algorithms for the analysis of Affymetrix microarray data as evaluated by co-expression of genes in known operons. *Nucleic Acids Res.* (2006) vol. 34 (2) pp. e8

[4] Hong & Breitling. A comparison of meta-analysis methods for detecting differentially expressed genes in microarray experiments. Bioinformatics (2008) vol. 24 (3) pp. 374-82

[5] Kim et al. Spearman's footrule as a measure of cDNA microarray reproducibility. *Genomics* (2004) vol. 84 (2) pp. 441-8

[6] Lut et al. Can Zipf's law be adapted to mormalize microarrays? *BMC Bioinformatics* (2005) vol. 6, pp. 37

[7] Martin-Requena et al. PreP+07: improvements of a user friendly tool to preprocess and analyse microarray data. *BMC Bioinformatics* (2009) vol. 10 pp. 16

[8] Quim et al. Empipirical evaluation of data transformations and ranking statistics for microarray analysis. *Nucleic Acids Res* (2004) 32, 5471-5479

[9] Ritchie et al. A comparison of background correction methods for two-colour microarrays. *Bioinformatics* (2007) vol. 23 (20) pp. 2700-7

[10] Xiong et al. Using generalized procrustes analysis (GPA) for normalization of cDNA microarray data. *BMC Bioinformatics* (2008) vol. 9 pp. 25

Thanks you for use MADE-4-2Colors!

Send us any coment to Noé Fernández Pozo

Apéndice C

Colaboración para analizar micromatrices de expresión

En colaboración con el grupo de investigación de Genómica y Mejora Animal del Departamento de Genética de la Facultad de Veterinaria de la Universidad de Córdoba se publicó el siguiente artículo, en el que colaboré anotando y organizando la información de las secuencias que tenían las sondas impresas en la micromatriz de *Affimetrix* de cerdo que se utilizó.

Sanz-Santos et al. BMC Proceedings 2011, 5(Suppl 4):S11
http://www.biomedcentral.com/1753-6561/5/S4/S11

BMC Proceedings

PROCEEDINGS Open Access

Gene expression pattern in swine neutrophils after lipopolysaccharide exposure: a time course comparison

Gema Sanz-Santos[1], Ángeles Jiménez-Marín[1], Rocío Bautista[2], Noé Fernández[2], Gonzalo M Claros[2], Juan J Garrido[1*]

From International Symposium on Animal Genomics for Animal Health (AGAH 2010)
Paris, France. 31 May – 2 June 2010

Abstract

Background: Experimental exposure of swine neutrophils to bacterial lipopolysaccharide (LPS) represents a model to study the innate immune response during bacterial infection. Neutrophils can effectively limit the infection by secreting lipid mediators, antimicrobial molecules and a combination of reactive oxygen species (ROS) without new synthesis of proteins. However, it is known that neutrophils can modify the gene expression after LPS exposure. We performed microarray gene expression analysis in order to elucidate the less known transcriptional response of neutrophils during infection.

Methods: Blood samples were collected from four healthy Iberian pigs and neutrophils were isolated and incubated during 6, 9 and 18 hrs in presence or absence of lipopolysaccharide (LPS) from *Salmonella enterica* serovar Typhimurium. RNA was isolated and hybridized to Affymetrix Porcine GeneChip®. Microarray data were normalized using Robust Microarray Analysis (RMA) and then, differential expression was obtained by an analysis of variance (ANOVA).

Results: ANOVA data analysis showed that the number of differentially expressed genes (DEG) after LPS treatment vary with time. The highest transcriptional response occurred at 9 hr post LPS stimulation with 1494 DEG whereas at 6 and 18 hr showed 125 and 108 DEG, respectively. Three different gene expression tendencies were observed: genes in cluster 1 showed a tendency toward up-regulation; cluster 2 genes showing a tendency for down-regulation at 9 hr; and cluster 3 genes were up-regulated at 9 hr post LPS stimulation. Ingenuity Pathway Analysis revealed a delay of neutrophil apoptosis at 9 hr. Many genes controlling biological functions were altered with time including those controlling metabolism and cell organization, ubiquitination, adhesion, movement or inflammatory response.

Conclusions: LPS stimulation alters the transcriptional pattern in neutrophils and the present results show that the robust transcriptional potential of neutrophils under infection conditions, indicating that active regulation of gene expression plays a major role in the neutrophil-mediated- innate immune response.

Background

Neutrophils play a key role in innate immune response. They initiate phagocytosis, degranulation and killing without new synthesis of proteins. However, it has been demonstrated that new gene transcription and protein synthesis are required to maintain full capacity for human neutrophil phagocytosis and associated bactericidal activity [1,2].

LPS treatment enhances neutrophil bactericidal activity, with an alteration in adhesion, respiratory burst, degranulation and motility [3,4]. Thus kinetic study of swine neutrophil response to LPS represents an in vitro model to investigate the innate immune response during bacterial infection.

To test the neutrophil transcriptional potential, global gene expression analysis was performed and the results indicated that the LPS-treated neutrophils increase their

* Correspondence: ge1gapaj@uco.es
[1]Grupo de Genómica y Mejora Animal, Departamento de Genética, Facultad de Veterinaria, Universidad de Córdoba, Campus de Rabanales, Edificio Gregor Mendel C5, 14071 Córdoba, Spain
Full list of author information is available at the end of the article

Apéndice D

Fichero de configuración de MADE4-2C

```
1  ##############################################################################
2  ##### MADE-4-2C: MicroArray Differential Expression For 2-Colors #######
3  #####                                                                 #######
4  ############## Noe Fernandez Pozo & M. Gonzalo Claros   ################
5  ##############################################################################
6
7  # clear the work space
8  rm(list=ls())
9
10 # This is the configuration file for MADE-4-2C
11 # All parameters and variables must be customized for each analysis
12 # This file can be place anywhere
13
14 ########################################
15 # Global variables defined in this file
16 #
17 # uUserName
18 # uProject
19 # uProjectInfo
20 # uSourceDir
21 # uWD
22 # uTargets
23 # uMaType
24 # uGalFile
25 # uSpotType
26 # uBadSpots
27 # uControlSpots
28 # uAbsences
29 # uFC
30 # uMinFC
31 # uPval
32 # uMethod
33 # uGraphOutput
34 # uQuick
35 # gOrigWD
36
37 ##############################################################################
38 #### DEFINE USER NAME AND PROJECT NAME
39 #
40 # You should include here your name (for the final report)
41 # and a project name to store all results
42 # Example:
43
44 uUserName = "Noe"
45 uProject  = "Swirl"
46
47 # You can also provide some informations about your microarray project
48 uProjectInfo = "esto es una prueba"
49
50 ##############################################################################
51 #### DEFINE WHERE ARE THE MADE-4-2C FILES
52 #
53 # You should include here the path where "MADE-4-2C_files" can be found
54 # on your computer
55 # Example:
```

```
56      uSourceDir = "~/usr/local/MADE-4-2C/MADE-4-2C_files/"
57
58  ##############################################################################
59  #### DEFINE YOUR WORKING DIRECTORY
60  #
61  # You should include here the path where this file is on your computer
62  # Example:
63      uWD = "~/Documents/My_MA_data/this_experiment/"
64
65  ##############################################################################
66  #### LOAD YOUR "TARGETS" FILE
67  #
68  # "Targets" file serves to define your experiment identifiers, the files used
69  # and the experimental design
70  #
71  # "Targets file can have any name and MUST be located in the directory you have
72  # indicated in the variable uWD"
73  #
74  # Files containing microarray data must be in GPR or Spot format
75  #
76  # "Targets" is a text tabulated file that MUST contain columns named as:
77  # Label, SlideNumber, FileName, Cy3, Cy5, repBiol
78  #
79  # Example of "targets" file:
80  #
81  # Label        SlideNumber  FileName                Cy3   Cy5   repBiol
82  # 50Baja-A     50a-A        dir1/finename1-A.gpr    CTRL  TREAT  1
83  # 50Baja-Z     50b-Z        dir1/finename2-Z.gpr    CTRL  TREAT  1
84  # 51Baja-A     51b-A        dir1/finename3-A.gpr    TREAT CTRL   2
85  # 51Baja-Z     51b-Z        dir1/finename4-Z.gpr    TREAT CTRL   2
86
87  # uTargets = "targets_AB.txt"
88   uTargets = "SwirlSample.txt"
89
90  ##############################################################################
91  #### TYPE OF IMAGE ANALYSIS PROGRAM
92  #
93  # Microarray data can be obtained by means of several analysis
94  # programs. You can select from:
95  #
96  # "gp" for GenePix (file extension .gpr)
97  # "sp" for Spot (file extension .spot)
98
99  # Example:
100 # uMaType = "gp"
101  uMaType = "sp"
102
103 ##############################################################################
104 #### LOAD YOUR "GALFILE" (optional)
105 #
106 # The GalFile can contain more information than your GPR data. Hence, you can
107 # incorporate it here
108 #
109 # This is optional. If you do not want to include it, define it as ""
110 #
111 # The GalFile can have any name and MUST be located in the directory you have
112 # indicated in the variable uWD
113 #
114 # Definition of an empty GalFile:
115
116 # uGalFile = ""
117 # uGalFile = "Pinarray.gal"
118  uGalFile = "fish.gal"
119
120 ##############################################################################
121 #### LOAD YOUR "SpotType" FILE (optional)
122 #
123 # The SptoType contain information to colour specific spots on MA plots
124 #
125 # This is optional. If you do not want to include it, define it as ""
126 #
127 # The SpotType can have any name and MUST be located in the directory you have
128 # indicated in the variable uWD
129 #
130 # Example of a SpotType file:
131 #
132 # SpotType  ID            Name          Color
133 # cDNA      *             *             black
134 # control   control       *             yellow
135 # Norh-Inv  13-7 XL A6    13-7 XL A6    blue
136 # LAT       lateral       *             red
137 # COMP      compres       *             green
```

```
#
# Definition of an empty SpotType file:

# uSpotType = "SpotTypes controles.txt"
 uSpotType = ""

#############################################################################
#### LOAD YOUR "Bad Spots" FILE (optional)
#
# The Bad Spots file contain information about printed spots that are known to
# be uninformative or have been mistakenly printed
#
# This is optional. If you do not want to include it, define it as ""
#
# The Bad Spots can have any name and MUST be located in the directory you have
# indicated in the variable uWD
#
# The Bad Spots is a list of "Names"
#
# Example of a Spot Type file:
#
# Q5H1
# Q4D21
# Q1K13
#
# Definition of an empty SpotType file:

# uBadSpots = "malos.txt"
 uBadSpots = ""

#############################################################################
#### LOAD YOUR IDENTIFIED OR CONTROL SPOTS FILE (optional)
#
# This file is to contain a list of "Names" of spots that you want to
# have localizable on a final vulcano plot report. They can be whatever
# you want provided that the name of each spot is the name corresponding
# to the column "Names" of the array.
#
# Every name must be in a new line.
#
# This is optional. If you do not want to include it, define it as ""
#
# The file can have any name and MUST be located in the directory you have
# indicated in the variable uWD
#
# The Bad Spots is a list of "Names"
#
# Example of a control spots file:
#
# Q5H1
# Q4D21
# Q1K13
#
# Definition of an empty SpotType file:

# uControlSpots = "miscontroles.txt"
 uControlSpots = ""

#############################################################################
#### NUMBER OF SPOT ABSENCES IN YOUR DATA
#
# Good spot data are those that appear in all slides. However, you can
# indicate the number of absences that you permit in your analysis
#
# Note that is should not be higher that the half of the number of slides
# in your analysis
#
# Example:

uAbsences <- 0

#############################################################################
#### DEFINE YOUR FOLD-CHANGE
#
# You must specify the fold-change threshold (in absolute value) for a
# spot signal to be considered as differentially expressed
#
# Remember that a two-fold change is 2X, so an uFC = 2
# Therefore, uFC must be > 1
#
# Example:

```

```
uFC <- 1.5

################################################################################
#### DEFINE YOUR MINIMUM SIGNIFICATIVE FOLD-CHANGE
#
# You must specify a fold-change threshold (in absolute value) that will
# be used for removing spots considered as clearly invariable. A value
# equal to 0 means that all spots will be analysed for differential
# expression
#
# Remember that "1" means that the change is "1X", so, there is no change
# Therefore, uMinFC must be >= 1
#
# Example:

uMinFC <- 1.1

################################################################################
#### DEFINE YOUR P-VALUE
#
# You must specify the P-value threshold for a spot signal to be considered as
# significative
#
# Example:
# uPval <- 0.01
uPval <- 0.05

################################################################################
#### DEFINE METHOD TO COMPENSATE MULTITESTING
#
# You must specify the statistical method to use for false discovery
# correction
#
# "BH" for 'Benjamini y Hochberg' (1995) for usual FDR
# "bonferroni" for the most restrictive multiple testing correction based
#              on family wise error rate (FWER)
# "holm" for a restrictive FWER that outperforms the one of Benjamini
# Example:

uMethod <- "BH"

################################################################################
#### DO YOU WANT GRAPHICAL OUTPUTS?
#
# This script can provide a lot of graphical outputs concerning your analysis
# Here you decide how to get them (or not)
#
# "screen" if you want every graphical plot in a window
# "jpeg" if you want all graphics in JPEG format
# "pdf" if you want all graphics in a PDF report

uGraphOutput <- "pdf"

################################################################################
#### DO YOU WANT A QUICK ANALYSIS?
#
# If you do not want a final report nor all diagnostic images, but only
# differentially expressed genes, you can speed up the analysis setting
# this variable to true
#
# Please, note that the quick analysis will only display images on
# screen and that no latex report will be generated

uQuick <- FALSE

##############################
## END CONFIGURATION FILE ##
##############################

################################################################################
## DO NOT TOUCH THE FOLLOWING

# obtaining the working dir where you were before starting this script
gOrigWD <- getwd()
# load the true script
main_file <- paste(uSourceDir, "main.R", sep="")
source(main_file)
```

Apéndice E

Script de descarga de contaminantes para SeqTrimNext

Este *script* en Ruby se utiliza para descargar las bases de contaminantes de SeqTrimNext, mencionadas en el apartado 10.1.2, pág. 97.

```ruby
#!/usr/bin/env ruby

# 24-1-2011
# Noe Fernandez Pozo
# Script para actualizar las bases de datos de contaminantes de SeqTrimNext.
# Baja los genomas de bacterias, hongos, humano y organulos

# para ver la ayuda de la gema net/ftp:
# http://www.ensta.fr/~diam/ruby/online/ruby-doc-stdlib/libdoc/net/ftp/rdoc/index.html
# gema para acceder y descargar ficheros via ftp
require 'net/ftp'

# ---------------------------------------------- Funciones
def connect_to_ncbi
 $ftp = Net::FTP.new()

 # creamos los directorios de salida que necesitamos
 if !File.exists?('seqtrim_contaminants')
  Dir.mkdir('seqtrim_contaminants')
 end

 if !File.exists?("seqtrim_contaminants/core_bacteria_fungi")
  Dir.mkdir("seqtrim_contaminants/core_bacteria_fungi")
 end

 if !File.exists?("seqtrim_contaminants/Bacteria")
  Dir.mkdir("seqtrim_contaminants/Bacteria")
 end

 if !File.exists?("seqtrim_contaminants/Fungi")
  Dir.mkdir("seqtrim_contaminants/Fungi")
 end

 # cargamos en memoria la lista de contaminantes comunes
 $model_sps_hash = {}
 File.open('core_seqs.txt').each do |line|
  if (line =~ /Escherichia_coli_K_12_substr__D/)
   $model_sps_hash['Escherichia_coli_K_12_substr__D']='Escherichia_coli_K_12_substr__DH10B'
  elsif (line =~ /((\w+)\_\w+)/)
   ftp_dir_name = $1
   ftp_sps_name = $2
   $model_sps_hash[ftp_sps_name] = ftp_dir_name
  end
 end

 # nos conectamos al repositorio ftp de ncbi descargamos las secuencias
 $ftp.connect('ftp.ncbi.nlm.nih.gov')
 $ftp.login

  download_contaminants('Fungi')
  download_contaminants('Bacteria')
```

```
52      download_organelle('Chloroplasts/plastids/plastids', 'plastids')
53      download_organelle('MITOCHONDRIA/Metazoa', 'mitochondrias')
54      download_human
55
56    $ftp.close
57
58  end
59
60  def download_human
61
62    # vamos al directorio con el genoma humano y recorremos la lista de cromosomas
63    $ftp.chdir("/genomes/H_sapiens")
64    files = $ftp.list
65
66    files.each do |line|
67
68      if (line =~ /(CHR_\w\w*)/)
69        dir_name = $1
70      end
71
72      puts "dir_name: #{dir_name}"
73
74      if !File.exists?("seqtrim_contaminants/#{dir_name}")
75        Dir.mkdir("seqtrim_contaminants/#{dir_name}")
76      end
77
78      fasta_files = $ftp.list("#{dir_name}/*HuRef*.fa.gz")
79
80      # descargamos el fichero comprimido con el fasta de cada cromosoma
81      fasta_files.each do |fasta|
82        fasta =~ /\/([\_\w]+\.fa\.gz)/
83        file_name = $1
84        puts file_name
85        $ftp.getbinaryfile("#{dir_name}/#{file_name}", "seqtrim_contaminants/#{dir_name}/#{
              file_name}")
86      end
87    end
88  end
89
90  def download_organelle(my_path,my_organelle)
91
92    # vamos al directorio del organulo indicado por my_path y descargamos las secuencias
93    $ftp.chdir("/genomes/#{my_path}")
94    files = $ftp.list
95
96    if !File.exists?("seqtrim_contaminants/#{my_organelle}")
97      Dir.mkdir("seqtrim_contaminants/#{my_organelle}")
98    end
99
100   fasta_files = $ftp.list("*.fna")
101
102   # descargamos cada uno de los ficheros de secuencias
103   fasta_files.each do |fasta|
104     fasta =~ /([\_\w]+\.fna)/
105     file_name = $1
106     puts file_name
107     $ftp.getbinaryfile("#{file_name}", "seqtrim_contaminants/#{my_organelle}/#{file_name}")
108   end
109 end
110
111 def download_contaminants(my_path)
112   # para descargar las secuencias de hongos y bacterias segun se indique en my_path
113   $ftp.chdir("/genomes/#{my_path}")
114   files = $ftp.list
115
116   # solo se descarga un representatne de cada genero
117   previous_genus = ''
118
119   # parseamos la lista de contaminantes guardando el genero y el nombre especifico
120   files.each do |line|
121     line.chomp!
122
123     # el parseo es diferente cuando la base de datos comienzan por Candidatus o
          Blattabacterium
124     # es importante asegurarnos de que la cepa de Ecoli que bajamos sea k12dh10b
125     if (line =~ /Candidatus_/)
126       line =~ /(((Candidatus_[A-Z][a-z]+)_[^_]+)_.*)/
127     elsif (line =~ /Blattabacterium__/)
128       line =~ /(((Blattabacterium__[A-Z][a-z]+)_[^_]+)_.*)/
129     elsif (line =~ /Escherichia_coli_K_12_substr__DH10B/)
130       line =~ /(((Escherichia_coli_K_12_substr__D)H10B).*)/
131     else
```

```
132       line =~ /((([A-Z][a-z]+)_[^_]+)_.*)/
133     end
134
135     dir_name = $1
136     bacteria_name = $2
137     genus_name = $3
138
139     if (!$model_sps_hash[genus_name].nil?)
140      if ($model_sps_hash[genus_name] == bacteria_name)
141       # en este bloque descargaremos los contaminantes comunes ----------------------------
142       puts "$model_sps_hash[genus_name]: #{$model_sps_hash[genus_name]}"
143
144       if !File.exists?("seqtrim_contaminants/core_bacteria_fungi/#{$model_sps_hash[genus_name
              ]}")
145        Dir.mkdir("seqtrim_contaminants/core_bacteria_fungi/#{$model_sps_hash[genus_name]}")
146
147        puts "bacteria_name: #{bacteria_name}"
148        fasta_files = $ftp.list("#{dir_name}/*.fna")
149        fasta_files.each do |fasta|
150         fasta =~ /\/([\_\w]+.fna)/
151         file_name = $1
152         puts file_name
153         $ftp.getbinaryfile("#{dir_name}/#{file_name}", "seqtrim_contaminants/
               core_bacteria_fungi/#{$model_sps_hash[genus_name]}/#{file_name}")
154        end
155       end
156       #-------------------------------------------------------------------------------------
157      else
158       next
159      end
160     elsif genus_name != previous_genus
161      # en este bloque descargaremos las secuencias de bacterias y hongos ------------------
162      if !File.exists?("seqtrim_contaminants/#{my_path}/#{dir_name}")
163       Dir.mkdir("seqtrim_contaminants/#{my_path}/#{dir_name}")
164
165       puts "bacteria_name: #{bacteria_name}"
166       if (bacteria_name !~ /^Candidatus_/)
167        fasta_files = $ftp.list("#{dir_name}/*.fna")
168        fasta_files.each do |fasta|
169         fasta =~ /\/([\_\w]+.fna)/
170         file_name = $1
171         puts file_name
172         $ftp.getbinaryfile("#{dir_name}/#{file_name}", "seqtrim_contaminants/#{my_path}/#{
               dir_name}/#{file_name}")
173        end
174       end
175      end
176      #-----------------------------------------------------------------------------------------
177     else
178      puts "Ya tenemos este genero: #{bacteria_name} !!!"
179     end
180     previous_genus = genus_name
181   end
182 end
183
184 def connect_to_silva
185   current_release = ''
186
187   $ftp = Net::FTP.new()
188
189   if !File.exists?('seqtrim_contaminants')
190    Dir.mkdir('seqtrim_contaminants')
191   end
192
193   if !File.exists?("seqtrim_contaminants/rrna")
194    Dir.mkdir("seqtrim_contaminants/rrna")
195   end
196
197   # para conectarse a la base de datos de silva para descargar las secuencias de ARNr
198   $ftp.connect('ftp.arb-silva.de')
199   $ftp.login
200   $ftp.chdir("/current/Exports/")
201
202   files = $ftp.list
203   files.each do |line|
204    line.chomp!
205
206    if (line =~ /LSURef_(\d+)_tax_silva.fasta.tgz/)
207     current_release = $1
208     $ftp.getbinaryfile("LSURef_#{current_release}_tax_silva.fasta.tgz", "
           seqtrim_contaminants/rrna/rrna_lsu_#{current_release}.fasta.tgz")
```

```
209       $ftp.getbinaryfile("SSURef_#{current_release}_tax_silva.fasta.tgz", "
              seqtrim_contaminants/rrna/rrna_ssu_#{current_release}.fasta.tgz")
210     end
211
212    end
213
214    $ftp.close
215
216    return ["rrna_lsu_#{current_release}.fasta.tgz", "rrna_ssu_#{current_release}.fasta.tgz"]
217
218  end
219
220  def make_cdhit_sh_file
221   # creamos un fichero sh para ejecutar cdhit en el sistema de colas
222   # asi se reduce la cantidad de datos
223   sh_file = File.open('seqtrim_contaminants/rrna/rrna_cdhit.sh', 'w')
224
225   sh_file.puts "
226  # numero de cpus que empleara el calculo:
227  #PBS -l ncpus=4
228  # memoria que empleara el calculo:
229  #PBS -l mem=6000mb
230  # cuanto va a tardar el calculo como maximo :
231  #PBS -l walltime=40:00:00
232
233  # para que vaya al directorio actual:
234  cd $PBS_O_WORKDIR
235
236  # inicializa cd-hit
237  . ~cdhit/init_env
238
239  #ejecuta comando
240
241  cd-hit-est -i rrna_lsu.fasta -o rrna_lsu90.fasta -c 0.9 -n 7 -r 1 -T 4
242  cd-hit-est -i rrna_ssu.fasta -o rrna_ssu90.fasta -c 0.9 -n 7 -r 1 -T 4
243   "
244  end
245
246
247  # ------------------------------------------------ Principal
248
249  puts "downloading contaminants\n\n"
250
251  # nos conectamos al repositorio ftp de ncbi descargamos las secuencias contaminantes
252  connect_to_ncbi
253
254  # para descargar los ARNr
255  rrna_files = connect_to_silva
256
257  `tar -xvzOf seqtrim_contaminants/rrna/#{rrna_files[0]} > seqtrim_contaminants/rrna/rrna_lsu
         .fasta`
258  `tar -xvzOf seqtrim_contaminants/rrna/#{rrna_files[1]} > seqtrim_contaminants/rrna/rrna_ssu
         .fasta`
259  `rm seqtrim_contaminants/rrna/*.tgz`
260
261  # creamos un fichero sh para ejecutar cdhit en el sistema de colas
262  # asi se reduce la cantidad de datos
263  make_cdhit_sh_file
264
265  # comando para ejecutar el script de bash en el sistema de colas
266  # `qsub rrna_cd_hit.sh`
```

Apéndice F

Ejemplo de informe del preprocesamiento de SeqTrimNext

Este documento, que genera automáticamente SeqTrimNext se obtuvo al preprocesar el primer conjunto de secuencias de 454 del grupo BMBP que se menciona en el capítulo 11 de esta tesis.

SeqTrimNext
Statistics of pre-processing

Plataforma Andaluza de Bioinformática
Universidad de Málaga

June 22, 2011

1 Output Files

SeqTrimNext provides several files, the most interesting ones are in the following directories:

- output_files
  - `output.less`, containing an extensive information about the trimming of each sequence. It can be visualised on terminal using the command `less -R`.
  - `used_params.txt`, containing the complete set of parameters used for execution of SeqTrimNext with your data
  - `rejected.txt`, containing a list of rejected sequences together with the reason for their removal.
  - `initial_stats.json`, containing statistics for raw sequences.
  - `stats.json`, containing the statistics of the cleaning process.
  - There is a collection of `folders` that gather sequences with the same MID; each folder contains a `sequences` file (in FastQ format) with useful reads. There may also exists a file with reads containing low complexity regions. If you want to reconstruct a SFF with the useful segment of each pre-processed read, use `sff_info` file in combination with the original SFF file for the `sfffile` tool.
- graphs
  - `size_stats.png`, a graph with the distribution of read lengths in raw data (see Fig. 1).
  - `qualities.png`, a graph to inspect read qualities in raw data (see Fig. 2).
  - `PluginExtractInserts_insert_size.png`, a graph with the distribution of read lengths after SeqTrimNext pre-processing (see Fig. 3).
  - There are other graphs (mostly bar plots) that illustrate the quality of pre-processed reads. All are in PNG format.
- latex
  - It is provided as a compressed file `latex.zip` containing all ".tex" files required to compile this document. Graphs are taken from the `graph` folder

2 Relevant parameters

In this section, the relevant parameters used in your experiment are shown. Full information about the parameters can be obtained from file `used_params.txt`

2.1 General

Plugins applied to every sequence, separated by commas. Order is important

1. PluginLowHighSize
2. PluginMids
3. PluginIndeterminations
4. PluginAbAdapters
5. PluginFindPolyAt
6. PluginContaminants
7. PluginVectors
8. PluginLowQuality
9. PluginLowComplexity
10. PluginExtractInserts

Remove duplicated (clonal) sequences (using CD-HIT 454)
`remove_clonality: true`
Minimum insert size for every trimmed sequence
`min_insert_size_trimmed: 40`
Minimum insert size for each end of paired-end reads; true paired-ends have both single-ends longer than this value
`min_insert_size_paired: 40`
Seqtrim version
`seqtrim_version: 2.0.35`
Minimum size for a raw input sequence to be analysed (shorter reads are directly rejected without further analysis)
`min_sequence_size_raw: 40`

2.2 Quality

Minimum quality value for every nucleotide
`min_quality: 20`
Quality window for scanning low quality segments
`window_width: 15`

2.3 Contaminants

Blast E-value used as cut-off when searching for contaminations
`blast_evalue_contaminants: 1.0e-10`
Minimum required identity (%) for a reliable contamination
`blast_percent_contaminants: 85`
Minimum hit size (nt) for considering a true contamination
`min_contam_seq_presence: 40`

`genus:`

Is a contamination considered a source of sequence rejection? (setting to false will only trim contaminated sequences instead of rejecting the complete read)

`contaminants_reject: true`

Path for contaminants database

`contaminants.fasta`
`cont_ribosome.fasta`
`cont_mitochondrias.fasta`
`cont_plastids.fasta`

3 Pre-processing statistics

The following figure shows the size distribution of reads of input data

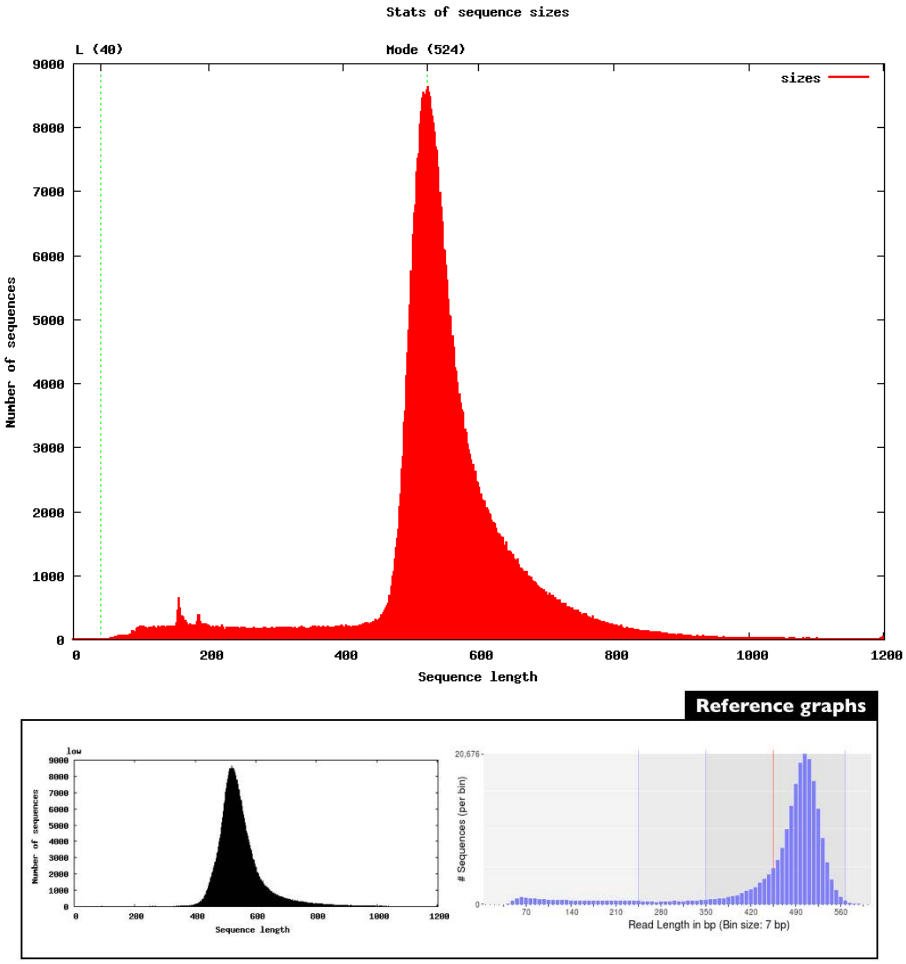


Figure 1: Upper plot: Size distribution of the reads analysed by SeqTrim-Next. If they come from GS-FLX (454 technology), the profile should be close to the one shown in the bottom image, where an example of an appropriate read length distribution is shown. Peak position (modal read length) will depend on the pyrosequencing technology used. [size_stats.png]

Next figure is illustrating the distribution of quality values (QV) for each position on the reads from the input data.

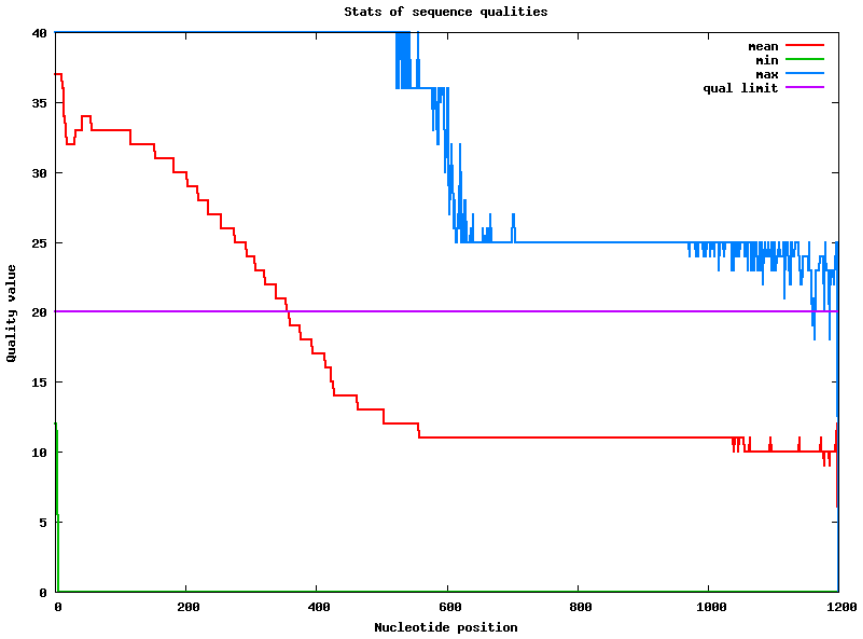


Figure 2: Distribution of the QV by position in the read. The useful part of sequences correspond to a mean QV$>$ 20. You can also see the maximum QV(that should be $\sim$ 40) and the minimum QV. When sequences are becoming very bad, minimum QV$>$ 0 [qualities.png]

Next figure is equivalent to Figure 1 but using output reads (useful sequences). The mode is expected to decrease but the shape of the plot should be similar.

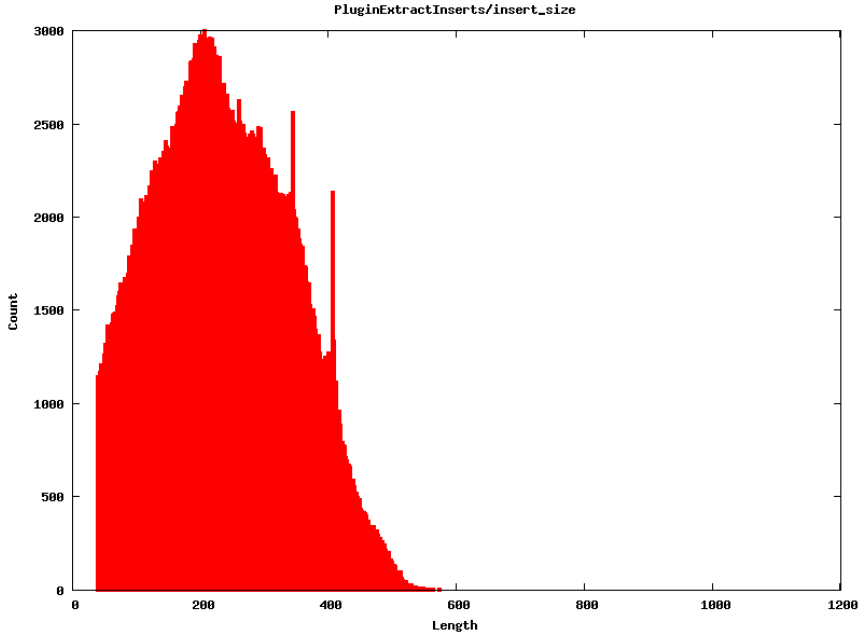


Figure 3: Size distribution of the output sequences. Short sequences (< `min_insert_size_trimmed`) were removed. [PluginExtractInserts_insert_size.png]

Summary statistics of the SeqTrimNext analysis. Be careful and read all warnings that are indicating concerns about your data. In the files `initial_stats.json` and `stats.json` can be found a full statistics of your data and SeqTrimNext pre-processing

Input reads:	total	913786
	Smallest read (bp)	49
	Largest read (bp)	1201
	Mode (bp)	524
	Mean (bp)	544.6
Output results:	total	717128
	Rejected	87281
	Low complexity reads	109377
	Mode (bp)	209
	Mean (bp)	236.0

number of reads with MID: 845390 (92.515%)

Table 1: List of the most frequent Vectors found among your reads

Vectors	sequences
Cloning vector pAS2-1	480
Enterobacteria phage lambda	153
Cloning vector pWormgate2, complete sequence.	149
Cloning vector pVLH/hsp	141
Cloning vector pKOHPRT complete sequence.	119

Table 2: List of the most frequent Adapters found among your reads

Adapters	sequences
rare_adapter_right	50315
rare_adapter_left	24543
rare_adapter_right adap_b_upper new_ab_adapter adap_b_roche	12159
rare_adapter_right adap_b_upper new_ab_adapter	11498
adap_b_upper new_ab_adapter adap_b_roche	8203

Table 3: List of the most frequent Contaminants found among your reads

Contaminants	sequences
Podospora anserina S mat+ unordered scaffolds, whole genome shotgun sequence	456
Penicillium chrysogenum Wisconsin 54-1255 unordered unplaced scaffolds, whole genome shotgun sequence	358
rRNA_long_subunit_Viridiplantae_Larix	278
Aspergillus niger CBS 513.88 clone An02	234
Schizosaccharomyces pombe 972h- chromosome III, complete sequence	220

Table 4: Summary of nucleotides removed in every plugin.

Plugin	Nucleotides	Percent	Warnings
Low Quality	207446036	41.683 %	ntW1
Low Complexity	13833249	2.780 %	ntW4
Adapters	4067903	0.817 %	OK
Poly T	1954436	0.393 %	OK
Poly A	962518	0.193 %	OK
Vectors	60727	0.012 %	OK
Indeterminations	1579384	0.317 %	ntW8
Contaminants	6859	0.001 %	OK
Inserts	195089434	39.200 %	iW1

iW1 Warning!, only 39.200 % of nucleotides are useful

ntW1 Warning!, there are too many (41.683 %) low quality nucleotides

ntW4 Warning!, there are too many (2.780 %) low complexity nucleotides

ntW8 Warning!, too many nucleotides (0.317 %) are indeterminations (Ns)

4 Rejected reads

Input sequences	913786
Output sequences	717128
Rejected sequences	87281
Low complexity sequences	109377

Table 5: Summary of reads removed in every plugin.

Case	Number of sequences	Percent	Warnings
Repeated Sequences	55071	6.027 %	OK
Short inserts	20578	2.252 %	OK
Low Complexity	7872	0.861 %	OK
Contaminants	3087	0.338 %	OK
Empty Inserts	528	0.058 %	OK
No Valid Inserts	113	0.012 %	OK
Indeterminations	31	0.003 %	OK
Unexpected Vector	1	0.000 %	OK
Total rejected	87281	9.552 %	OK

References

[1] Falgueras et al. SeqTrim: a high-throughput pipeline for preprocessing any type of sequence reads. *BMC Bioinformatics* 11:38 (2010)

[2] Weizhong Li & Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics* (2006) 22:1658-9

Thanks you for use SeqTrimNext! Send us any comment to scbi support

Apéndice G

Script para generar el informe de SeqTrimNext

Este *script* en Ruby se utiliza para crear el informe automático de SeqTrimNext en formato PDF, mencionado en el apartado 10.1.2. Se compone de varios ficheros: en primer lugar `generate_report.rb`, que es el fichero principal, que se encarga de analizar el fichero en formato json (no mostrado) para cargar los datos, llamar a las clases (otros *scripts* para dividir y organizar los análisis de datos) `params_report.rb`, `rejected_report.rb` y `stats_report.rb`, y generar el informe PDF. A continuación se presenta el código del *script* `generate_report.rb` y en las siguientes páginas puede encontrarse el código de las clases:

```
#!/usr/bin/env ruby

# Noe Fdez Pozo 2011-05-11. To build a PDF with latex parsing SeqTrimNext output files

#------------------------------------------------------ para indicar donde estan las clases
ROOT_PATH=File.dirname(File.dirname(__FILE__))
# $: << File.expand_path(File.join(ROOT_PATH, 'lib','seqtrimnext_report','classes'))
# $: << File.expand_path(File.join(ROOT_PATH, 'lib'))

# $: << '/Users/dariogf/progs/ruby/gems/seqtrimnext/lib'

#------------------------------------------------------ gems
require 'json'
#------------------------------------------------------ classes

require 'seqtrimnext'
require 'seqtrimnext_report'
require 'params'
require 'scbi_stats'
require 'params_report'
require 'stats_report'
require 'rejected_report'

#------------------------------------------------------ method to parse a json
def get_json_data(file)

  data=nil

  if File.exists?(file)
    file1 = File.open(file)
    text = file1.read
    file1.close

    text=text.gsub(/^\s*#.*$/,'').gsub(/^\n$/,'')
    if !text.nil? && !text.empty?
      data = JSON.parse(text)
    end
  end

  return data
end

if ARGV.count!=1
  puts "Usage: #{File.basename($0)} output_files_folder"
  exit(-1)
```

```
46  end
47
48  #------------------------------------------------- check if files exists
49  output_files=ARGV.shift
50  if !Dir.exists?(output_files)
51    puts "Directory #{output_files} doesn't exists"
52    exit(-1)
53  end
54
55  if !File.exist?(File.join(output_files,'used_params.txt'))
56    puts "used_params.txt file not found.\n"
57    exit(-1)
58  elsif !File.exist?(File.join(output_files,'initial_stats.json'))
59    puts "initial_stats.json file not found.\n"
60    exit(-1)
61  elsif !File.exist?(File.join(output_files,'stats.json'))
62    puts "stats.json file not found.\n"
63    exit(-1)
64  end
65
66  puts "Generating report"
67  puts "="*50
68  puts "used_params.txt, initial_stats.json and stats.json files were found"
69
70
71  #-------------------------------------------------- MAIN ----------------------------------
72  begin
73
74    # load used params
75    all_params=Params.new(File.join(output_files,'used_params.txt'))
76
77    # load initial and final stats
78    initial_stats = get_json_data(File.join(output_files,'initial_stats.json'))
79    stats = get_json_data(File.join(output_files,'stats.json'))
80
81          if (initial_stats.nil?)
82                puts "initial_stats.json info does not exist\n"
83            exit(-1)
84          end
85
86          if (stats.nil?)
87                puts "stats.json info does not exist\n"
88            exit(-1)
89          end
90
91    # load json configuration
92    plugin_fix_hash = get_json_data(File.join(File.dirname(__FILE__),'..','lib','
          seqtrimnext_report','config','plugin_seqs.json'))
93    plugin_nts_hash = get_json_data(File.join(File.dirname(__FILE__),'..','lib','
          seqtrimnext_report','config','plugin_nts.json'))
94
95    output_latex=File.join(output_files,'latex')
96
97    # copy latex required files to output_folder
98    `cp -r #{File.join(File.dirname(__FILE__),'..','lib','seqtrimnext_report','latex_src')}
          #{output_latex}`
99    `cp -r #{File.join(output_files,'..','graphs')} #{output_latex}`
100
101   #------------------------------------------------- Parameters
102   ParamsReport.new(all_params,output_files, output_latex)
103
104   #------------------------------------------------- Statistics
105   StatsReport.new(all_params,initial_stats,stats,plugin_nts_hash, output_files,
          output_latex)
106
107   #------------------------------------------------- Rejected
108   RejectedReport.new(stats,plugin_fix_hash,output_files, output_latex)
109
110   #------------------------------------------------- Build pdf
111   # system('pdflatex main.tex')
112   cmd="pushd .; cd #{output_latex}; pdflatex -halt-on-error main.tex; popd"
113   puts "Running pdflatex: #{cmd}"
114
115   # it must be repeated to solve links in latex
116   `#{cmd}`
117   `#{cmd}`
118
119   if File.exists?(File.join(output_latex,'main.pdf'))
120     `cp #{File.join(output_latex,'main.pdf')} #{File.join(output_files,'statistics_report.
          pdf')}`
121     `zip -r #{File.join(output_files,'latex.zip')} #{output_latex}`
122     `rm -r #{output_latex}`
```

```
123     end
124
125   # rescue
126
127     # puts "Output PDF couldn't be created for this dataset"
128
129   end
130   # system(cmd)
```

params_report.rb es una clase del *script* `generate_report.rb`, utilizada para imprimir en el informe los parámetros utilizados en la limpieza.

```
class ParamsReport

  def initialize(all_params,output_files, output_latex)

    # all_params.print_parameters

    params_in_pdf = {}

    params_in_pdf['general']=['plugin_list','remove_clonality','min_insert_size_trimmed','
        min_insert_size_paired','seqtrim_version','min_sequence_size_raw']
    params_in_pdf['quality']=['min_quality','window_width']
    params_in_pdf['contaminants']=['blast_evalue_contaminants','blast_percent_contaminants'
        ,'min_contam_seq_presence','genus','contaminants_reject','contaminants_db']

    output=File.open(File.join(output_latex,'UsedParams.tex'), 'w')
    output.puts "%!TEX root = FinalReport.tex"

    output.puts '\subsection{General}'
    print_each_group_of_params(output,all_params,params_in_pdf,'general')

    if (all_params.get_param('min_quality'))
      output.puts '\subsection{Quality}'
      print_each_group_of_params(output,all_params,params_in_pdf,'quality')
    end

    if (all_params.get_param('blast_evalue_contaminants'))
      output.puts '\subsection{Contaminants}'
      print_each_group_of_params(output,all_params,params_in_pdf,'contaminants')
    end

    output.close

    puts "used parameters information was added to the report"

  end

  def print_each_group_of_params(output,all_params,params_in_pdf,my_group)

    params_in_pdf[my_group].each do |param_name|

      if !all_params.get_comment('params',param_name).nil?
        description = all_params.get_comment('params',param_name).last

        description.gsub!('%','\%')
        description.gsub!('_','\_')

        if description =~/^\#\#+/
          description = ''
        end
      else
        description = ''
      end

      value = all_params.get_param(param_name)

      if (param_name == 'plugin_list')
        puts value
        values = value.split(",")
        output.puts "#{description}"+'\\\\'

        count = 0
        values.each do |plugin|
          count += 1
          output.puts '\indent'+"#{count}\. #{plugin}"+'\\\\'
        end

        output.puts '\\\\'
      elsif (param_name == 'contaminants_db')
        value.gsub!('"','')
        values = value.split(" ")
        output.puts "#{description}"+'\\\\'
        values.each do |plugin|
          plugin = File.basename(plugin)
          plugin.gsub!('_','\_')
          output.puts '\indent \texttt{'+" #{plugin}"+'}\\\\'
        end
      else
        param_name.gsub!('_','\_')
```

```
77            output.puts "#{description}"+'\\\\'
78            output.puts '\indent \texttt{'+"#{param_name}: #{value}"+'}\\\\'
79          end
80        end
81      end
82
83    end
```

stats_report.rb es otra clase del *script generate_report.rb*. Se utiliza para obtener las estadísticas de la limpieza, secuencias de entrada y salida, pareadas, baja complejidad, gráficas de entrada y salida y de calidad, la moda y la media, LOS MID encontrados, vectores, adaptadores y contaminantes.

```
1  class StatsReport
2
3    def initialize(all_params,initial_stats,stats,plugin_nts_hash,output_folder,output_latex)
4
5      output2=File.open(File.join(output_latex,'stats.tex'), 'w')
6      output2.puts "%!TEX root = FinalReport.tex"
7
8      if (stats['sequences'].nil?) || (stats['sequences']['count'].nil?)
9        puts "sequences info does not exist in stats.json\n"
10       exit(-1)
11     end
12
13     input_seqs = stats['sequences']['count']['input_count'].to_i
14     rejected_seqs = stats['sequences']['count']['rejected'].to_i
15     output_seqs = stats['sequences']['count']['output_seqs'].to_i
16
17     #-------------------------------------------------------------------------------
18     # solo cuando hay pareadas
19     output_seqs_paired = 0
20     total_output_seqs = 0
21
22     if (!stats['sequences']['count']['output_seqs_paired'].nil?)
23       output_seqs_paired = stats['sequences']['count']['output_seqs_paired'].to_i
24       total_output_seqs = output_seqs_paired+output_seqs
25     end
26     #-------------------------------------------------------------------------------
27     # solo cuando hay baja complejidad
28     low_complex = 0
29     if (!stats['sequences']['count']['output_seqs_low_complexity'].nil?)
30       low_complex = stats['sequences']['count']['output_seqs_low_complexity'].to_i
31     end
32     # graph files ------------------------------------------------------------------
33
34     if File.exist?(File.join(output_latex,'graphs','size_stats.png'))
35     output2.puts '\input{input_graph}'
36     end
37
38     if File.exist?(File.join(output_latex,'graphs','qualities.png'))
39     output2.puts '\input{qv_graph}'
40     end
41
42     if File.exist?(File.join(output_latex,'graphs','PluginExtractInserts_insert_size.png'))
43     output2.puts '\input{output_graph}'+"\n\n"
44     end
45     #-------------------------------------------------------------------------------
46     # obtenemos la moda y la media para los datos,
47     # antes y despues de ser preprocesados por SeqTrimNext
48     (input_mode, output_mode) = get_mode(initial_stats,stats)
49     (input_mean, output_mean) = get_mean(initial_stats,stats)
50
51     #------------------------------------------------------------------- build table
52     output2.puts '\begin{table}[H]'
53     output2.puts '\begin{center}'
54     output2.puts '\begin{tabular}{l r r}'
55     output2.puts " \\hline"
56     if (!input_seqs.nil?)
57       output2.puts "Input reads: & total & #{input_seqs} \\\\"
58     end
59     if (!initial_stats['smallest_sequence_size'].nil?)
60       output2.puts " & Smallest read (bp) & #{initial_stats['smallest_sequence_size'].to_i}
           \\\\"
61     end
62     if (!initial_stats['biggest_sequence_size'].nil?)
63       output2.puts " & Largest read (bp)& #{initial_stats['biggest_sequence_size'].to_i}
           \\\\"
64     end
65     output2.puts " & Mode (bp) & #{input_mode} \\\\"
66     output2.puts " & Mean (bp)& #{input_mean} \\\\"
67
68     output2.puts " \\\\ \\hline"
69     output2.puts "Output results: & total & #{output_seqs} \\\\"
70     output2.puts " & Rejected & #{rejected_seqs} \\\\"
71     if (low_complex != 0)
72       output2.puts " & Low complexity reads & #{low_complex} \\\\"
73     end
74     output2.puts " & Mode (bp)& #{output_mode} \\\\"
75     output2.puts " & Mean (bp)& #{output_mean} \\\\"
```

```
76
77      #------------------------------------------------------------ solo cuando hay pareadas
78      output2.puts "\\\\"
79      if (output_seqs_paired != 0)
80        output2.puts " & Output paired reads & #{output_seqs_paired} \\\\"
81        output2.puts " & Total output reads & #{total_output_seqs} \\\\"
82        output2.puts "\\\\ \\hline"
83        output2.puts "Linkers: & & \\\\"
84
85        if (!stats['PluginLinker'].nil?)
86          if (!stats['PluginLinker']['linker_id'].nil?)
87            stats['PluginLinker']['linker_id'].each do |linker|
88              output2.puts " & #{linker[0]} & #{linker[1]} \\\\"
89            end
90          end
91          output2.puts "\\\\ \\hline"
92          if (!stats['PluginLinker']['without_linker'].nil?)
93            output2.puts "Without linkers: & total & #{stats['PluginLinker']['without_linker
                ']['0']} \\\\"
94          end
95
96          output2.puts "\\\\ \\hline"
97          output2.puts "Multiple linkers: & & \\\\"
98
99          if (!stats['PluginLinker']['multiple_linker_id'].nil?)
100           stats['PluginLinker']['multiple_linker_id'].each do |linker|
101             output2.puts " & #{linker[0]} & #{linker[1]} \\\\"
102           end
103         end
104         if (!stats['PluginLinker']['multiple_linker_count'].nil?)
105           stats['PluginLinker']['multiple_linker_count'].each do |linker|
106             output2.puts " & With #{linker[0]} linkers & #{linker[1]} \\\\"
107           end
108         end
109       end
110     end
111
112     #------------------------------------------------------------------------ end pareadas
113
114     output2.puts "\\hline"
115
116     output2.puts '\end{tabular}'
117     output2.puts '\label{table:nonlin}'
118     output2.puts '\end{center}'
119     output2.puts '\end{table}'+"\n\n"
120     #------------------------------------------------------------------------------ end table
121
122     #------------------------------------------------------------------------------------ MIDs
123     if (!stats['PluginMids'].nil?) && (!stats['PluginMids']['mid_id'].nil?)
124       mid_seqs = stats['PluginMids']['mid_id']['total']
125       mid_seqs_percent = sprintf("%0.3f", (mid_seqs.to_f*100/input_seqs.to_f))
126       output2.puts '\noindent \begin{minipage}{\linewidth}'
127       output2.puts "number of reads with MID: #{mid_seqs} \(#{mid_seqs_percent}\\%\)"+'\\\\
            '+'\\\\'
128
129       if (mid_seqs_percent.to_f <= 1)
130         output2.puts '\fcolorbox{black}{yellow}{'+"\n"+'\begin{minipage}{\linewidth}{'+"\n"
              +'\textbf{WARNING: The number of reads with MID is so low that can be
              interpreted as a random finding. Your useful sequences are in the no\_MID
              folder, but you can also add any read classified as having a MID}'+"\n"+'}'+"\n
              "+'\end{minipage}'+"\n"+'}\\\\\\\\'
131       end
132       output2.puts '\end{minipage}'+"\n\n"
133     end
134     #------------------------------------------------------------------------------------
135
136     #------------------------------------------------------------ make top five tables
137     if !(stats['PluginVectors']).nil?
138       if !(top_hash = stats['PluginVectors']['vectors_ids']).nil?
139         make_a_top_five(output2, top_hash, 'Vectors')
140       end
141     end
142
143     if !(stats['PluginAbAdapters']).nil?
144       if !(top_hash = stats['PluginAbAdapters']['adapter_id']).nil?
145         make_a_top_five(output2, top_hash, 'Adapters')
146       end
147     end
148
149     if !(stats['PluginContaminants']).nil?
150       if !(top_hash = stats['PluginContaminants']['contaminants_ids']).nil?
151         make_a_top_five(output2, top_hash, 'Contaminants')
```

```
152         end
153       end
154       #-------------------------------------------------------------------------------
155
156       # en las pareadas utilizamos el inserto de izq y derecha ---------------------
157       # solo cuando hay pareadas
158       paired_nts=0
159       if (stats['PluginExtractInserts']['left_insert_size'])
160         stats['PluginExtractInserts']['left_insert_size'].each do |element|
161           paired_nts += element[0].to_i*element[1].to_i
162         end
163       end
164       if (stats['PluginExtractInserts']['right_insert_size'])
165         stats['PluginExtractInserts']['right_insert_size'].each do |element|
166           paired_nts += element[0].to_i*element[1].to_i
167         end
168       end
169       #-------------------------------------------------------------------------------
170
171       nts_total = initial_stats['nucleotide_count']
172       print_trimmed_nts_stats_table(stats, output2, plugin_nts_hash,nts_total,paired_nts)
173
174       output2.close
175
176       puts "Statistic information was added to the report"
177
178     end
179
180     def get_mode(initial_stats,stats)
181       output_mode = 0
182       mode_array = []
183
184       # take the mode from initial_stats.json
185       if (!initial_stats.nil? and !initial_stats.empty?)
186         input_mode = initial_stats['mode_of_sizes']
187       else
188         input_mode = 0
189       end
190
191       # calculate the mode using data from stats.json
192       if (!stats['PluginExtractInserts']['insert_size'].nil?)
193         stats['PluginExtractInserts']['insert_size'].each do |key,value|
194           mode_array[key.to_i]=value
195         end
196
197         mode_array.map!{|e| e || 0}
198         s=ScbiStats.new(mode_array)
199
200         output_mode = s.fat_mode
201       else
202         output_mode = 0
203       end
204
205       return [input_mode, output_mode]
206     end
207
208     def get_mean(initial_stats,stats)
209       output_mean = 0
210
211       # take the mean from initial_stats.json
212       if (!initial_stats.nil? and !initial_stats.empty?)
213         input_mean = sprintf("%0.1f", (initial_stats['mean_of_sequence_sizes']))
214       else
215         input_mean = 0
216       end
217
218       # calculate the mean using data from stats.json
219       nts_count = 0
220       seqs_count = 0
221       if (!stats['PluginExtractInserts']['insert_size'].nil?)
222         stats['PluginExtractInserts']['insert_size'].each do |key,value|
223           seqs_count += value.to_i
224           nts_count += (key.to_f*value)
225         end
226
227         if (nts_count == 0 || seqs_count == 0)
228           output_mean = 0
229         else
230           output_mean = sprintf("%0.1f", (nts_count/seqs_count))
231         end
232       else
233         output_mean = 0
```

```
234       end
235
236       return [input_mean, output_mean]
237     end
238
239     def make_a_top_five(output2,top_hash,name)
240       #------------------------------------------------- build table
241       output2.puts '\begin{table}[H]'
242       output2.puts '\caption{'+"List of the most frequent~#{name}~found among your reads"+'}'
243       output2.puts '\vspace{-0.5cm}'
244       output2.puts '\begin{center}'
245       output2.puts '\begin{tabular}{|p{11cm}|r|}'
246       output2.puts '\hline'
247       output2.puts "#{name} " +'& sequences \\\\ [0.5ex]'
248       output2.puts '\hline'
249
250       cont = 0
251       top_hash.sort{|a,b| b[1]<=>a[1]}.each do |elem|
252         tmp_name = elem[0].gsub('_','\_')
253         output2.puts "#{tmp_name} \& #{elem[1]}"+'\\\\'
254         cont+=1
255         if (cont == 5)
256           break
257         end
258       end
259
260       output2.puts '\hline'
261       output2.puts '\end{tabular}'
262       output2.puts '\end{center}'
263       output2.puts '\end{table}'+"\n\n"
264       #------------------------------------------------- end table
265     end
266
267     def print_trimmed_nts_stats_table(stats, output2, plugin_nts_hash, nts_total,paired_nts)
268
269       nts_table_hash = {}
270       insert_array = []
271       warning_array = []
272
273       plugin_nts_hash.each do |plugin|
274
275         my_name = plugin[0]
276         plugin_name = plugin[1]['plugin']
277         plugin_field = plugin[1]['field']
278         plugin_msg = plugin[1]['msg']
279         plugin_threshold = plugin[1]['threshold']
280         plugin_warning = plugin[1]['warning']
281
282         if (!stats[plugin_name].nil?)
283           if (!stats[plugin_name][plugin_field].nil?)
284
285             count = 0
286             stats[plugin_name][plugin_field].each do |element|
287               count += element[0].to_i*element[1].to_i
288             end
289
290             if (plugin_name == 'PluginExtractInserts') && (plugin_field == 'insert_size') &&
                    (paired_nts > 0)
291               count += paired_nts
292             end
293
294             my_percent = sprintf("%0.3f", (count.to_f*100/nts_total.to_f))
295
296             if (plugin_name == 'PluginExtractInserts')
297               if (my_percent.to_f <= plugin_threshold)
298                 plugin_msg.gsub!('my_percent',"#{my_percent}")
299                 insert_array.push '\noindent \fcolorbox{black}{pink}{'+"\n"+'\begin{minipage
                        }{\linewidth}{'+"\n"+'\textbf{'+"#{plugin_warning}  #{plugin_msg}"+'}'+"\
                        n"+'}'+"\n"+'\end{minipage}'+"\n"+'}\\\\\\\\'
300               else
301                 plugin_warning = 'OK'
302               end
303             else
304               if (my_percent.to_f >= plugin_threshold)
305                 plugin_msg.gsub!('my_percent',"#{my_percent}")
306                 warning_array.push '\noindent \fcolorbox{black}{yellow}{'+"\n"+'\begin{
                        minipage}{\linewidth}{'+"\n"+'\textbf{'+"#{plugin_warning}  #{plugin_msg}
                        "+'}'+"\n"+'}'+"\n"+'\end{minipage}'+"\n"+'}\\\\\\\\'
307               else
308                 plugin_warning = 'OK'
309               end
310             end
```

```
311
312             nts_table_hash[plugin_field] = ["#{my_name}&#{count}&#{my_percent} \\%&#{
                    plugin_warning}\\\\",my_percent]
313
314           end
315         end
316       end
317
318       #------------------------------------------------------ build table
319       output2.puts '\begin{table}[H]'
320       output2.puts '\caption{Summary of nucleotides removed in every plugin.}'
321       output2.puts '\begin{center}'
322       output2.puts '\begin{tabular}{l r r c}'
323       output2.puts '\hline'
324       output2.puts 'Plugin & Nucleotides & Percent & Warnings \\\\ [0.5ex]'
325       output2.puts '\hline'
326
327       #the hash of hashes is ordered by value (number of sequences rejected)
328       nts_table_ordered = nts_table_hash.sort {|a,b| b[1][1].to_i<=>a[1][1].to_i}
329
330       nts_table_ordered.each do |element|
331         if (element[0] != 'insert_size')
332           output2.puts element[1][0]
333         end
334       end
335
336       output2.puts '\hline'
337       if (!nts_table_hash['insert_size'].nil?)
338         output2.puts nts_table_hash['insert_size'][0]
339       end
340       output2.puts '\hline'
341       output2.puts '\end{tabular}'
342       output2.puts '\label{table:nonlin}'
343       output2.puts '\end{center}'
344       output2.puts '\end{table}'+"\n\n"
345       #-------------------------------------------------------- end table
346
347       output2.puts '\noindent \begin{minipage}{\textwidth}'
348
349       output2.puts insert_array.join("\n")
350       output2.puts warning_array.join("\n")
351
352       output2.puts '\end{minipage}'+"\n\n"
353
354     end
355
356   end
```

rejected_report.rb es otra clase del *script generate_report.rb*. Se utiliza para obtener las estadísticas de las secuencias rechazadas y para crear un fichero json con la estructura necesaria para guardar los valores de corte y los mensajes de advertencia de cada una de las estadísticas que se realizan en este informe.

```
1  class RejectedReport
2
3    def initialize(stats,plugin_fix_hash,output_files, output_latex)
4
5      # write_plugin_json
6
7      output3=File.open(File.join(output_latex,'rejected.tex'), 'w')
8      output3.puts "%!TEX root = FinalReport.tex\n\n"
9
10     input_seqs = stats['sequences']['count']['input_count'].to_i
11     rejected_seqs = stats['sequences']['count']['rejected'].to_i
12     output_seqs = stats['sequences']['count']['output_seqs'].to_i
13
14     # secuencias pareadas
15     output_seqs_paired = 0
16     total_output_seqs = 0
17     if (!stats['sequences']['count']['output_seqs_paired'].nil?)
18       output_seqs_paired = stats['sequences']['count']['output_seqs_paired'].to_i
19       total_output_seqs = output_seqs_paired+output_seqs
20     end
21     # secuencias de baja complejidad
22     low_complex = 0
23     if (!stats['sequences']['count']['output_seqs_low_complexity'].nil?)
24       low_complex = stats['sequences']['count']['output_seqs_low_complexity'].to_i
25     end
26
27     rejected_hash = {}
28     data_hash = {}
29
30     data_hash['value'] = rejected_seqs
31     data_hash['warning'] = 'OK'
32     data_hash['warning_msg'] = ''
33     data_hash['percent'] = sprintf("%0.3f", (rejected_seqs.to_f*100/input_seqs.to_f))
34     rejected_hash['rejected']=data_hash
35
36     if (!stats['sequences']['rejected'].nil?)
37       rejected_hash = load_plugins_info(stats, rejected_hash, input_seqs, plugin_fix_hash)
38
39       #------------------------------------------------- build table
40       output3.puts '\begin{table}[H]'
41       output3.puts '\begin{center}'
42       output3.puts '\begin{tabular}{r r}'
43
44       output3.puts "Input sequences & #{input_seqs}\\\\"
45       output3.puts "Output sequences & #{output_seqs}\\\\"
46       output3.puts "Rejected sequences & #{rejected_seqs}\\\\"
47       if (output_seqs_paired != 0)
48         output3.puts "Output paired sequences & #{output_seqs_paired} \\\\"
49         output3.puts "Total output sequences & #{total_output_seqs} \\\\"
50       end
51       if (low_complex != 0)
52         output3.puts "Low complexity sequences & #{low_complex} \\\\"
53       end
54
55       output3.puts '\end{tabular}'
56       output3.puts '\label{table:input_seqs}'
57       output3.puts '\end{center}'
58       output3.puts '\end{table}'+"\n\n"
59       #------------------------------------------------- end table
60
61       #------------------------------------------------- build table
62       output3.puts '\begin{table}[H]'
63       output3.puts '\caption{Summary of reads removed in every plugin.}'
64       output3.puts '\begin{center}'
65       output3.puts '\begin{tabular}{l r r c}'
66       output3.puts '\hline'
67       output3.puts 'Case & Number of sequences & Percent & Warnings \\\\ [0.5ex]'
68       output3.puts '\hline'
69
70       #the hash of hashes is ordered by value (number of sequences rejected)
71       rejected_ordered = rejected_hash.sort {|a,b| b[1]['value']<=>a[1]['value']}
72
73       rejected_ordered.each do |plugin|
74
75         my_name = plugin[1]['name']
```

```
76          my_value = plugin[1]['value']
77          my_percent = plugin[1]['percent']
78          my_warning = plugin[1]['warning']
79
80          if (plugin[0] != 'rejected')
81            if (plugin[0] == 'Indeterminations in middle of sequence')
82              plugin[0] = plugin[0].sub(' in middle of sequence','')
83            end
84            output3.puts "#{my_name}&#{my_value}&#{my_percent} \\%&#{my_warning}\\\\"
85          end
86        end
87
88        output3.puts '\hline'
89        output3.puts "Total rejected&#{rejected_hash['rejected']['value']}&#{rejected_hash['
              rejected']['percent']} \\%&#{rejected_hash['rejected']['warning']}\\\\ [1ex]"
90        output3.puts '\hline'
91        output3.puts '\end{tabular}'
92        output3.puts '\end{center}'
93        output3.puts '\label{table:reads_removed}'
94        output3.puts '\end{table}'+"\n\n"
95        #------------------------------------------------------ end table
96
97        rejected_ordered.each do |plugin|
98          if (plugin[1]['warning'] != 'OK')
99            plugin[1]['warning_msg'].gsub!('my_percent',"#{rejected_hash["#{plugin[0]}"]['
                  percent']}")
100           output3.puts '\noindent\fcolorbox{black}{yellow}{'+"\n"+'\begin{minipage}{\
                  linewidth}{'+"\n"+'\textbf{'+"#{plugin[1]['warning']}    #{plugin[1]['
                  warning_msg']}"+'}'+"\n"+'}'+"\n"+'\end{minipage}'+"\n"+'}\\\\\\\\'
101         end
102       end
103     else
104       output3.puts 'There are not rejected sequences\\\\'
105     end
106
107     output3.close
108
109     puts "Information about rejected sequences was added to the report"
110   end
111
112   def load_plugins_info(stats, rejected_hash, input_seqs, plugin_fix_hash)
113     data_hash = {}
114
115     stats['sequences']['rejected'].each do |rejected|
116       data_hash = {}
117       if plugin_fix_hash[rejected[0]]
118         data_hash['name'] = plugin_fix_hash[rejected[0]]['name']
119         data_hash['value'] = rejected[1]
120         data_hash['warning'] = 'OK'
121         data_hash['warning_msg'] = ''
122         data_hash['percent'] = sprintf("%0.3f", (rejected[1].to_f*100/input_seqs.to_f))
123         rejected_hash[rejected[0]]=data_hash
124       end
125     end
126
127     rejected_hash.each_key do |key|
128       if (rejected_hash[key]['percent'].to_f >= plugin_fix_hash[key]['threshold'] )
129         rejected_hash[key]['warning'] = plugin_fix_hash[key]['warning']
130         rejected_hash[key]['warning_msg'] = plugin_fix_hash[key]['msg']
131       end
132     end
133
134     return rejected_hash
135   end
136
137   def write_plugin_json
138
139     plugin_fix_hash = {}
140     msgs_hash = {}
141
142     msgs_hash['msg'] = "Warning!, there are a my_percent \\% of repeated sequences"
143     msgs_hash['threshold'] = 9
144     msgs_hash['warning'] = 'W1'
145
146     plugin_fix_hash['repeated'] = msgs_hash
147     msgs_hash = {}
148
149     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences are too short"
150     msgs_hash['threshold'] = 10
151     msgs_hash['warning'] = 'W2'
152
153     plugin_fix_hash['short insert'] = msgs_hash
```

```
154     msgs_hash = {}
155
156     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences are empty (without an
            insert)"
157     msgs_hash['warning'] = 'W3'
158     msgs_hash['threshold'] = 1
159
160     plugin_fix_hash['empty insert'] = msgs_hash
161     msgs_hash = {}
162
163     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences are from a contaminant
            organism or from organelles"
164     msgs_hash['warning'] = 'W4'
165     msgs_hash['threshold'] = 1
166
167     plugin_fix_hash['contaminated'] = msgs_hash
168     msgs_hash = {}
169
170     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences are no valid sequences"
171     msgs_hash['threshold'] = 0.1
172     msgs_hash['warning'] = 'W5'
173
174     plugin_fix_hash['No valid inserts found'] = msgs_hash
175     msgs_hash = {}
176
177     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences are low complexity
            sequences"
178     msgs_hash['warning'] = 'W6'
179     msgs_hash['threshold'] = 1
180
181     plugin_fix_hash['low complexity by polyt'] = msgs_hash
182     msgs_hash = {}
183
184     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences contain a vector in an
            unexpected position"
185     msgs_hash['warning'] = 'W7'
186     msgs_hash['threshold'] = 1
187
188     plugin_fix_hash['unexpected vector'] = msgs_hash
189     msgs_hash = {}
190
191     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences contain too much
            indeterminations"
192     msgs_hash['threshold'] = 0.1
193     msgs_hash['warning'] = 'W8'
194
195     plugin_fix_hash['Indeterminations in middle of sequence'] = msgs_hash
196     msgs_hash = {}
197
198     msgs_hash['msg'] = "Warning!, a my_percent \\% of your sequences are too big or too
            small"
199     msgs_hash['threshold'] = 1
200     msgs_hash['warning'] = 'W9'
201
202     plugin_fix_hash['size out of limits'] = msgs_hash
203     msgs_hash = {}
204
205     msgs_hash['msg'] = "WT  Warning!, a my_percent \\% of your sequences were rejected!"
206     msgs_hash['threshold'] = 25
207     msgs_hash['warning'] = 'WT'
208
209     plugin_fix_hash['rejected'] = msgs_hash
210     msgs_hash = {}
211
212     puts JSON.pretty_generate(plugin_fix_hash)
213
214   end
215
216 end
```

Apéndice H

Plantilla para la importación de proyectos a SPDB

Ejemplo de la plantilla que contiene los datos de un nuevo proyecto que va a importarse a la base de datos SustainPineDB. Es de uso exclusivo de los administradores de la base de datos y sirve para aportar toda la información necesaria del proyecto, de modo que uno o varios proyectos (cada uno con su plantilla) puedan importarse de modo automático y en un solo paso.

```
1  # Please fill the template as in the examples format
2  # clear the example if you don't have any information in that field
3
4
5  ### start of assembly block ------------------------------------------------------------------------
6  # a name for your project or assembly
7  assembly_name= Project Name
8
9  # a short name to identify the project name in unigenes name.
10 # eg: sp, in case of sp_unigene1
11 assembly_tag= sp
12
13 # version in database.
14 assembly_version= 2
15
16 # a description of your project (in one line)
17 description= Version 2 of the assembly.
18
19 # organim from your sequences were extracted
20 organism= Pinus pinaster
21
22 # tissue used in your experiment
23 tissue= several
24
25 # name of the ace file
26 ace_name= my_project_v2.ace
27
28 # name of the fasta file with the unigenes of your experiment
29 unigenes_fasta_file= my_project_unigenes_v2.fasta
30 ### end of assembly block --------------------------------------------------------------------------
31
32
33 ### start of sff block -----------------------------------------------------------------------------
34 # copy this block for every sff file
35 # in sff_file_name field you can add a file name or a set of files separated by commas
36 sff_file_name=my_project1.sff, my_project2.sff, my_project3.sff, my_project4.sff
37
38 # description of construction used in the sequencing process  (in one line)
39 sff_description= SFF file from 454 Titanium
40
41 # url to find your sff files in an external server
42 sff_url=
43 ### end of sff block -------------------------------------------------------------------------------
44
45 ### start of sff block -----------------------------------------------------------------------------
46 # copy this block for every sff file
47 sff_file_name=OtherSff_file
48
```

```
49 # description of construction used in the sequencing process  (in one line)
50 sff_description=sra files from other library
51
52 # url to find your sff files in an external server
53 sff_url=http://www.ncbi.nlm.nih.gov/sra/SRP004500
54 ### end of sff block ----------------------------------------------------------------------
55
56 ### start of sff block --------------------------------------------------------------------
57 # copy this block for every sff file
58 sff_file_name=OtherSff_file2
59
60 # description of construction used in the sequencing process  (in one line)
61 sff_description=sra files from other library
62
63 # url to find your sff files in an external server
64 sff_url=http://www.ncbi.nlm.nih.gov/sra/SRP004769
65 ### end of sff block ----------------------------------------------------------------------
66
67
68 # if you want to delete a previous version of this project, with the same name,
69 # before import the new version, write yes
70 delete_previous_version= no
```

Apéndice I

Script para obtener las secuencias de una lista

Este *script* resulta de gran utilidad para obtener una selección de secuencias de un fasta de gran tamaño. Indicando el fichero fasta que contiene todas las secuencias y una lista con los nombres de las secuencias que se desean extraer se obtiene un nuevo fichero de secuencias en formato fasta formado por las secuencias incluidas en la lista.

```
1  #!/usr/bin/env ruby
2
3  # 06-04-2010
4  # Noe Fdez Pozo
5  # Script para conseguir las secuencias de una lista en formato fasta
6
7  if ARGV.size != 2
8          puts "incorrect number of arguments, you need a fasta file and a list of sequences
               in a text file"
9    Process.exit(-1);
10 end
11
12 # get file name
13 (fasta_file,seq_list)=ARGV
14
15
16 def cargar_fasta_en_hash(fasta_file)
17         hash={}
18         name = nil
19         fasta = ''
20
21         File.open(fasta_file).each_line do |line|
22                 line.chomp!
23
24                 if line =~ /^>([^\s]+)/
25                         if fasta != ''
26                                 hash[name]=fasta
27                         end
28
29                         name = $1
30                         fasta = ''
31                 else
32                         fasta += line
33                 end
34         end
35         if fasta != ''
36                 hash[name]=fasta
37         end
38         return(hash)
39 end
40
41
42 def create_fasta(fasta_h, list_file)
43         name = nil
44         File.open(list_file).each_line do |line|
45                 line.chomp!
46                 name = line     # la lista y el fasta son iguales
47
```

```
48                      if (!fasta_h[name].nil?)
49                              puts ">#{name}\n#{fasta_h[name]}"
50                      end
51              end
52      end
53
54      fasta_hash=cargar_fasta_en_hash(fasta_file)
55
56      create_fasta(fasta_hash, seq_list)
```

Parte VIII

Comunicaciones a congresos

XXXIII Congreso de la Sociedad Española de Bioquímica y Biología Molecular

Córdoba14-17Septiembre

Organiza

SEBBM Sociedad Española de Bioquímica y Biología Molecular

Para más información: www.sebbm.es/XXXIIICongreso

SEBBM2010

Colabora

UNIVERSIDAD DE CÓRDOBA

XXXIII Congreso de la Sociedad Española de Bioquímica y Biología Molecular

XXXIII Congreso de la Sociedad Española de Bioquímica y Biología Molecular
Edita: SEBBM, Sociedad Española de Bioquímica y Biología Molecular
Diseño y maquetación: Izanet Global Services, SL
Impresión: Diputación de Córdoba. Imprenta Provincial
ISBN: 978-84-693-4612-9
Deposito legal:

PoR81 - EuroPineDB: una base de datos con metadatos de secuencias de tres especies de pino europeas

Noé Fernández Pozo1, Darío Guerrero2, Rocío Bautista2, David P. Villalobos1, Sar Díaz-Moreno1, Arantxa Flores-Monterroso1, Javier Canales1, M. Ángeles Guevara3, Pedro Pedriguero4, Carmen Collada3,4, M. Teresa Cervera3,4, Álvaro Soto3,4, Ricardo Ordás5, Concepción Ávila1, Francisco R. Cantón1, Francisco M. Cánovas1, M. González Claros1,2.
1 Departamento de Biología Molecular y Bioquímica, Universidad de Málaga, 2 Plataforma Andaluza de Bioinformática, Universidad de Málaga, 3 Departamento de Ecología y Genética Forestal, CIFOR-UNIA, 4 UM Genómica y Ecofisiología Forestal INIA-UPM, Universidad Politécnica de Madrid, 5 Instituto Biouniversitario de Biotecnología de Asturias. Universidad de Oviedo.

EuroPineDB es un portal Web que da acceso a una base de datos concebida como recurso para la investigación y apoyo de los proyectos de secuenciación de coníferas. Contiene secuencias transcriptómicas anotadas de Pinus pinaster, Pinus sylvestris (dos especies modelo y de utilidad agroeconómica) y Pinus pinea, procedentes de genotecas de varios grupos de investigación europeos y completadas con secuencias publicas del EMBL. EuroPineDB está organizada por genotecas, especies, anotaciones, unigenes y micromatrices. También incluyen un motor de búsqueda para las anotaciones y la posibilidad de encontrar secuencias parecidas mediante BLAST. Para obtener las anotaciones de las secuencias sólo se utilizaron herramientas de código abierto. Las secuencias se preprocesaron, ensamblaron y anotaron convenientemente (Gene Ontology, Enzime Commission, KEGG, descripciones, InterPro, Full-Length, SSR, ORF, SNP), lo que no resulta trivial porque las coníferas carecen de un genoma secuenciado completamente y su parentesco con las angiospermas, las plantas modelo principalmente, es remoto. Se puede descargar cualquier secuencia de la base de datos, así como, sus ensamblajes y anotaciones. Además, EuropineDB permite visualizar las secuencias, inspeccionar su disposición en las micromatrices de pino y en los ensamblajes que forman los unigenes de cada especie o genoteca. En EuroPineDB, por tanto, se puede hallar una metainformación que no está disponible en otras bases de datos que contengan las mismas secuencias. Gracias a la interfaz Web es fácil de utilizar y ofrece una información que sirve para apoyar directamente el trabajo experimental en las especies de pino y otras coníferas.
EuroPineDB está disponible en http://www.scbi.uma.es/pindb/

MICINN

PoR129 - Integración funcional de perfiles de expresión genómicos y proteómicos en neutrófilos porcinos estimulados con LPS

Gema Sanz, Ángeles Jiménez-Marín, Ángela Moreno, Rocío Bautista, Noé Fernández, Gonzalo Claros, Juan José Garrido
Grupo de Genómica y Mejora Animal, Departamento de Genética, *Facultad de Veterinaria, Universidad de Córdoba, Córdoba*

Hasta la fecha, la integración de datos de transcriptómica y proteómica es una tarea ardua, principalmente debido la discordancia entre los resultados de expresión de genes y proteínas. Esto dificulta la interpretación conjunta de ambos tipos de datos. Esta falta de concordancia podría depender, en parte, del momento en el que se determina el cambio de expresión, por lo que un estudio temporal podría proporcionar una visión unificada de los datos. Por otro lado, centrar la integración en la correlación directa entre la regulación de genes y proteínas también podría ser la causa de dicha falta de concordancia

En este estudio hemos combinado datos de expresión génica y abundancia de proteínas para una mejor comprensión de la base molecular de la respuesta de los neutrófilos porcinos a la estimulación con LPS de Salmonella Typhimurium. Para ello, los resultados de un ensayo temporal se han analizado con la herramienta Ingenuity Pathway Analysis® en el contexto de funciones y rutas biológicas. Aunque en general se observa una falta de concordancia entre los genes y proteínas diferencialmente expresados, existe un alto grado de correlación a nivel funcional. Así, la integración de ambos datos muestra que a las 6 horas de estimulación con LPS se regulan funciones relacionadas con movimiento, estructura celular y respuesta inflamatoria. A las 9 horas también existe una elevada correlación entre funciones obtenidas con datos genómicos y proteómicos, destacando la función de muerte celular y las rutas canónicas de respuesta en fase aguda y endocitosis mediada por caveolas. Finalmente, a las 18 horas destacan las funciones de muerte celular, señalización célula-célula y estructura y movimiento celular. En conclusión, nuestros resultados muestran que en el sistema analizado, la integración de datos procedentes del análisis transcriptómico y proteómico a gran escala sólo es posible mediante la caracterización funcional de los genes y proteínas diferencialmente expresados.

Book of Abstracts

Xth Spanish Symposium on Bioinformatics

Malaga, Spain, October 27-29, 2010

Organized by

National Institute of Bioinformatics (INB-Spain)

Portuguese Bioinformatics Network (PBN-Portugal)

Bioinformatics and Information Technologies Laboratory (Bitlab), Computer Architecture Department, Universidad de Málaga

Editors

Alfonso Valencia Herrera
Victoria Martín Requena
Oswaldo Trelles Salazar

GeNOTE: a web tool for annotation of non-model, eukaryotic, unfinished sequences

Noé Fernández-Pozo[1], Darío Guerrero-Fernández[2], Rocío Bautista[2], J. Gómez-Maldonado[1], C. Avila[1], Francisco M. Cánovas[1], M. Gonzalo Claros[1,2]

[1]Departamento de Biología Molecular y Bioquímica, Universidad de Málaga, Campus de Teatinos, 29071 Málaga, Spain
{noefp, pgomez, cavila, canovas, claros}@uma.es,
WWW home page: http://www.bmbq.uma.es/fmp

[2]Plataforma Andaluza de Bioinformática, Universidad de Málaga, Severo Ochoa 34, 29590 Málaga, Spain
{dariogf, rociobm, claros}@scbi.uma.es,
WWW home page: http://www.scbi.uma.es/pab

Abstract. *De novo* identification of genes in newly-sequenced eukaryotic genomes is based on sensors, which are not available in non-model organisms. Many annotation tools have been developed and most of them require sequence training, computer skills and accessibility to sufficient computational power. The main need of non-model organisms is finding genes, and this can be done with GeNOTE, which can be accessed as a web tool for researchers without bioinformatics skills. It facilitates the annotation of new, unfinished sequences with descriptions, GO terms, EC numbers and KEEG pathways. It is also able to sort contigs or scaffolds from a BAC clone. Results are provided in GFF format and in tab-delimited text readable in viewers like Apollo or Artemis.

Keywords: Annotation, web tool, unfinished sequence, gene finding, non-model species

1 Introduction

In recent years, the biological community has started to see the dramatic impact of new sequencing technologies on the number of sequenced genomes, and it is expected that this influx of data will continue to escalate in the near future. Annotation of newly-sequenced eukaryotic genomes is based on sensors such as promoters, splice sites, start and stop codons or untranslated regions. Sensors can be predicted only for well known species and microorganisms [3], but produce highly incorrect predictions in newly-sequenced organisms [5]. Genome annotation is therefore becoming the bottleneck in genomics today. Eukaryotic genomes are particularly at risk as their large size and intron-containing genes make them difficult targets for annotation.

Many annotation tools have been developed and most of them require a cumbersome installation including installation of external executables, sequence training, programming skills and accessibility to sufficient computational power. In fact, many laboratories where

XI Jornadas de Bioinformática

January 23-35th, 2012 @ PRBB/BARCELONA http://jbi2012.org

BOOK OF ABSTRACTS

OUR THANKS TO:

ORGANIZATION SUPPORT:

Genome Annotation. Talk #3

Highly efficient pre-processing of NGS reads and identification of full-length genes.

Darío Guerrero-Fernández[1], Noé Fernández-Pozo[2], Almudena Bocinos[1], Rocío Bautista[1] and M. Gonzalo Claros[1,2]

1 Plataforma Andaluza de Bioinformática-Centro de Supercomputación y Bioinformática, Universidad de Málaga, Málaga. 2 Departamento de Biología Molecular y Bioquímica, Facultad de Ciencias, Universidad de Málaga, Málaga.

The advent of the technologies so-called next-generation sequencing (NGS) is giving the ability to obtain large amounts of sequences. However, NGS reads are not clean and it is necessary to remove (depending on the experimental approach) polyA/T, adaptors, contaminations, low quality portions, low complexity segments, artefactual duplicates, tags or sample identifiers, etc. Therefore, there is a need for new, fast, efficient, reliable, easy- to-install, user-friendly, pre-processing software for NGS reads for a wide range of computers and experimental conditions (e.g. de novo assembling, mapping, amplicons). SeqTrimNext has been developed to fill these necessities and to cope with all NGS peculiarities, including parallelisation, managing of paired-end reads, managing and grouping sequences by barcodes or tags, and providing output files that can be used as input for downstream analyses. The modular architecture of SeqTrimNext, based on a pipeline of orthogonal plugins, is especially suitable for addition, removal, or reordering of plugins and easy adaptation for future evolution. It also provides a detailed statistics on the input and output datasets and a PDF report where users can find clues for the their sequence qualities. SeqTrimNext recovers more true paired-ends than others (e.g. Newbler), is released with a customised database for contaminants (even if users can use their own contamination database), and provides pre-coded configuration files for the most common NGS analyses. Assembly of SeqTrimNext-treated reads takes less time, provide less but longer contigs (net increase of N50), dramatically diminished numbers of repeated targets and multiply mapped reads, reduce de amount of chimerical contigs and reduces the subsequent time of manual curation of assemblies and mappings because the obtained results lack misconnections due to artefactual sequences. It can be used at http://www.scbi.uma.es/seqtrimnext

When NGS reads were obtained from transcriptomics experiments, the assembly would require a deeper analysis to know its accuracy and reliability, as well as if any full-length gene has been reconstructed and the presence of putative new genes. This can be achieved by means of FullLengtherNext. Using customised DNA and protein databases (FullLengtherNext is provided with a script for constructing the user-adapted database for the taxonomic group of interest), it is able to predict the completeness of sequences (full- length, internal, N-terminal or C-terminal), predict the artefactual contigs (e.g., both strands seems to code for the same or different genes), corrects putative indels to obtain the most probable ORF, and annotates the sequence with the description of the most similar subject with a reliable description (avoiding when possible the «unknown» or «predicted» tags). It works in parallelised or distributed systems and can be tested at http://www.scbi.uma.es/fulllengther2.