Universitat Politècnica de València
Escola Tècnica Superior D'Enginyeria Agronòmica i del Medi Natural

Integration of multi-omics data to describe link between developmental exposure to pesticides and impaired neurodevelopment

Biotechnology Bachelor's Thesis
(*Trabajo Fin de Grado en Biotecnología*)
Academic year 2015-2016

STUDENT: Elena Bernabeu Gómez
TUTOR: Sonia Tarazona Campos
EXTERNAL TUTOR: Ana Conesa Cegarra
EXTERNAL TUTOR: Vicente Felipo Orts

Valencia, June 2016

Title **-** Integration of multi-omics data to describe link between developmental exposure to pesticides and impaired neurodevelopment

Author – Dña. Elena Bernabeu Gómez

Academic tutor – Prof. Dña. Sonia Tarazona Campos

Co-tutor – Dña. Ana Conesa Cegarra

Additional co-tutor – D. Vicente Felipo Orts

Location and date – Valencia, June 2016

Abstract

Abstract – In recent years the omics disciplines have made their way across a wider spectrum of research groups, thus leading to the generation of multi-omics data sets in a great number of studies. While traditional omics studies only focused on a single biological level, the multi-omics approach has the potential of studying systems in further detail. However, along with this great potential comes the challenge of integrating and analyzing data far more complex in nature than that of a single omic discipline.

One of such multi-omics data integration challenges is exemplified by the EU-funded DENAMIC project, which investigated neurotoxic effects of low-concentration mixtures of pesticides and a number of common environmental pollutants in children using a *Rattus norvegicus* animal model. To meet this feat, several omics platforms were employed using brain tissue samples from rats treated with the aforementioned pollutants: proteomics, metabolomics and transcriptomics (RNA-seq and miRNA-seq). Clinical data in the form of learning tests was obtained prior to brain tissue sample extraction as well.

This project aimed to develop a strategy for the integration of multi-omics and clinical data for the DENAMIC experimental set up by means of creating multi-omic models regarding the neural response to toxic compounds, as well as to confirm previous conclusions of the DENAMIC project and to obtain new information about the global effect of pesticide developmental exposure at both molecular and physiological levels from a multi-omic point of view.

The strategy presented here tackles the different challenges of integrative analysis: First, the pre-processing of multi-omic data and the treatment of missing values; second, the establishment of potential associations between mRNAs, miRNAs, proteins and metabolites, while trying to filter out spurious associations to increase the mapping specificity; third, the visualization of these associations by mapping onto KEGG pathways, which allows the identification and study of relevant pathways and components for the various omics studied as well as their interactions; and finally, the association of molecular changes to phenotypic changes, as represented by the clinical data. The results obtained could potentially help locate markers of neurotoxicity and explain the molecular basis of impaired neurodevelopment.

Keywords – multi-omic data integration, multivariate statistics, transcriptomics, proteomics, metabolomics, bioinformatics, neurobiology, pesticide exposure

Resumen

Resumen – En los últimos años las ciencias ómicas se han hecho hueco en un espectro cada vez más amplio de grupos de investigación, lo que ha resultado en la generación de datos multiómicos en un gran número de estudios. Mientras que los estudios ómicos tradicionales se centraban en un único nivel biológico, el enfoque multi-ómico permite estudiar los sistemas en mucho más detalle. No obstante, este gran potencial viene acompañado del reto de interpretar y analizar datos de naturaleza más compleja que los de una única ómica.

Un estudio que ejemplifica el reto de la integración multi-ómica es el proyecto Europeo DENAMIC, que investigó los efectos neurotóxicos de mezclas de pesticidas a bajas concentraciones y de contaminantes ambientales comunes en niños, utilizando como modelo animal *Rattus norvegicus.* Para ello, se emplearon distintas plataformas ómicas utilizando muestras cerebrales de ratas tratadas con los pesticidas mencionados anteriormente: proteómica, metabolómica y transcriptómica (tanto RNA-seq como miRNA-seq). Además, se obtuvieron datos clínicos mediante tests de aprendizaje realizados con anterioridad al sacrificio de las ratas y extracción de tejido cerebral.

Este proyecto pretende desarrollar una estrategia de integración de datos multi-ómicos y clínicos para el diseño experimental del proyecto DENAMIC mediante la creación de modelos multi-ómicos relacionados con la respuesta neuronal a compuestos tóxicos, así como confirmar conclusiones extraídas en previas etapas del proyecto DENAMIC y obtener nueva información acerca del efecto global de la exposición a pesticidas durante el desarrollo a niveles tanto moleculares como fisiológicos desde un punto de vista multi-ómico.

La estrategia que presentamos aborda los retos de los análisis de integración: en primer lugar, el pre-procesado de los datos multi-ómicos y el tratamiento de los valores faltantes. En segundo lugar, el establecimiento de asociaciones potenciales entre mRNAs, miRNAs, proteínas y metabolitos, tratando de filtrar asociaciones falsas. En tercer lugar, la visualización de dichas asociaciones en rutas biológicas de KEGG, permitiendo la identificación y el estudio de rutas y componentes relevantes para las distintas ómicas así como su interacción. Finalmente, la asociación de los cambios moleculares con los cambios en el fenotipo representados por los datos clínicos. Los resultados obtenidos podrían servir para la identificación de marcadores de neurotoxicidad y para explicar las bases moleculares en un neurodesarrollo deficiente.

Palabras clave – integración de datos multi-ómicos, estadística multivariante, transcriptómica, proteómica, metabolómica, bioinformática, neurobiología, exposición a pesticidas

Resum

Resum – En els últims anys, les ciències òmiques s'han fet lloc en un espectre cada vegada més ampli de grups d'investigació, la qual cosa ha donat com a resultat la generació de dades multiòmiques en un gran nombre d'estudis. Mentres que els estudis òmics tradicionals se centraven en un únic nivell biològic, l'enfocament multiòmic permet estudiar els sistemes amb molt més detall. No obstant això, aquest gran potencial ve acompanyat del repte d'interpretar i analitzar dades de naturalesa més complexa que els d'una única òmica.

Un estudi que exemplifica el repte de la integració multiòmica és el projecte Europeu DENAMIC, que va investigar els efectes neurotòxics de mescles de pesticides a baixes concentracions i de contaminants ambientals comuns en xiquets, utilitzant com a model animal *Rattus norvegicus*. Per a això, es van fer servir distintes plataformes òmiques emprant mostres cerebrals de rates tractades amb els pesticides mencionats anteriorment: proteòmica, metabolòmica i transcriptòmica (tant RNA-seq com miRNA-seq) . A més, es van obtenir dades clíniques per mitjà de tests d'aprenentatge realitzats amb anterioritat al sacrifici de les rates i extracció de teixit cerebral.

Aquest projecte pretén desenvolupar una estratègia d'integració de dades multiòmiques i clíniques per al disseny experimental del projecte DENAMIC per mitjà de la creació de models multiòmics relacionats amb la resposta neuronal a compostos tòxics, així com confirmar conclusions extretes en etapes prèvies del projecte DENAMIC i obtenir informació nova sobre l'efecte global de l'exposició a pesticides durant el desenvolupament, tant a nivells moleculars com fisiològics, des d'un punt de vista multiòmic.

L'estratègia que presentem consisteix a abordar els reptes de les anàlisis d'integració. En primer lloc, el preprocessat de les dades multiòmiques i el tractament dels valors faltants. En segon lloc, l'establiment d'associacions potencials entre mRNAs, miRNAs, proteïnes i metabòlits, tractant de filtrar associacions falses. En tercer lloc, la visualització de les dites associacions en rutes biològiques de KEGG, permetent la identificació i l'estudi de rutes i components rellevants per a les distintes òmiques, així com la seua interacció. Finalment, l'associació dels canvis moleculars amb els canvis en el fenotip representats per les dades clíniques. Els resultats obtinguts podrien servir per a la identificació de marcadors de neurotoxicitat i per a explicar les bases moleculars en un neurodesenvolupament deficient.

Paraules clau – integració de dades multi-òmiques, estadística multivariant, transcriptòmica, proteòmica, metabolòmica, bioinformàtica, neurobiologia, exposició a pesticides

For all the friends I've made these past four years, whose existence was as lovely to me as incompatible with our group projects.

For Isabel and Gema, who I know in 50 years time will still be dancing to bad 90's pop hits with me while eating disgusting food.

For my family. Mom, thank you for force-feeding and taking care of me throughout all exam seasons these past four years. Dad, thank you for proving that you can be incredibly wise and still tell equally terrible jokes. Joan, for the last time, please turn that music down.

For Pablo. Chinese tonight?

Acknowledgements

I'd like to thank all the teachers that have helped me make it this far, both throughout my childhood as well as through my brief encounter with adulthood. Don't ever stop believing that your work is irrelevant or unimportant.

I'd also like to thank the Genomics of Gene Expression lab at the Príncipe Felipe Research Center. Twice already they've welcomed me with open arms, taking time off their busy schedules to help a confused Biotechnologist make her way into the Bioinformatics world.

I'd especially like to thank the wonderful and bright women that are Ana Conesa and Sonia Tarazona, whom without this project would not have been possible. From the very beginning they were supporting and understanding, all while challenging me to do my best. Thank you for all your time and all the knowledge shared.

General Index

Table Index

Figure Index

Abbreviations

ASCA	ANOVA simultaneous component analysis
bp	Base pairs
cDNA	Complementary DNA
CAR	Carbaryl
CB	Cerebellum
CHLOR	Chlorpyrifos
CIPF	Centro de Investigación Príncipe Felipe
CYP	Cypermethrin
CX	Cortex
DE	Differentially expressed
DENAMIC	Developmental Neurotoxicity Assessment of Mixtures in Children
DNA	Deoxyribonucleic acid
END	Endosulfan
EST	Expressed sequence tag
FPKM	Fragments Per Kilobase Mapped
GABA	Gamma-aminobutyric acid
Gb	Gigabytes
GC-HRTOF-MS	Gas Chromatography High Resolution Time of Flight Mass Spectrometry
HP	Hippocampus
HRTOF-MS	High Resolution Time of Flight Mass Spectrometry
Kb	kilobases
KEGG	Kyoto Encyclopedia of Genes and Genomes
LC-HRTOF-MS	Liquid Chromatography High Resolution Time of Flight Mass Spectrometry
mRNA	Messenger RNA
MS	Mass spectrometry
MWM	Morris Water Maze
NA	Missing value
NGS	Next-Generation Sequencing
PCA	Principal Components Analysis
PD	Parkinson's Disease
RNA	Ribonucleic acid
ST	Striatum
UTR	Untranslated region
VH	Vehicle/control substance

1. Introduction

1.1. The "omics" disciplines

High-throughput and large scale data began being recognized as a potent tool in research in the biological field with the first whole-genome sequencing effort, performed by Fleischmann *et al.* in 1995, of the bacteria *Haemophilus influenzae*. This led to the development of various omics disciplines, each focusing on a certain type of biomolecule within a sample. Nowadays, over 30 omics disciplines have been recognized, being metabolomics, proteomics, genomics and transcriptomics the most studied (Mayer, 2011). What makes the omics disciplines unique is the potential to study biological systems by gathering large amounts of data on as many targets as possible rather than focusing on a few elements at a single biological level.

In this project we focus on metabolomics, proteomics and transcriptomics, whose main difference when compared to the genomics discipline is that where an organism only has one genome, an organism can have more than one metabolome, proteome or transcriptome, due to differences between tissues or cells, developmental stages and environmental stimuli. This allows for the comparison of results between different conditions to identify genes, metabolites and proteins that are differentially expressed in distinct cell populations, or in response to different treatments, which in turn leads to a further understanding in organism-wide changes, the finding of new biomarkers for diseases and more.

1.1.1. Transcriptomics

Transcriptomics aims to study all RNA transcripts found in a sample, under specific circumstances or in a specific cell, using high-throughput technologies such as microarrays or RNA-seq. The key aims of transcriptomics include: cataloguing all species of transcript, including mRNAs, non-coding RNAs and small RNAs; determining the transcriptional structure of genes, in terms of their start sites, 5' and 3' ends, splicing patterns and other post-transcriptional modifications; and quantifying the changing expression levels of each transcript during development and under different conditions.

Various technologies have been developed to deduce and quantify the transcriptome, including hybridization or sequence-based approaches. Hybridization based approaches have several inconveniences, including reliance upon existing knowledge about the genome sequence, high background levels due to cross-hybridization and a limited range of detection due to both background and saturation of signals. Sequence-based technologies previously consisted on the use of Sanger sequencing of cDNAs and ESTs, but with the development of novel high-throughput sequencing technologies, a new method for both mapping and quantifying transcriptomes has since taken over, termed RNA-seq (short for RNA-sequencing) (Wang *et al.*, 2009).

RNA-seq is the application of any next generation sequencing (NGS) technique to study RNA. While the sequencing techniques are generally the same as those used in the study of DNA, the library

preparation and analysis widely differs between the two. For example, RNA-seq library preparation usually includes reverse transcription. Also, data analysis of RNA-seq may include transcript assembly, alternatively spliced transcript, or novel transcript discovery and transcript quantification (Chu & Corey, 2012).

Very briefly, the methodology employed by RNA-seq can be divided into the following steps: (1) A population of RNAs from the problem sample is converted to a library of cDNA fragments with adaptors attached to one or both ends. (2) Molecules are sequenced with the high-throughput technology of the researcher's choosing, such as the Illumina, Applied Biosystems SOLiD or the Roche 454 Life Science platforms, from one end or both ends (single-end sequencing and pair-end sequencing respectively). (3) Reads are aligned to a reference genome or reference transcripts, or assembled *de novo* without the genomic sequence. (4) A genome-scale transcription map that consists of both the transcriptional structure and/or level of expression for each gene is created. This methodology is represented graphically in Figure 1.

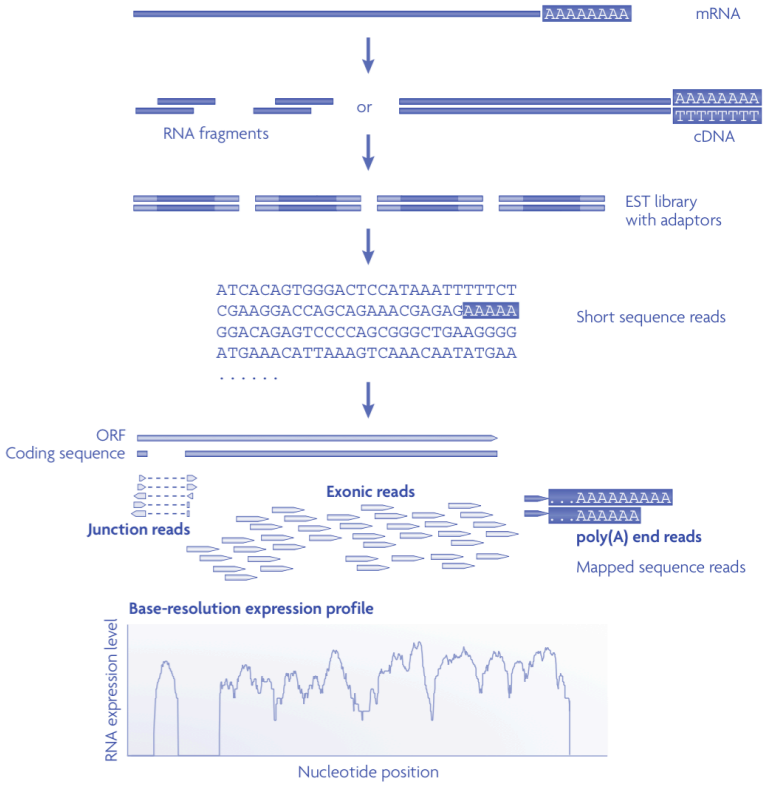


Figure 1. A typical RNA-seq experiment (Wang et al. 2009).

MicroRNA-seq (also known as miRNA-seq) is a type of RNA-seq in which small-RNAs are enriched so as to study the population of microRNAs in a sample using NGS technologies. MicroRNAs (miRNAs) are a family of 21–25-nucleotide small RNAs that negatively regulate gene expression at the post-transcriptional level (He & Hannon, 2004). The mode of action of the mature miRNA in mammalian systems is dependent on complementary base pairing primarily to the 3'-UTR region of the target mRNA, thereafter causing the inhibition of translation and/or the degradation of the mRNA (Picardi, 2015). Due to the important regulatory function that these molecules have, miRNA-seq is implemented in transcriptomics analyses in many research projects.

1.1.2. Metabolomics and Proteomics

Proteomics is the large-scale study of proteomes, known as a set of proteins produced by an organism, system or biological context. The proteome is associated with the underlying transcriptome, but protein expression is also modulated by many other factors, thus the importance of its study. Metabolomics can be defined as the systematic study of the chemical fingerprints that cellular processes leave behind, in other words, the study of their small-molecule metabolite profiles. This is also known as the metabolome.

Several high-throughput technologies have been developed to analyze proteomes and metabolomes, being the most prominent mass spectrometry (MS) based techniques and gel-based techniques (Griffiths & Wang, 2009).

1.2. Multi-omics integration

Advances in high-throughput sequencing methods, mass spectrometry, computational power and algorithmic have coordinately led to a higher ability to acquire multivariate datasets from different omics disciplines due to increased efficiency and ease of use as well as decreased costs.

When studying multi-omics data, besides a difference inherent to the biology and biochemistry of each sample, discrepancies can also be found in the results that each omics discipline contributes, which depend on the technical evolution of the instruments used, each carrying a set of advantages and disadvantages. This often leads to the acquirement of scattered information across the omics, each bound to miss part of the complexity of a biological system. This has led to an increased interest in the integration of multiple complementary components, interactional or functional states datasets, leading to a practice known as the integration of multi-omics, whose goal is to obtain a more holistic and complete picture of a biological system (Assche *et al.*, 2015). To gain a bigger picture of the system considered, taking into account the inherent relationships between the different biological levels represented by each omic discipline can prove useful. The most well known relationship is the central dogma of molecular biology, which considers the transformation from genes to transcripts and from transcripts to proteins. However, more relationships exist, as portrayed in Figure 2 (Buescher & Driggers, 2016).

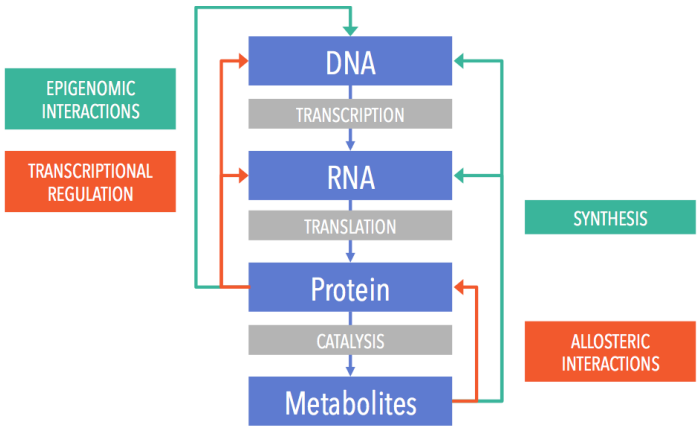


Figure 2. Relationship between omics disciplines beyond the central dogma of molecular biology.

The integration of multi-omics data has not proven to be an easy task. Its complexity lies not only in the homogenization of data from different omics sources for comparison, but also in the lack of efficient statistic methods that permit a correct analysis. To battle these challenges, several consortiums and projects have been created, including the FP7 STATegra project (Stategra.eu, 2016), lead by the Genomics of Gene Expression Lab at the Príncipe Felipe Research Center (CIPF), where the efforts reflected in this work were completed.

1.3. The DENAMIC project

In recent years, concern has arisen regarding the association between an increased incidence of learning and developmental disorders in children and pesticides. Studies have shown that the developing central nervous system is often more vulnerable than the adult one. This is due to the fact that the blood-brain barrier, the immune system and the detoxification system are not completely matured until later on in life, thus making a child more exposed to toxic agents and insecticides (Giordano & Costa, 2012; Llop *et al.*, 2013).

Many of the almost 200 chemicals known to be neurotoxic are developmental neurotoxicants, thus their ability to contribute to a variety of neurodevelopmental and neurological disorders when exposed *in utero* or during childhood. Furthermore, they could also lead to silent damage diseases that only manifest as the individual ages, and include Parkinson's and Alzheimer's (Giordano & Costa, 2012). Sadly, this exposure is fairly common, as insecticides are widely used in the domestic setting, even during pregnancy or in the presence of school-age children, according to studies conducted in several countries. Just considering the Spanish population, 54% of pregnant women of the INMA (Environment and Childhood) cohort were listed as using at least one type of pesticide (Llop *et al.*, 2013).

In light of this situation, the European commission-funded project DENAMIC (Developmental Neurotoxicity Assessment of Mixtures in Children) was born, with the objective of developing methods and obtaining results that could guide risk management in the European Union (EU) and World Health Organization (WHO) as well as support the EU chemicals legislation for identifying potential neurotoxicants. During its run from January 2012 to December 2015, this project investigated neurotoxic effects of low-concentration mixtures of pesticides and a number of common environmental pollutants in children, focusing on effects on learning (cognitive skills) and developmental disorders (such as Attention Deficit Hyperactivity Disorder (ADHD), autism spectrum disorders and anxiety disorders). The DENAMIC project was made up of a unique consortium of 13 partners of numerous universities, research institutes, and small and medium enterprises (SME's) with extensive knowledge of a wide range of technologies, such as chemistry, toxicology and omics techniques (Denamic-project.eu, 2016).

Amongst the many partners that this project had, the Laboratory of Neurobiology at the Centro de Investigación Principe Felipe (CIPF), led by Dr. Vicente Felipo, was at the center of a collaborative work between several of the project's members, whose objective was to assess the effect of 4 different pesticides on the neurodevelopment of the animal model *Rattus norvegicus,* based on learning tests which evaluated motor and cognitive skills, as well as the analysis of various brain tissue samples by several omics disciplines (proteomics, metabolomics and transcriptomics).

Having analyzed the different omics individually, the Neurobiology lab wanted to obtain a more thorough look at the results obtained through an integrative multi-omics effort, thus reaching out to the Genomics of Gene Expression Lab, beginning a new scientific collaboration on which this work is based.

To understand how the data was obtained, and where this work fits into the global scheme of the DENAMIC project, the details regarding the experimental set-up leading up to this work are explained in the following section.

1.3.1. Experimental design

The animal model used was *Rattus norvegicus*, more precisely, pregnant Wistar rats. In charge of carrying out all tasks associated with the care and study of the experimental animals was Dr. Felipo's Neurobiology lab. To assess the effects of pesticide exposure on the offspring, the pregnant rats were treated with 4 different pesticides (as well as combinations of them and a control) starting on the seventh day of gestation through the end of their pregnancy. Once the offspring was born, the mothers continued being treated up until post-natal day 21, the babies receiving the pesticides through the mother's milk. From post-natal days 60 to 75 these rats went through several learning tests to assess their cognitive and motor abilities. Finally, on post-natal day 90 the offspring litter was sacrificed and brain samples from the hippocampus, cortex, striatum and cerebellum were obtained for each rat. These samples were later sent to different labs, each specialized in a different field, for a thorough omic analysis (Figure 3). The UK-based company Proteome Sciences was assigned the task of performing a proteomic analysis on the samples, while the Vrije Universiteit Amsterdam (VU) was in charge of the metabolic analysis. Finally, the Spanish company Imegen was sent brain samples for the transcriptomic analysis. Having received most of the data from the corresponding entities, the Genomics of Gene Expression lab at the CIPF began working on the integration of the individual omics disciplines. These efforts are the topic of this project.

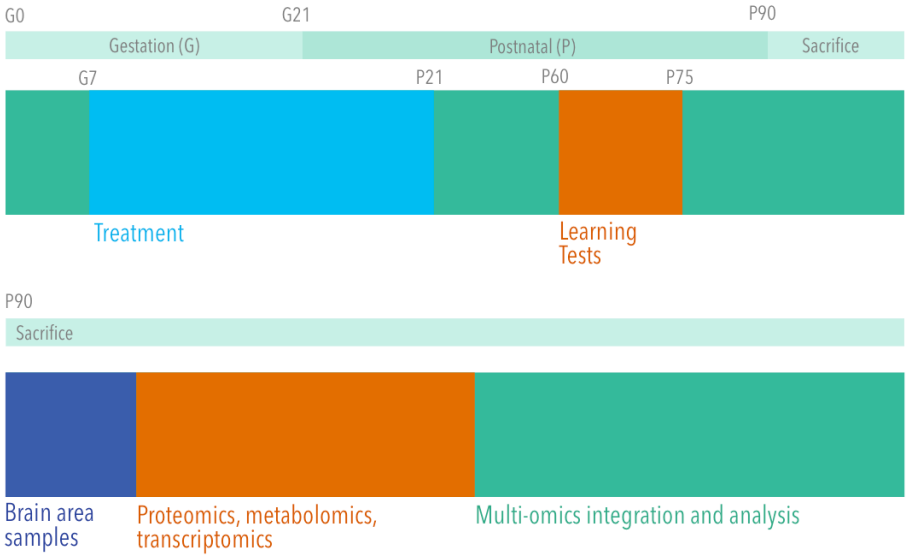


Figure 3. Experimental timeline. The upper axis represents the moment in time (G: gestation day, P: post-natal day) and the lower axis the event that was carried out at that moment. Explanation found in text.

The pesticides employed in the treatment of the pregnant Wistar rats include endosulfan, chlorpyrifos, carbaryl and cypermethrin. These were chosen due to their representativeness of a larger group of pesticides commonly found in our environment.

Due to the impracticality of maintaining live rats in large numbers confined in a limited space, the work was carried out in batches (or sets) of rats. Each set corresponds to a different moment in time, and the rats corresponding to each set were treated simultaneously. Each set consists on a combination of offspring rats born from pregnant Wistar rats treated with several of the aforementioned pesticides, and for each rat different brain tissues were obtained. The so-called Set 01, 03 and 10 were destined towards proteomics (Sets 01, 03 and 10) and metabolomics (Sets 01 and 10, the latter's data pending arrival) studies. For the transcriptomics analysis, different rats were used; samples were obtained from rats corresponding to 3 different experimental batches. However, for ease of understanding, we will refer to this combination of rat batches as the transcriptomics set.

Due to the fact that the main objective of this work was to integrate different omics data, out of the 3 sets destined towards proteomics and metabolomics it was decided to continue this analysis only for the data corresponding to Set 01, for which both metabolomics and proteomics data was available. Therefore, only Set 01 and the transcriptomics set were considered in this integrative effort, even though Sets 03 and 10 will be used in the future. Details regarding these two sets of samples are found in Tables 1 and 2. Set 03 and 10 details can be found in Attachment XII.

Set 01 contains rats treated with the vehicle/control substance (VH), endosulfan (END), and cypermethrin (CYP). For all these rats, 4 tissue samples were obtained for the brain areas hippocampus (HP), cortex (CX), cerebellum (CB) and striatum (ST). Samples destined towards transcriptomics, on the other hand, sometimes consisted on a pool of rat sub-samples, thus complicating later associations with scores obtained by the rats in the learning tests. Also, only tissue samples for the HP and CB were obtained. The treatments performed on these include the vehicle substance (VH), endosulfan (END), and a combination of cypermethrin and endosulfan (CYP + END).

The learning tests performed valued both cognitive and motor abilities, and they included the Radial Maze test (Olton & Samuelson, 1976), the Morris Water Maze (MWM) test, the Beam Walking test and the Rotarod test (Jones & Roberts, 1968) (Figure 4). The Radial Maze and the Morris Water Maze (MWM) tested cognitive skill, while the Beam Walking and Rotarod tested value motor skill.

Metabolomics samples were subjected to targeted and untargeted cross platform metabolomic approaches using LC-HRTOF-MS, GC-HRTOF-MS, and shotgun HRTOF-MS (lipids). Proteomics samples were homogenized and labeled with TMT6plex or TMT10plex reagents. MS were acquired in the Orbitrap and TOP10 MS/MS were acquired for peptide identification and quantitation. For MS2 quantitation, HCD-MS2 was acquired in the Orbitrap. For MS3 quantitation, MS2 was acquired in the ion trap using CID for identification. MS2 fragments ions were selected for further HCD-MS3 quantitation in the Orbitrap. MS data was processed with Proteome Discoverer. This proprietary TMT-MS3 discovery proteomics workflow enables accurate profiling of any sample set for as many proteins as possible.

The samples pertaining to the transcriptomics set were sequenced using RNA-seq and miRNA-seq procedures. For RNA-seq, the Illumina HiSeq2000 platform was employed, obtaining 100b paired-end reads. For miRNA-seq, the Illumina HiSeq2500 platform was employed, obtaining 51b non-paired-end reads.

Table 1. Set 01. Colors correspond to the different sexes, green for Females and orange for Males. Each row represents a treatment. The first four columns represent the 4 tissues, and the numbers correspond to the number of samples obtained per sex and per treatment. The last column indicates the number of rats used per treatment (male or female), or the number of samples per tissue. Finally, the total number of rats for the set is indicated in the bottom row.

	Tissues								F M
Treatment	CB		HP		CX		ST		TOTAL / TISSUE
VH	3	4	3	4	3	4	3	4	7
END	4	6	4	6	4	6	4	6	10
CYP	4	4	4	4	4	4	4	4	8
							TOTAL RATS:		25

Table 2. Transcriptomics set. Same as Table 1 except for the last column, which represents the total number of samples for each treatment. The final row represents the number of samples employed total (36).

	Tissues				F M
Treatment	CB		HP		TOTAL SAMPLES
VH	3	3	3	3	12
END	3	3	3	3	12
END+CYP	3	3	3	3	12
					36

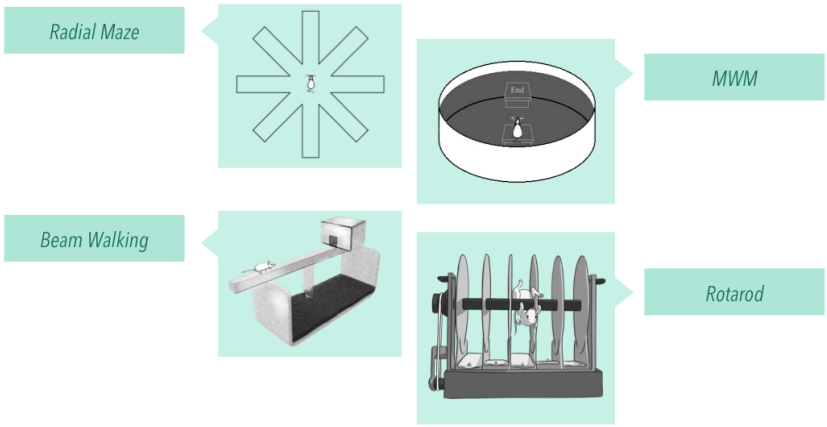


Figure 4. Learning tests performed on rats prior to sacrifice.

2. Objectives

This work's main objective is analyzing the data obtained from multiple omics disciplines through their integration to understand the molecular basis of impaired neurodevelopment due to developmental exposure to pesticides.

To do so, data will be pre-processed to minimize noise not due to biological causes. An exploratory analysis will accompany the pre-processing stage, allowing for a close observation and surveillance of how changes affect the data. A differential expression analysis will then be performed in order to gain a better understanding of the changes occurred in each individual omic. Finally, the data will be integrated as a new source of information in the quest for finding relevant pathways and components that could potentially act as biomarkers of neurotoxicity and explain the molecular basis of impaired neurodevelopment.

Therefore, the main objective can be divided into several milestones:

1. Manage data coming from different sources and of different natures, formatting accordingly for its future use.

2. Develop a pre-processing and exploratory analysis pipeline for all the omics disciplines provided.

3. Perform a differential expression analysis for all the omics disciplines provided.

4. Integrate data by visualizing changes in omics levels on biological pathways using the Paintomics tool.

5. Relate these results with values obtained in learning tests and previously published literature to gain a better understanding of how pesticides affect neural function after their exposure in developmental stages.

3. Materials and Methods

3.1. The DENAMIC data

3.1.1. Proteomics, Metabolomics and Transcriptomics data

Data for numerous rat sets was supplied. However, as mentioned in the Introduction, only data pertaining to Set 01 was considered in this effort, as it was the only set with metabolomics and proteomics data, as well as the transcriptomics set.

Proteomics

The proteomics data came in the form of a relative quantification for all sets. According to the protocol provided by Proteome Sciences, peptide measurements for each sample were median-scaled, and then a ratio was calculated with this value relative to the average measurement for the references used by the technology. This ratio was then log2-transformed. To obtain quantitative protein data instead of peptide data, the peptide matrix was transformed into a protein matrix (log2-ratios of the peptides median-summarized to a protein ratio in log2-scale). Missing values indicate that a protein was not detected.

Out of the proteomics data received, only Set 10 (not considered in this effort) was pre-processed. However, as the idea is to include Set 10 in the integration analysis as soon as all data is available, we tried to homogenize the pre-processing methodology as much as possible to compare results for all Sets in the future in a more accurate manner. Therefore, we decided to apply the pre-processing strategies used by the Proteome Sciences team on Set 10 across all data sets, as described in section 3.4.2.

Metabolomics

The metabolomics data came in the form of an absolute quantification of those metabolites detected by the technology employed (LC-HRTOF-MS and GC-HRTOF-MS). According to the metabolomics team, the values provided pertain to the highest peak divided by the sample weight used for the analysis (normalization by wet weight). Missing values indicate that the metabolite was not found. This is not equivalent to 0 as the limit of detection depends on the compound.

Transcriptomics

We obtained miRNA-seq and RNA-seq data in the form of FPKMs per isoform. Sequencing reads had already undergone quality control using FASTQC software, used to evaluate quality distribution across reads, GC content, the presence of adapters, indexes and/or over-represented reads. Afterwards trimming was performed on all bases or complete reads that did not follow the pre-

requisites set. Furthermore, reads were aligned to the reference genome using TopHat2 and expression was quantified using Cufflinks.

3.1.2. Clinical data

As mentioned in the Introduction, learning tests were performed on our animal model prior to their sacrifice in order to assess their motor and cognitive capabilities. The same tests were performed on the rats independently of the Set to which they pertained. However, due to time constraints not all rats had these tests performed. All tests returned a quantitative value:

- For the Morris Water Maze, the value corresponds to the escape latency on Day 3 (larger values corresponding to lesser cognitive abilities).
- For the Rotarod test, the value corresponds to the time the rat was able to keep itself atop the device. Thus, higher values correspond to higher motor coordination abilities.
- For the Beam Walking test, the value corresponds to the number of faults performed by the rat, thus a higher value indicates lesser motor abilities.
- For the Radial Maze test, the value corresponds to the working and reference errors, so the higher the value the lower the cognitive skill.

3.1.3. Data to analyze and integrate

The basic exploratory analysis and pre-processing for each omic was performed on all data from Set 01 and the transcriptomics set. However, for the integration effort we could only use those tissues and pesticides in common for Set 01 and the transcriptomic set. This means that only the data pertaining to the treatment with endosulfan (END) and the vehicle or control (VH), as well as only the tissues hippocampus and cerebellum were considered (Figure 5).

In regards to the clinical data, only Set 01 is considered, as the transcriptomics set presented ambiguities in the matter of the rats from which each sample came from, thus establishing a difficulty in associating them to their learning test scores.

3.2. Platform and software

This analysis has been performed on a Macbook Pro running Mac OS X with a 2,3 GHz Intel Core i7 processor, 16GB 1600 MHz DDR3 memory, and an NVIDIA GeForce GT 750M 2048 MB graphics card.

When coming across omics data analyses, heavy-duty statistics tools are required. R (Ihaka & Gentleman, 1996) is a system for statistical computation and graphics, and it consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files (R-project.org, 2016). The bioinformatics community has embraced R as an essential player when coming across such analyses due to its wide array of statistics tools available by default, as well as the large support and user community surrounding the system, with a large development of open-source external implementations. These implementations, known as packages or libraries, are stored and organized in public repositories, being Bioconductor the most prominent in the Bioinformatics and Biomedical field. The

Bioconductor (Bioconductor.org, 2016) project is an initiative for the collaborative creation of extensible software for computational biology and bioinformatics.

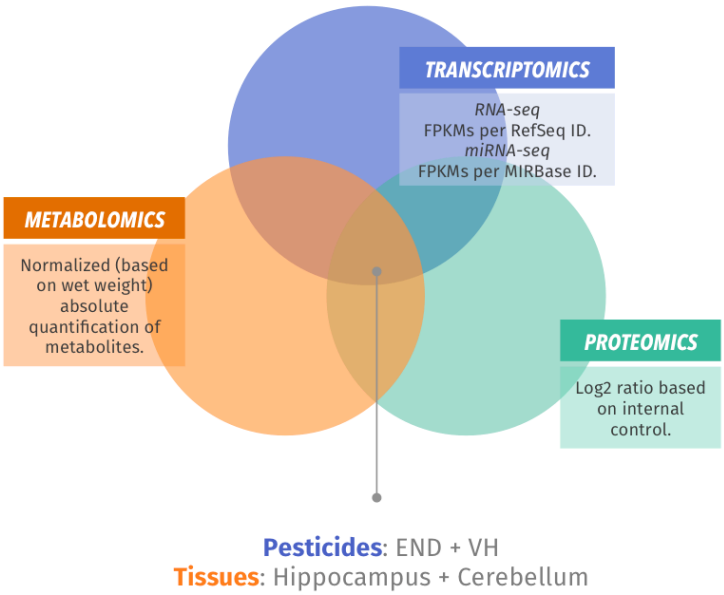


Figure 5. Summary of data provided. Arrow indicates intersection of the 3 omics in regards to tissues and pesticides.

Several graphic interfaces have been developed for this programming language, being RStudio (Racine, 2011) one of the most popular due to its efficiency, productivity and the possibility of being used in any operating system.

For all these reasons, most of this effort was performed in R. All scripts were programmed and run in R version 3.2.1 under the RStudio version 0.98.1102 graphic interface. Scripts can be found in section 7.1.

Paintomics v3.0 (Bioinfo.cipf.es/paintomics, 2016), a web based tool explained in further detail in section 3.5, was also used and was the basis of the integration step.

3.3. Statistical tools

Throughout this effort, data analysis and visualization tools of various natures and complexities were employed, some of the simplest ones including boxplots, histograms or Venn diagrams. In this section some of the most important ones are described.

3.3.1. Principal Components Analysis (PCA)

Principal components analyses (Pearson, 1901) were used in order to identify patterns in the omics data, by expressing it in such a way as to highlight their similarities and differences by creating a new set of coordinates to represent both samples and features (proteins, genes and metabolites). Since patterns can be hard to find in data of high dimension, where graphical representation is not

available, PCA is a powerful analysis tool. Briefly, PCA is a procedure for identifying a smaller number of uncorrelated variables, called "principal components", from a large set of data. The goal of a PCA is to explain the maximum amount of variance with the fewest number of principal components, thus allowing for the reduction of dimensions in the data. The number of principal components is less than or equal to the number of original variables. This transformation is defined in such a way that the first principal component accounts for as much of the variability in the data as possible, and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. PCAs were applied to all our omics data once it had been pre-processed to assess if there were any clear patterns separating samples differing in treatment, sex or tissue. However, they were also applied throughout the pre-processing pipeline in order to survey the effects of the changes applied.

3.3.2. Heatmaps

Heatmaps were also used for the visualization of the metabolomics and proteomics data, being less useful in regards to transcriptomics due to the excessive number of genes and miRNAs, which limited their use to only those differentially expressed features. Heatmaps are essentially a color map that allows samples to be grouped by a hierarchical clustering according to the correlation between them, the color representing the expression value for a certain feature (in this case, genes, proteins or metabolites). In this biological case study, heatmaps were used to find groups of features that could be potentially associated as shown by matching expression patterns throughout the samples and vice-versa, in order to find groupings of samples corresponding to changes in sex, tissue and treatment.

3.3.3. Wilcoxon rank sum tests

The Wilcoxon rank sum test (also known as the Mann-Whitney test) is the non-parametric alternative to Student's t-test when the data does not follow a normal distribution. The null hypothesis to be tested is that the distributions of the two samples compared differ by a location shift.

Wilcoxon rank sum tests were employed to compare clinical data values between different treatments (endosulfan and the vehicle substance), in order to find significant differences between the two to declare whether said treatment could potentially lead to a change in motor or cognitive skills. This test was chosen over Student's t-test as it could not be assumed that the clinical variables were normally distributed.

3.3.4. Pearson correlation coefficient

Another important statistical tool used was the Pearson correlation coefficient. In statistics, the Pearson product-moment correlation coefficient is a measure of the linear relationship between two variables X and Y, with a value between +1 and −1 inclusive, where 1 is total positive correlation, 0 is no correlation, and −1 is total negative correlation. This coefficient was employed in order to find a relationship between the different clinical data values and the expression of the DE proteins, metabolites, genes and miRNA, so as to find potential biomarkers or participants of what had been declared as significant changes between the pesticide treatment and the control in motor and cognitive skill by the Wilcoxon rank sum tests. To visualize these results correlation

plots were created using the *corrplot()* function from the *corrplot* R package (Cran.r-project.org, 2016).

3.4. Data preparation

3.4.1. Data formatting

Due to the difference in sources and participants in the omics disciplines, sample IDs were not homogeneous across the different data sets provided for the multi-omics integration, nor were they easily accessible for analysis. That is why the first step in this work was to organize and homogenize the data, to easily and intuitively use it to the work's purpose.

Since metabolomics and proteomics data corresponded to the same samples, the different excel files provided were transformed into a single tab-delimited file, as shown in Figure 6. The clinical data was also included for each sample ID, to ease future comparisons between learning tests and the omics. miRNA-seq and RNA-seq data was also re-formatted for ease of use.

RNA-seq IDs came in the form of RefSeq transcripts. Due to the more global use of ENSEMBL IDs as well as their need in future steps regarding the integration of gene expression (as described in section 3.5), an ID change for the different transcripts was performed, in which each RefSeq transcript returned by the RNA-seq effort was transformed into its corresponding ENSEMBL Gene ID. Due to the fact that more than one transcript can be associated to a single gene ID, when this happened the median of FPKMs for the different transcripts was calculated. As a reference for this ID change, the Biomart database was downloaded for all known genes in *Rattus norvegicus* (Ensembl.org, 2016).

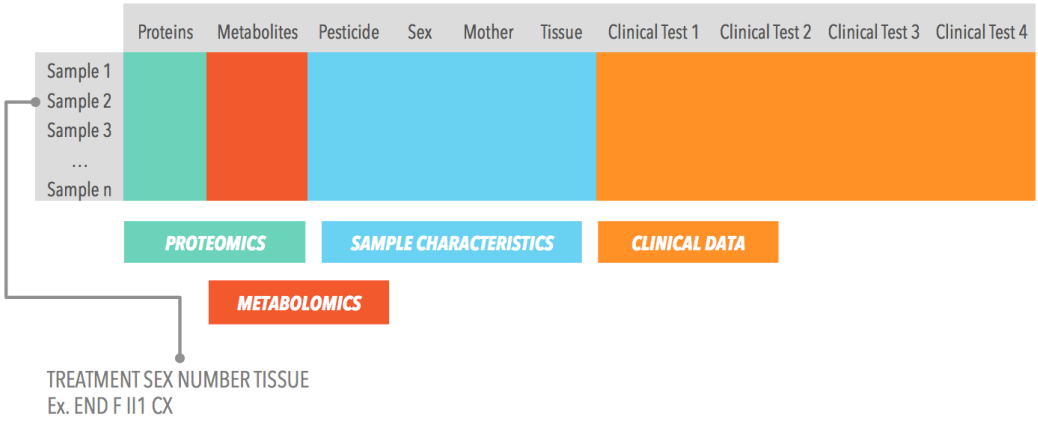


Figure 6. Representation of metabolomic and proteomic data formatting.

3.4.2. Data pre-processing

When coming across large data sets, the pre-processing step is often overlooked, when in truth it is one of the most important and defining steps in data analysis, on which the end-result and the obtaining of conclusions can depend. The methodologies employed have to be tailored to the data received, and many times this can result in a trial and error situation.

In this section, the final implemented pre-processing pipeline for this effort is explained. Worth noting is that this was not a pre-defined pipeline, meaning that observations made through exploratory analyses in the pre-processing stage guided it in certain ways. Some of the defining observations found throughout this process are represented in the Results and Discussion section. This methodology is also graphically represented in Figure 7 for ease of understanding.

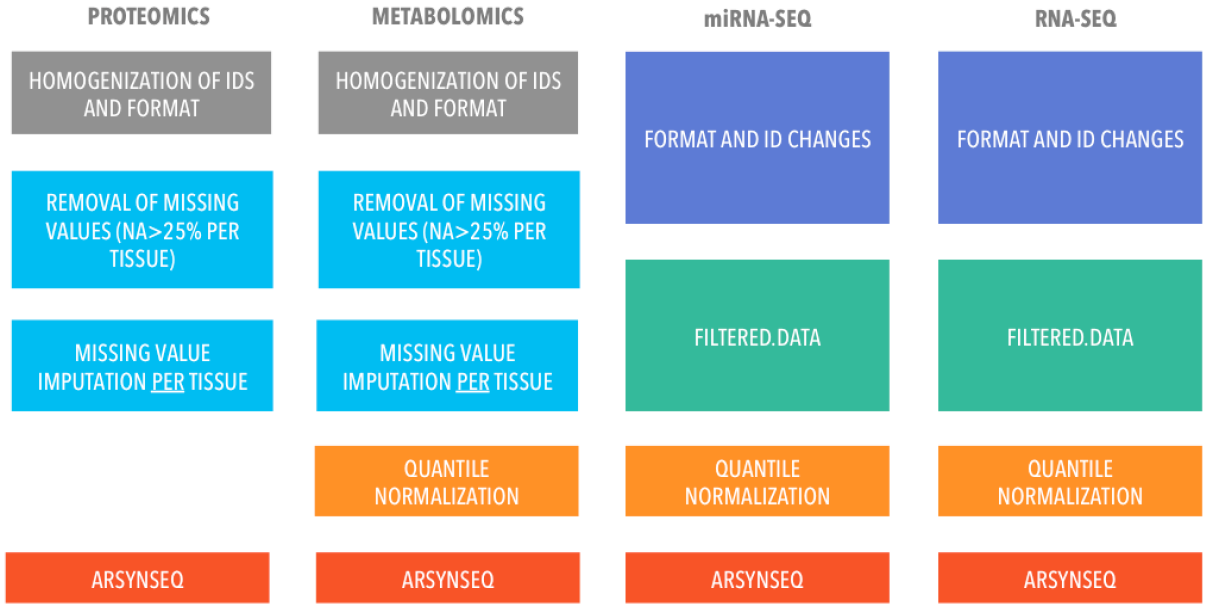


Figure 7. Formatting and pre-processing outline for the different omic disciplines.

Missing value treatment

As mentioned previously, the pre-processing methodology followed for both proteomics and metabolomics as well as the treatment of missing values closely resembles that performed by Proteome Sciences on Set 10, as described in their analysis report for the proteome analysis of the former. The next step for these two disciplines involved the removal of proteins and metabolites with over 25% missing values (NAs) for all samples. This removal was performed per tissue to preserve as many features as possible, characteristic of certain cell lines and not necessarily shared by others.

While analyzing the data, a sample was found to be missing (END M II2 – HP). Also, one cortex sample was removed in the metabolomics analysis prior to pre-processing due to its high content in NAs. After the first formatting efforts were performed, the resulting data consisted on the features depicted in Table 3 for metabolites and proteins corresponding to Set 01. After the removal of NAs, the data was left looking as depicted in Table 4 and 5.

In order to make the most out of the data provided, the remaining NAs were then imputed. Imputation methods involve replacing missing values with estimated ones based on information available in the data set, avoiding as much bias as possible and preserving the representativeness of the results. The data was imputed using the K-nearest neighbor (knn) imputation method (in R, the function *knn()* from the *impute* R package was used), being k = 2 (Altman, 1992; Hastie *et al.*, 2016). This imputation uses the k-nearest neighbors to fill in the unknown values in a data set. For each case with any NA value it will search for its k most similar cases and use the values of these

cases to fill in the unknowns. The function will use a weighted average of the values of the neighbors, which are given by the exponential Euclidean distance between the case with NAs and the neighbor k.

Table 3. Original data features. Proteomics in light blue and metabolomics in dark blue.

	samples	*proteins*	*samples*	*metabolites*
ALL TISSUES	99	3546	98	188
CORTEX	25	3546	24	188
HIPPOCAMPUS	24	3546	24	188
STRIATUM	25	3546	25	188
CEREBELLUM	25	3546	25	188

Table 4. Proteomics data after removal of NAs.

	Before removal ***Samples***	*After removal* ***Samples***	*Before removal* ***Proteins***	*After removal* ***Proteins***
CORTEX	25	25	3546	836
HIPPOCAMPUS	24	24	3546	797
STRIATUM	25	25	3546	829
CEREBELLUM	25	25	3546	820

Table 5. Metabolomics data after removal of NAs.

Before removal ***Samples***	*After removal* ***Samples***	*Before removal* ***Metabolites***	*After removal* ***Metabolites***
24	24	188	113
24	24	188	95
25	25	188	91
25	25	188	112

Filtering low expressed genes

It has often been argued that when quantifying gene expression with high-throughput sequencing techniques such as RNA-seq or miRNA-seq, expression estimation for low count genes is less reliable because read counts could have been assigned by chance (McIntyre *et al.*, 2011). Thus, excluding features with low counts may improve the results of statistical analyses because the level of noise is reduced.

In this work, the CPM (Counts Per Million) strategy in the *filtered.data()* function from the *NOISeq* R package (Tarazona *et al.*, 2012) was employed in order to eliminate the low-count features in RNA-seq and miRNA-seq. The CPM method removes those features that have an average expression per condition less than 1 CPM (1 FPKM in our case).

After performing the RNA-seq ID change (which resulted in a reduction in the number of genes total left for analysis due to the missing equivalencies between databases as well as the fact that various mRNAs pertained to a single ENSEMBL gene ID) as well as filtering, the results were left as represented in Table 6.

Table 6. Transcriptomics features before and after filtering.

	miRNA CB	*miRNA HP*	*RNA CB*	*RNA HP*
Before	1133	1133	15717	15717
After	786	789	10604	10693

Normalization

Generally, normalization is an essential step in omics analyses because it helps reduce the potential biases of the technology and makes the samples and features comparable. Two types of normalization may be considered: within-sample and between-sample. An example of the need for within-sample normalization is the gene length bias in RNA-seq, given that longer genes tend to obtain a higher number of read counts and hence a higher estimation of their expression level. FPKM normalization is one of the many methods aiming to reduce this length bias. On the other hand, between-sample normalization corrects systematic differences among samples that are not due to biological effects. In RNA-seq, for instance, genes in samples with higher sequencing depth (number of reads sequenced) tend to get a higher expression quantification, and the comparison of the gene expression across samples is not fair. There are plenty of methods that correct for this bias, being FPKM or TMM some examples.

A between-sample normalization approach that is not specific for RNA-seq but can be used for any type of data is the quantile normalization. This technique makes the distribution of values of the different samples identical in statistical properties, meaning that the quantiles of each distribution will be identical and therefore the samples will be comparable. Due to anomalies observed in the data, as described in the Results and Discussion section, a quantile normalization was performed on the metabolomics, miRNA-seq and RNA-seq data.

Noise reduction

When a batch effect is detected in the data or the samples are not properly clustered due to an unknown source of technical noise, it is usually appropriate to remove this batch effect or noise before proceeding with the statistical analysis.

ARSyNseq (ASCA Removal of Systematic Noise for sequencing data) is an R function implemented in the *NOISeq* R package (Tarazona *et al.*, 2015) that is designed to filter the noise associated to identified or unidentified batch effects. The ARSyN method (Nueda *et al.*, 2011) combines analysis of variance (ANOVA) modeling and multivariate analysis of estimated effects (PCA) to identify the structured variation of either the effect of the batch (if the batch information is provided) or the ANOVA errors (if the batch information is unknown). As ARSyN was initially designed for microarray data, it was adapted into ARSyNseq to be used on sequencing data. Thus, ARSyNseq returns a filtered data set that is rich in the information of interest and includes only the random noise required for inferential analysis. In this work, we applied ARSyNseq to all the omics in order to reduce the systematic noise, as shown in the Results section.

3.5. Integration: Paintomics

To integrate the biological case's multi-omics data, Paintomics (García-Alcalde *et al.*, 2011) was used, which is a web-based tool developed by the Genomics of Gene Expression lab for the integrative visualization of multiple omic datasets onto Kyoto Encyclopedia of Genes and Genomes (KEGG) pathways. Paintomics v3.0 differs from its predecessors in that it is able to accept widely diverse data, including common techniques such as Transcriptomics, Metabolomics and Proteomics, but also emerging approaches such as DNase-seq, ChIP-seq or Methyl-seq, when in the past it could only accept gene and metabolite expression data.

The KEGG (Kanehisa, 2000) Pathway database is a collection of pathway maps integrating many entities including genes, proteins, RNAs, chemical compounds, glycans, and chemical reactions, as well as disease genes and drug targets, which are stored as individual entries in the other databases of KEGG. By mapping the features introduced into Paintomics onto these pathways, the user can gain a more thorough, visual and integrated look at the events that occur inside the problem organism.

Paintomics works in 3 main steps: data upload (it requires data matrices containing feature quantification data values as fold-changes as well as a list of significant features for each omic), ID and name matching as well as metabolite assignment, and finally pathway selection.

KEGG pathways mainly consist on genes and metabolites. Gene-based data, such as RNA-seq or proteomics, is imputed back to an origin gene, which codifies for the corresponding mRNA and protein. Metabolite-based data is imputed to a corresponding metabolite. Worth noting is that Paintomics is unable to accept miRNA data directly, needing as an input miRNA targets in order to impute them to a gene. The current version requires the miRNA IDs to be substituted by their targets' ENSEMBL IDs by the user. In the future, however, Paintomics will perform this step automatically, asking the user for a list of miRNA-target associations as well as their miRNA-seq data.

IDs need to be given in a certain format. Metabolites can be recognized by name, proteins by their Uniprot ID or PDB ID and mRNAs (and miRNA targets) by their ENSEMBL or Entrez Gene ID. In order to map features onto KEGG pathways, all gene-based data will be translated by Paintomics to its corresponding Entrez Gene IDs, while metabolites will be mapped directly.

To decide which pathways are significantly altered or not, and for which omics, Paintomics uses the significant features lists provided by the user. Briefly, for each omic Paintomics compares the list of significant features mapped to a pathway with the rest of the features also mapped to that pathway. As final step, the tool computes the significance of the overlap using Fisher's exact test (Fisher, 1922). The obtained p-value can be interpreted as a measure of the confidence that this overlap is due to chance. In order to obtain the joint significance for all omics combined for a certain pathway, Paintomics uses Fisher's combined probability test (Fisher, 1938), a statistical method that allows combining the results from several independent tests for similar null hypotheses. This method combines the p-values for each test into one test statistic (X) using the formula:

$$X = -2\sum_{i=1}^{k} \ln(p_i)$$

where p_i is the p-value for the ith hypothesis test, k is the number of tests being combined and with X following a $\chi 2$ distribution with 2k degrees of freedom, from which a p-value for the global hypothesis can be easily obtained.

What Paintomics returns is a list of all the KEGG pathways to which the data has been mapped, highlighting significant omics (or a combination of them) for each pathway (p-value < 0.05). Each pathway can then be visualized individually, where the changes for each mapped component can

be observed for the different experimental conditions. Significant features are highlighted using a thicker black line around the component on the corresponding KEGG pathway.

Paintomics and DENAMIC

Paintomics was originally designed to easily visualize time-based omics studies. However, other experimental conditions can be visualized depending on the format of the input. The format chosen for this biological case is represented in Figure 8, allowing the user to visualize changes between two different tissues and the two different sexes.

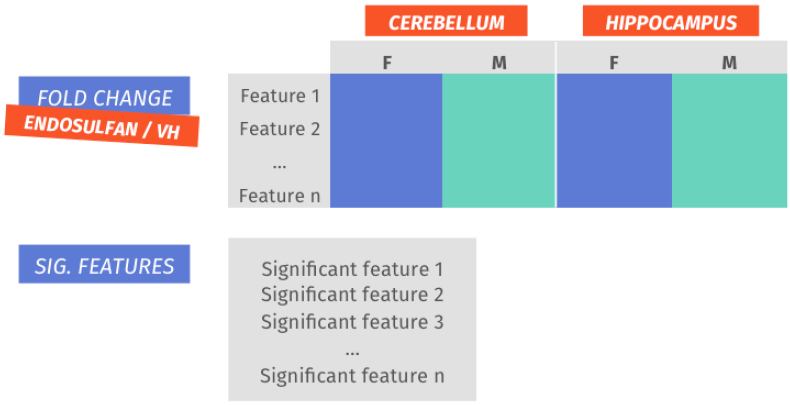


Figure 8. Paintomics data input for this biological case.

The fold-change values for each condition were obtained manually by calculating the averages for each sex, tissue and treatment, while the significant features were obtained through a differential expression analysis, as described in 3.6. The assessment of the miRNA targets for this biological case study in order to correctly input miRNA-seq data is described in further detail in 3.7.

For this biological case, Paintomics is run twice, once per Tissue considered (Cerebellum and Hippocampus). However, all fold-change (FC) data is inserted, the runs varying in the significant features used as an input for each tissue. Since the different tissues could have different features not present in the other, once joining the data missing values were replaced by 0. Also, all previously filtered features are added back into the data with FC = 0 so as to improve mapping efficiency, although this holds no biological meaning for metabolomics or proteomics.

A toy example of a pathway returned by Paintomics for this biological case is shown in Figure 9.

3.6. Differential expression analysis

Differential expression analyses are probably the most performed tests in the omics disciplines, allowing for the discovery of features whose levels (expression, concentration, etc.) change between different conditions, an integral part of understanding the molecular basis of phenotypic variation (Soneson & Delorenzi, 2013). A differential expression analysis was carried out in this work to obtain the significant features for future input into Paintomics.

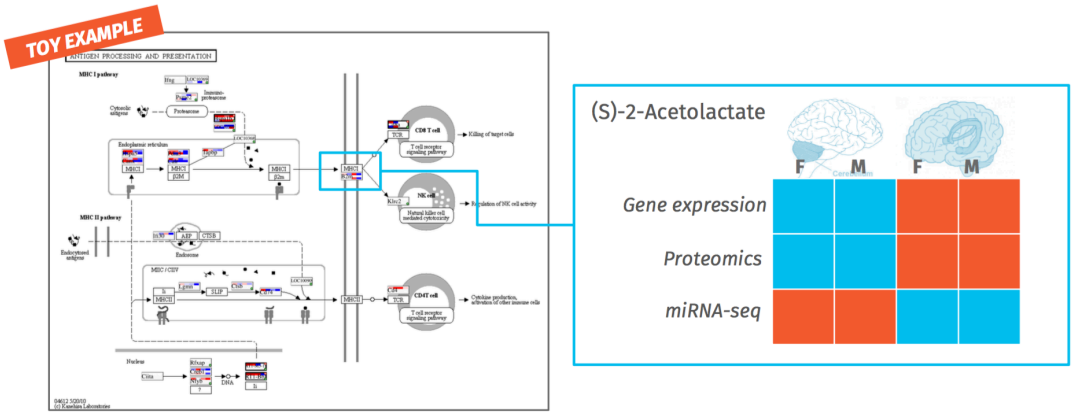


Figure 9. Paintomics toy example. On the left a KEGG pathway is represented, to which the provided features (metabolites, proteins, miRNAs and genes) have been mapped. On the right what Paintomics would show for each mapped feature in this biological case study is represented, two columns representing values for the Hippocampus tissue and two columns representing values for the Cerebellum tissue (as represented by the images, which do not appear in the live version of Paintomics and have been added for clarification). Differences between sexes are also shown. Blue represents under-expression and red over-expression.

The tool of choice to perform this analysis was the *Limma* R package (Ritchie *et al.*, 2015), which provides an integrated solution for analyzing data from gene expression experiments. It contains rich features for handling complex experimental designs and for information borrowing to overcome the problem of small sample sizes. The Limma pipeline includes linear modeling to analyze complex experiments with multiple treatment factors, quantitative weights to account for variations in precision between different observations, and empirical Bayes statistical methods to borrow strength between genes. Although it was initially designed for microarray gene expression data, it can be used for other omics as long as the data follows approximately a Gaussian distribution, which is one of the requirements of the linear modeling underlying Limma, or has undergone a proper transformation to meet this requirement. For instance, this is the case of RNA-seq, whose expression quantification consists of integer counts, unlike microarrays, which yield intensities that are essentially continuous numerical measurements. Because of this, statisticians were interested in applying normal-based microarray like-statistical methods to RNA-seq read counts. However, an obstacle to applying normal-based statistical methods to read counts is that the counts have markedly unequal variabilities, even after log-transformation. In light of this, the voom transformation was created (Law *et al.*, 2014).

Therefore, in order to apply the *Limma* pipeline to our data, the first step consisted on the application of the voom transformation to the transcriptomics and metabolomics data. The proteomics data, which came as a log2 ratio, after first inspection looked to be normally distributed, and as such it was considered that the transformation was not required.

To find the relevant features in this study we were interested in metabolites, proteins and genes that were differentially expressed due to the effects of either the pesticide or the interaction of the pesticide with the sex of the animal model. There was no interest in the differences due to sex, as the objective of this effort was to assess the effects of different pesticides on our model organism.

Visually, an example the expected expression profiles for those considered significant for these variables is shown in Figure 10.

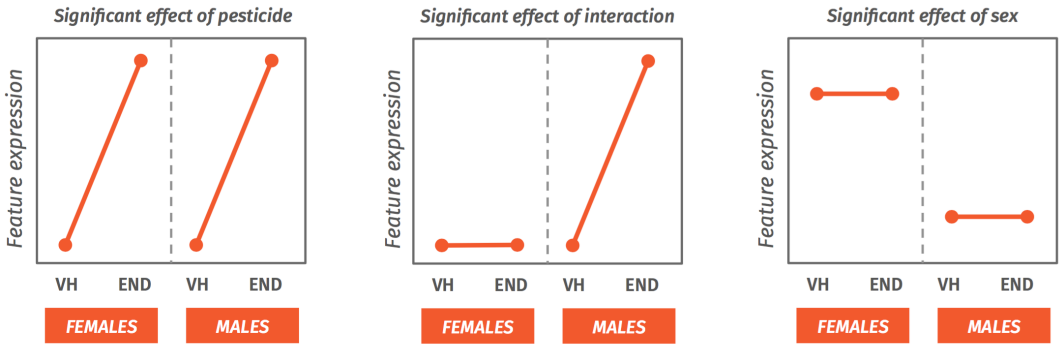


Figure 10. Expected expression profiles for significant pesticide effect (left), significant pesticide·sex effect (center) and significant sex effect (right).

The linear regression model considered for each feature was:

$$y = \beta_0 + \boldsymbol{\beta_1 \cdot x_{pesti}} + \beta_2 \cdot x_{sex} + \boldsymbol{\beta_3 \cdot x_{pesti \cdot sex}}$$

Where *y* represents the feature expression quantification. Again, only those features whose p-value for the effect of the pesticide and/or the effect of the pesticide in combination with the sex was lower than 0.05 were considered to be significantly affected by the pesticide and thus provided relevant features for Paintomics.

3.7. miRNA target prediction

As mentioned previously, Paintomics requires as an input miRNA targets for the miRNA data to be imputed back to a gene. Furthermore, as of right now this represents an extra step on behalf of the user, who is required to change their microRNA IDs by the targets of each corresponding microRNA. This extra step will be commented further in 3.7.1.

To first associate each miRNA in our data to their target genes, various miRNA target prediction solutions were considered, looking to find targets for all known *Rattus norvegicus* miRNAs with the highest possible validity, so as to avoid noisy and false associations, one of the problems that many bioinformaticians are facing nowadays when coming across miRNA analyses. Another challenge to overcome was the heterogeneous nature of miRNA ID conventions. Furthermore, one major issue in miRNA studies is the lack of bioinformatics programs to accurately predict miRNA targets. Animal miRNAs have limited sequence complementarity to their gene targets, which makes it challenging to select relevant biological features to build target prediction models with high specificity.

miRDB (Mirdb.org, 2016) is an online database for predicted microRNA targets in animals. It is based on a new miRNA target prediction program based on support vector machines (SVMs) and high-throughput training datasets. By systematically analyzing public high-throughput

experimental data, the database claims to have identified novel features that are important to target downregulation. These new features as well as other known features have been integrated in an SVM machine-learning framework for the training of the target prediction model. Also, the prediction algorithm has been validated by independent experimental data for its improved selectivity on predicting a large number of miRNA down-regulated gene targets.

After evaluating the options we had, miRDB was found to be a convenient source for target predictions from which predictions for all known *Rattus norvegicus* miRNAs were downloaded, each with an associated prediction score from 50 to 100 (the higher the better). Right off the bat, a cut-off score of 80 was picked as lower values were considered to be too permissive and a smaller number of predictions were favored. The distribution of genes associated to a single miRNA and miRNAs associated to a single gene can be found in Figure 11. A density plot with the effect of the cut-off can be found in Attachment XXI.

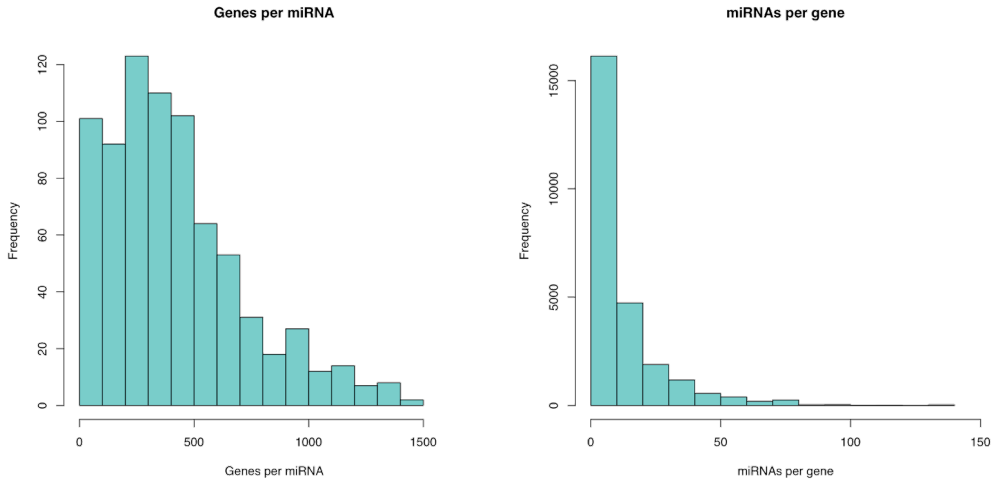


Figure 11. Histograms of genes per miRNA and miRNAs per gene for miRDB *Rattus norvegicus* targets, using cut-off = 80.

A reduction of the total number of target predictions found was of interest by means of picking a high score cut-off. A validation of the scores and the database was also of importance. To do so several approaches were employed:

1. A PubMed search was done, looking for experimentally validated brain miRNAs and their target genes. These were later searched for in the miRDB database, checking their score. A comparison with miRTarBase, a database containing experimentally validated microRNA targets, was also performed, again checking the score in miRDB for all the entries.
2. Finally, in order to make sure that the score picked was not too restrictive so as to reduce the data significantly, an analysis on how each score cut-off affected the number of miRNAs in each tissue for the DENAMIC miRNA-seq data was performed.

From the Pubmed search recent articles associated with brain miRNAs were found where miRNAs were associated with an experimentally validated target (Hanin *et al.,* 2014; Huang *et al.,* 2015; Ma *et al.,* 2015; Long *et al.,* 2014; Lopez *et al.,* 2015; Shen *et al.,* 2015; Yang *et al.,* 2015; Xing *et al.,* 2015; Zou *et al.,* 2014). The scores pertaining to these associations were then found in the miRDB predictions so as to check their validity. The basic statistics for the articles' scores is found in Figure 12. To gain

a more thorough comparison, the miRTarBase database was downloaded and compared with the miRDB database scores, the resulting statistics again represented in Figure 12. From these comparisons, the validity of the miRDB database was proven, as most experimentally validated interactions were found to have a score over 80.

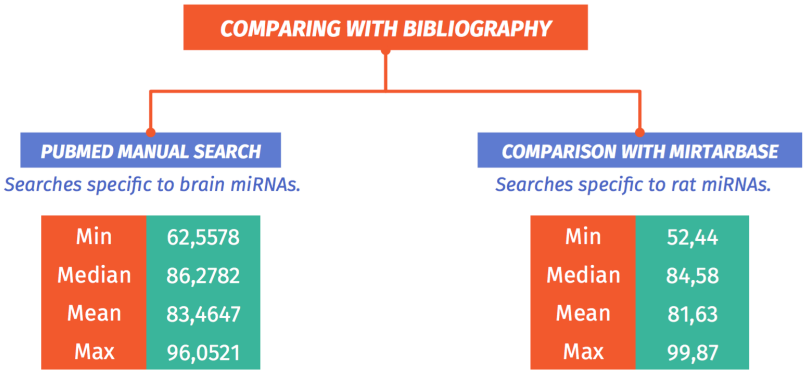


Figure 12. Basic statistics for bibliography comparison with miRDB database predictions.

However, the number of predictions was still fairly large, and the highest possible cut-off score was desirable so as to filter out the highest number of false positive miRNA targets without sacrificing a large amount of data from the miRNA-seq procedure. To choose said cut-off score a simple analysis comparing the effects of the different cut-off scores on our data was performed, the results depicted in Table 7. This analysis was performed over both filtered and non-filtered data, since all miRNAs, filtered or non-filtered, were due to be used as an input for Paintomics and thus required an ID change. After evaluating the results, cut-off score of 85 was chosen and used to filter lower-scored target predictions.

Table 7. Effects of different cut-off scores on filtered and non-filtered data.

	Median (genes per miRNA)	Median (miRNAs per gene)	Num. miRNAs	Num. genes	Intersection miRNAs CB	Intersection miRNAs HP	Intersection genes CB	Intersection genes HP
80	90	3	747	15733	409	419	2124	2128
85	63.5	3	732	13036	402	411	1899	1900
90	37.5	2	714	9966	393	401	1646	1651
95	17	2	686	6441	378	386	1175	1176

Over filtered data

Over non-filtered data

Intersection miRNAs CB	Intersection miRNAs HP	Intersection genes CB	Intersection genes HP
616	616	3155	3155
605	605	2794	2794
592	592	2381	2381
569	569	1676	1676

3.7.1. Preparing miRNA data for Paintomics

Once the final list of microRNA-target interactions had been defined, each miRNA ID was converted to its target's ENSEMBL gene ID in order to obtain the necessary input for Paintomics.

Due to differences in naming conventions (miRNAs were represented in different ways in our data and in the target predictions), several resources were used as stepping stones when converting IDs, the first being the miRBase database (which possessed the equivalencies between different miRNA naming conventions) and the second being Biomart (which allowed changing targets (represented as RefSeq transcripts) to ENSEMBL genes. These were used to transform the predictions file, which in the end contained miRNAs as miRBase Accession Codes and targets as ENSEMBL Genes.

Finally, using this pre-processed predictions file, our miRNA-seq ID's were converted to their targets' IDs. Due to the fact that each miRNA had several targets predicted, and that these targets could be shared between different miRNAs, to obtain an only entry for each target gene the sum of the fold changes of all the miRNAs sharing the same target was calculated. The significant features for the miRNA-seq data correspond to the genes that were targeted by at least one differentially expressed miRNA. The overall process followed is summarized in Figure 13.

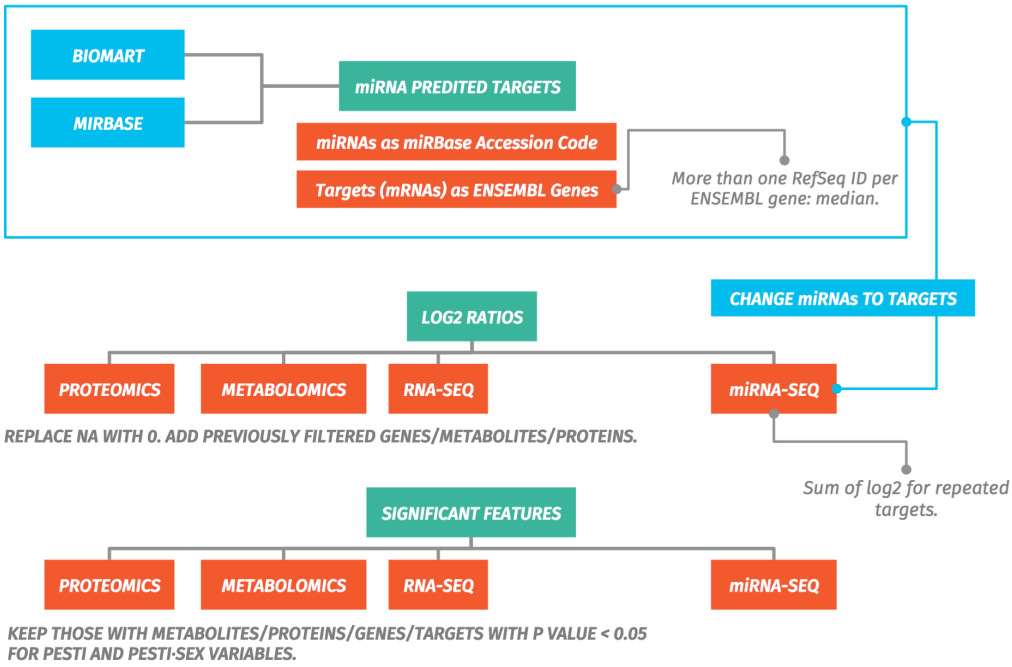


Figure 13. Diagram of ID changes for miRNAs and obtaining inputs for Paintomics.

4. Results and Discussion

4.1. Exploratory analysis and pre-processing

Proteomics and metabolomics

Once the missing values were imputed, an exploratory analysis of the metabolomics and proteomics data was performed to check for preliminary patterns and anomalies that might require addressing. First of all, a PCA for each omic was performed, for which proteomics data was centered, and metabolomics data was log-transformed, centered and scaled. PCA results are represented in Figure 14.

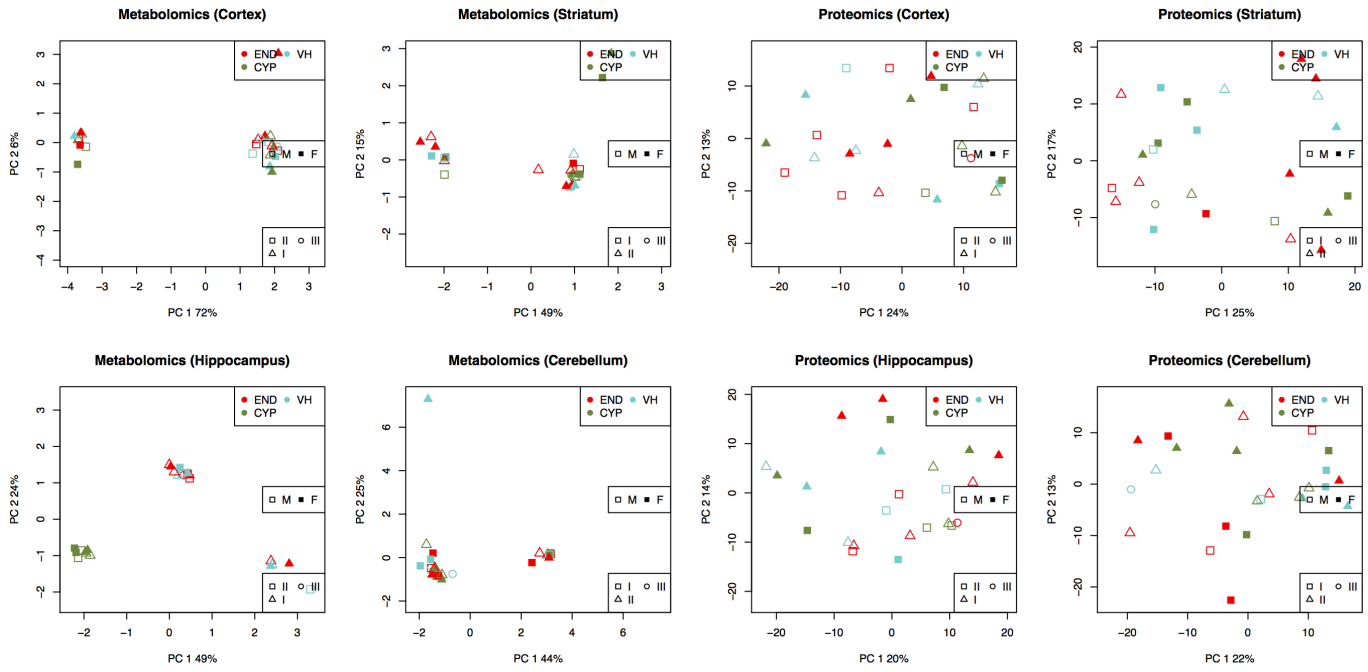


Figure 14. PCA plots for Principal Component 1 and 2 for each tissue in proteomics and metabolomics data. Color represents the treatment performed on the rat from which the sample was obtained (pesticide). Whether the shape is filled in or not corresponds to the sex. Finally, the shape corresponds to the mother of the rat from which the sample was obtained.

In the metabolomics PCA, the first two principal components explained from 64% to 78% of the total variability in the data, depending on the tissue. Unfortunately this variability seemed to be due to unknown sources rather than to the effect of the pesticide or the sex of the rats. Interestingly, the boxplots (Figure 15) and the heatmaps (Attachment XIV) showed a systematically

lower expression of metabolites for CYP pesticide in the cortex and cerebellum that, according to the experts in the DENAMIC project, had no biological relevance. To make sure that this was not an artifact of the missing value imputation, the analysis was again performed on the data after removing all metabolites with missing values in any of the samples, and the same results were obtained.

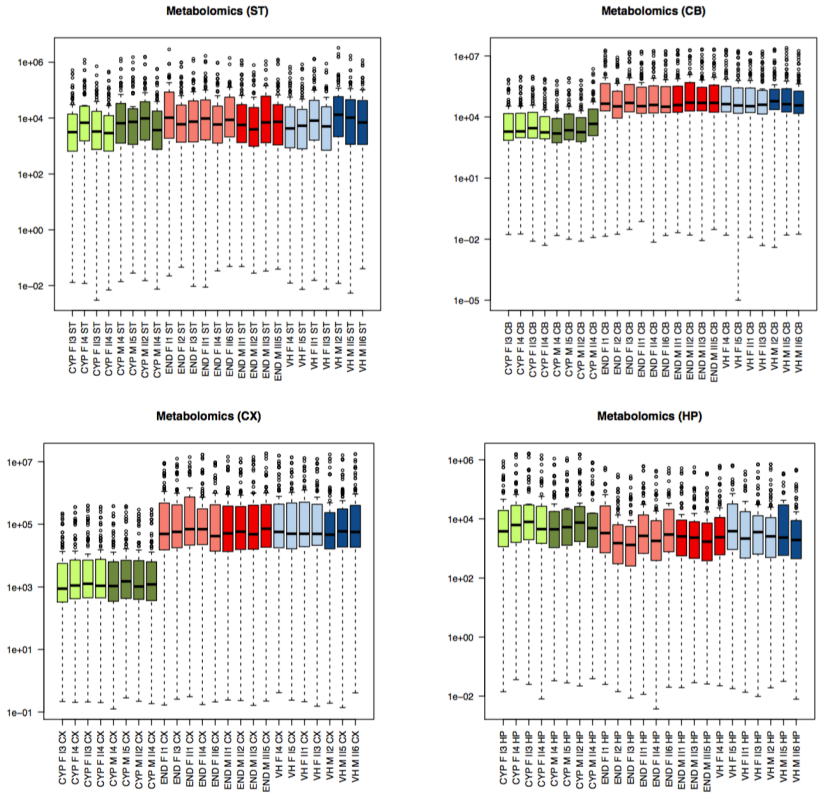


Figure 15. Boxplot representation of metabolomic data. Different colors correspond to different treatments for ease of visualization (reds to END, greens to CYP and blues to VH). The lighter shades correspond to females and the darker shades to males.

To correct this, a quantile normalization was first applied per tissue to make all the samples vary in the same range and make them comparable. However, the normalization was not enough to remove the noise in the data so the ARSyN correction method was used as a last attempt to reduce as much as possible the unwanted sources of variability. The PCA for the final data (Figure 16) still showed a high variability of the replicates pertaining to the same experimental condition but, in general, an improvement of the separation among experimental conditions was seen, although still far from perfect, probably because there are no sets of metabolites acting in coordination but a few of them presenting changes amongst conditions. This makes sense since the effects of the low-concentration pesticides are expected to affect only a small number of omic features in the organism.

Regarding the proteomics data we did not find the strong clustering due to unknown effects that we'd observed in the metabolomics data. There was no need for quantile normalization but ARSyNseq was also applied to try to reduce the noise in the data.

Finally, for both metabolomics and proteomics, the PCAs returned showed that the effect of the mother in the clustering of the rats was not important, so it was not necessary to take it into account for further analyses.

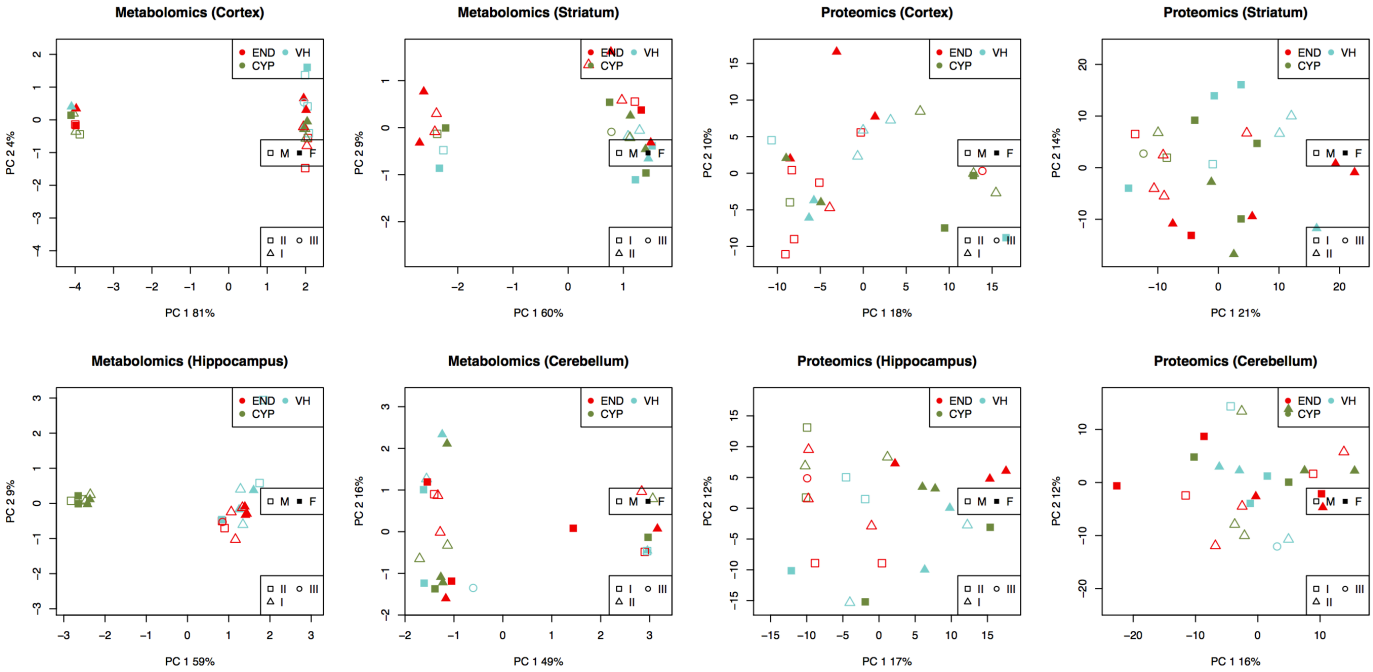


Figure 16. PCA plots for Principal Component 1 and 2 for each tissue in proteomics and metabolomics data after pre-processing. See Figure 14 caption for more details.

All boxplot representations throughout these transformations can be found in Attachment XIII, as well as heatmaps in Attachment XIV and PCAs in Attachment XV.

Transcriptomics

Once low-count genes were filtered out of the data, patterns were also looked for through a PCA analysis, although no clear groupings were found.

As the boxplots showed a very different data distribution amongst samples, especially in miRNA-seq, the TMM normalization was performed. However, the differences were so pronounced that a stronger correction such as the quantile normalization needed to be applied. Again, the ARSyN method was also used. The PCA results for the final data (Figure 17) showed a clear reduction in noise as well as the visibility of certain patterns. For example, for RNA-seq in the Hippocampus, PC1 is able to cluster treatments together, as well as sexes. The separation between sexes can also be visualized for RNA-seq in the cerebellum through PC1, although not as clearly.

All PCAs and boxplots obtained throughout the pre-processing stages for the transcriptomics data can be found in Attachment XVI and XVII respectively.

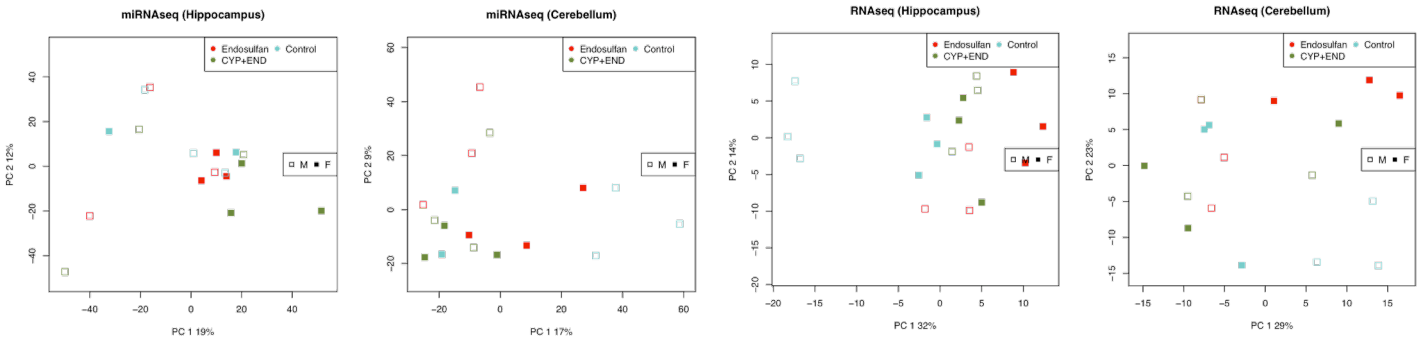


Figure 17. PCA scores 1 & 2 for miRNA and RNA-seq data after normalization and arsynseq. Colors represent treatments and filled-in/empty shapes represent the sexes.

4.2. Differential expression analysis

As mentioned previously, to gain a further understanding into the changes that pesticide treatment brought upon the different omics, as well as to obtain the significant features for later use in the Paintomics analysis, a Limma analysis was performed on all omics data sets. The results are represented as Venn diagrams for the Cerebellum and Hippocampus tissues, as well as for the effects of the Pesticide and the interaction between Pesticide and Sex, in Figure 18, when considering p-value = 0.05 as a cut-off for significance.

The fact that few features are returned as differentially expressed is coherent with the intra-condition variability observed in the previous exploratory analyses. The expression profiles for each significant feature can be visualized in Appendix XVIII. Heatmaps corresponding to the significant features found for transcriptomics can be visualized in Appendix XX.

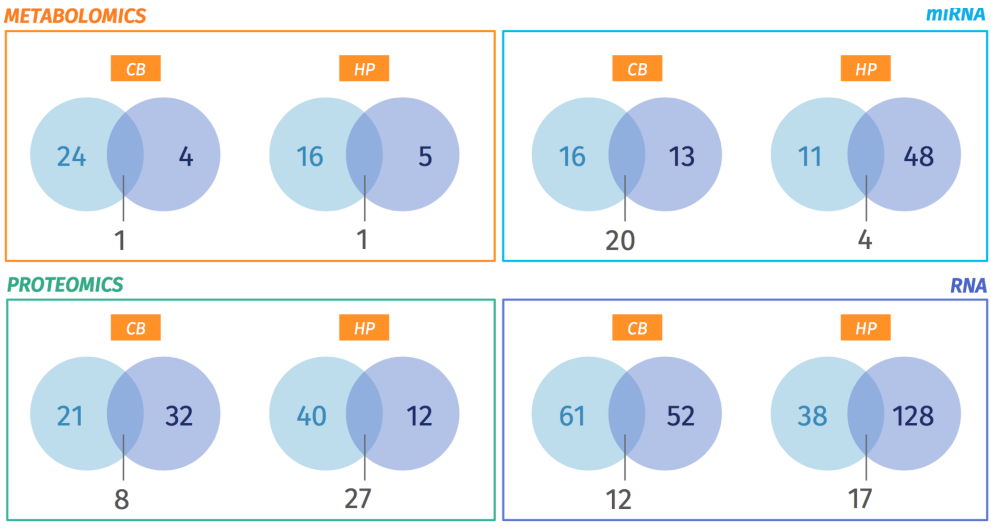
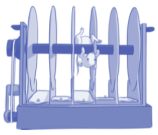
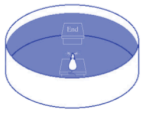


Figure 18. Venn diagrams showing number of DE features per omic and tissue. Light blue corresponds to features with a p-value lower than 0.05 for the effect of the pesticide, and dark blue features significant for the effect of the interaction between sex and pesticide. Thus, their intersection corresponds to features that were called as significant for both the pesticide and the interaction.

4.3. Learning tests

Molecular changes have to be translated into phenotypic responses for them to be relevant, so a brief analysis of the learning tests to later correlate between physiological and molecular responses was of interest. Although the Neurobiology lab had already performed statistical tests using this same data along with that of other sets, we decided to contrast them through a simple series of Wilcoxon tests for the clinical data pertaining to Set 01 in order to deduce whether or not the END treatment had any considerable effect when compared to the control. Results are found in Table 8, being the most significant those highlighted in a darker color.

Table 8. Results of Wilcoxon tests applied on learning test scores for Set 01 to compare rats treated with Endosulfan (END) or vehicle (VH) in general and for each sex.

		MOTOR		COGNITIVE		
		 ROTAROD	 BEAM WALK	 RADIAL MAZE		 MWM
END vs VH	P-value	0.4698	0.2398	0.1382	1	0.2571
	Dif. median	-85	0.5	3	-1.5	38.5
♂ END vs VH	P-value	0.7	0.1642	0.2	1	0.2
	Dif. median	13	0.5	4	-2	99.5
♀ END vs VH	P-value	0.1143	1	0.6973	0.69373	0.8
	Dif. median	-104	0	0.5	-9	-15
				working	total	
					WILCOXON TEST	

As can be observed, no significant effect of the pesticide (p-value < 0.05) was found. However, one must consider that a non-parametric test was performed, which although more robust and simple, requires larger sample sizes than a parametric test (such as a t-test) to draw conclusions with the same degree of confidence, which were not available in this case, where many times just 2 or 3 rats were available for each condition. Also, previous t-tests carried out by the Neurobiology lab further contribute to the idea that these highlighted effects were significant.

An interesting and previously observed effect of endosulfan treatment was the decrease of motor coordination in female rats as measured by the Rotarod test. Motor coordination is modulated by extracellular glutamate and gamma-aminobutyric acid (GABA) in the cerebellum (Chiu *et al.*, 2005; Hanchar et al., 2005) and an increase in GABA in the extracellular space has been associated with a decrease in motor coordination in previous studies (Boix *et al.*, 2010).

Another interesting effect found, also previously observed by the Neurobiology lab, was a difference in learning ability and memory in spatial tasks between males and females when treated with Endosulfan, as shown by the Radial Maze and MWM test scores, where the males were found to have a significant change but the females were not. Long-term potentiation in the hippocampus

is considered the basis for spatial learning and memory, which has been found to differ between male and female rats. The mechanism underlying this remains unknown, however, it has been postulated it could be due to differences in the cGMP-PKG signaling pathway for the different sexes (Monfort *et al.*, 2015).

Owing to the previously published bibliography and to the thorough knowledge of the Neurobiology team regarding these behaviors, when coming across the analysis of the integration results, the GABA synapse pathway in the cerebellum as well as the cGMP-PKG signaling pathway in the hippocampus was prioritized (section 4.5).

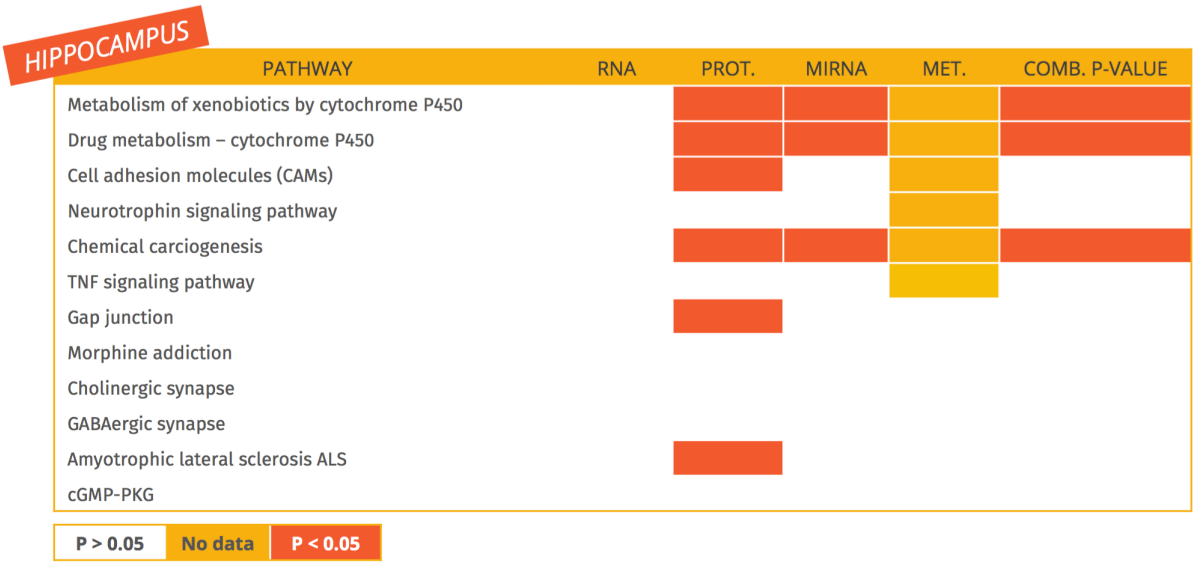
4.4. Integration

Once the data had been formatted properly, Paintomics was run for the hippocampus and cerebellum data. As a result, 294 pathways were found to which the features were mapped, and 29 and 9 were found significant for the combination of the omics in the case of the cerebellum and hippocampus respectively.

Many pathways were returned, however, the objective of this work was to understand the molecular basis of impaired neurodevelopment, and so many of the pathways were not of interest, because they were not exclusive to neurons or associated with said event. In order to ease the analysis task, the pathways were reduced to those of interest, represented in Tables 9 and 10.

Table 9. Returned Paintomics pathways of interest for the Cerebellum.

Table 10. Returned Paintomics pathways of interest for the Hippocampus.



4.5. Discussion

In this section only several pathways will be discussed briefly, assessing previously published literature, learning test data, Paintomics results and correlations between features and cognitive and motor abilities.

Difference between tissues and sexes

Paintomics returned 29 significant pathways for the cerebellum, while only 9 for the hippocampus. This difference between the tissues was to be expected, as the cerebellum receives a larger blood flow than the hippocampus, thus making it more susceptible to changes in the organism.

Most features in this study presented clear differences in behavior between males and females, as shown by opposite fold changes as well as by the different learning tests. Sex-based differences in the toxic effects of endosulfan have already been observed in *Rattus norvegicus* (Paul *et al.*, 1995). Endosulfan has been shown to have estrogenic effects in primary neural cultures (Briz *et al.*, 2011) and estrogens in turn have been cited as potential neuroprotectors in brain areas that are not primarily involved in reproduction (such as the hippocampus or cerebellum) (Brann *et al.*, 2007). This could potentially be the reason behind the differences observed between the sexes (Lafuente & Pereiro, 2013).

The GABA pathway

As mentioned in 4.3, it has been predicted that the observed reduction in female rat motor skill could be due to an increase in GABA in the cerebellum. GABA is an inhibitory neurotransmitter, so an increase in its concentration could act by potentially reducing neuronal excitability, thus leading to impaired coordination.

Involvement of endosulfan in the central nervous system (CNS) has been the subject of various studies. Those who have suffered occupational exposure to endosulfan have been reported to exhibit convulsions, epilepsy, hyperactivity, irritability, tremor, and paralysis, as well as neurobehavioral symptoms manifested by agitation, memory defects, partial aphasia, limited cognition and impairment of motor coordination (Chan *et al.,* 2006). The mechanism of neurotoxicity of endosulfan has been documented to be dominated by its capacity to inhibit non-competitively the GABA-A type of receptors, which function as chloride channels with the capacity to hyperpolarize and inhibit neurons in the central nervous system (Cole and Casida, 1986; Chan et al., 2006). When GABA binds to its receptor, the chloride ion channels are opened, leading to an influx of chloride into neurons through an electrochemical gradient. The result is hyperpolarization of the cell membrane and inhibited neuron firing. When endosulfan binds this event is inhibited, there's a blockage of the influx of chloride ions into the nerve cells, causing uncontrollable excitation, as GABA's binding to its receptor is unable to repress the neural response (Jang *et al.,* 2016).

As mentioned in the introduction, the developing nervous system is proposed to be a potentially sensitive target for pesticide exposure. The occurrence of the aforementioned stress on the fetus as well as on the first few weeks of the offspring's life, as modeled by the DENAMIC project, could lead to the observed increase in GABA and motor incoordination in females' later life stages.

To understand the molecular basis of this potential sequela on the brain post developmental exposure, we took a closer look at the GABA pathway returned by Paintomics (Figure 19), and we observed that 4 features had been marked as significant. We explored these four features in more detail (Figure 20), and observed that 2 proteins and 2 genes were significantly over or under expressed. To know whether these proteins or genes could be associated with decreased motor coordination in females owing to changes in the Cerebellum, the left-most column was of interest (according to the biological case study input used in this work), seeing that for some over-expression had been observed but for others under-expression was found. Wanting to know if these changes were correlated in any way with the changes in clinical data, Pearson correlation coefficients were found between the clinical variables and the different features, their value also indicated in Figure 20. All correlation plots between the DE features for all omics and learning tests can be found in Appendix XIX.

According to these preliminary findings, we found that the over expression of the GABA Vesicular Transporter (VGAT) is correlated with a decrease in the time a mouse is able to maintain itself on the Rotarod, thus its association with impaired motor coordination. This transporter is in charge of pumping GABA into the synaptic vesicle, which later fuses with the axon membrane to release GABA into the synaptic cleft (Saito *et al.,* 2010).

Interestingly, a study that employed mice as an in vivo model carried out by Wilson *et al.* (2014) found a significant reduction of VGAT in cortex GABAergic neurons of male offspring whose mothers had been exposed to endosulfan prior to their pregnancy. Also, previous studies observed significant alterations in GABA in the prefrontal cortex of male offspring whose mothers had been exposed to endosulfan during gestation and lactation (Cabaleiro *et al*., 2008). This supports the idea that alterations in VGAT levels could have profound effects on GABA levels.

Therefore, one could think that a reduction in GABA signaling in the cerebellum during early stages of development due to developmental endosulfan exposure could lead to changes that ensure an increase in GABA in later stages. Furthermore, a hypothesis to how the over-expression of VGAT could lead to an increase of extracellular GABA could be that an increase in the levels of this transporter could consequently lead to an increase in GABA in the synaptic vesicle, thus its increase when being released into the extracellular space. Of course, this is just a preliminary postulation and further studies are required.

Also worth noting is that a decrease in the solute carrier family 38, member 5 (SLC38A5/SNAT5 peptide) gene expression was associated with a decrease in motor coordination in females. SNAT5 is a member of the System N family, expressed in glial cells in the adult brain, able to transport glutamine, histidine or glycine among other substrates (Cubelos *et al.*, 2005, Rodriguez *et al.*, 2014). How the under-expression of this gene could lead to an increase in GABA in the extracellular matrix is unknown. It could also potentially act through another underlying mechanism contributing to a decrease in motor coordination. Also, sadly, no proteomic data was available in order confirm whether this change in gene expression also translated into an important change in transporter concentration.

As a conclusion it could be said that these features could potentially act as biomarkers or treatment targets for pesticide induced impaired neurodevelopment. However, further studies are required to validate these predictions.

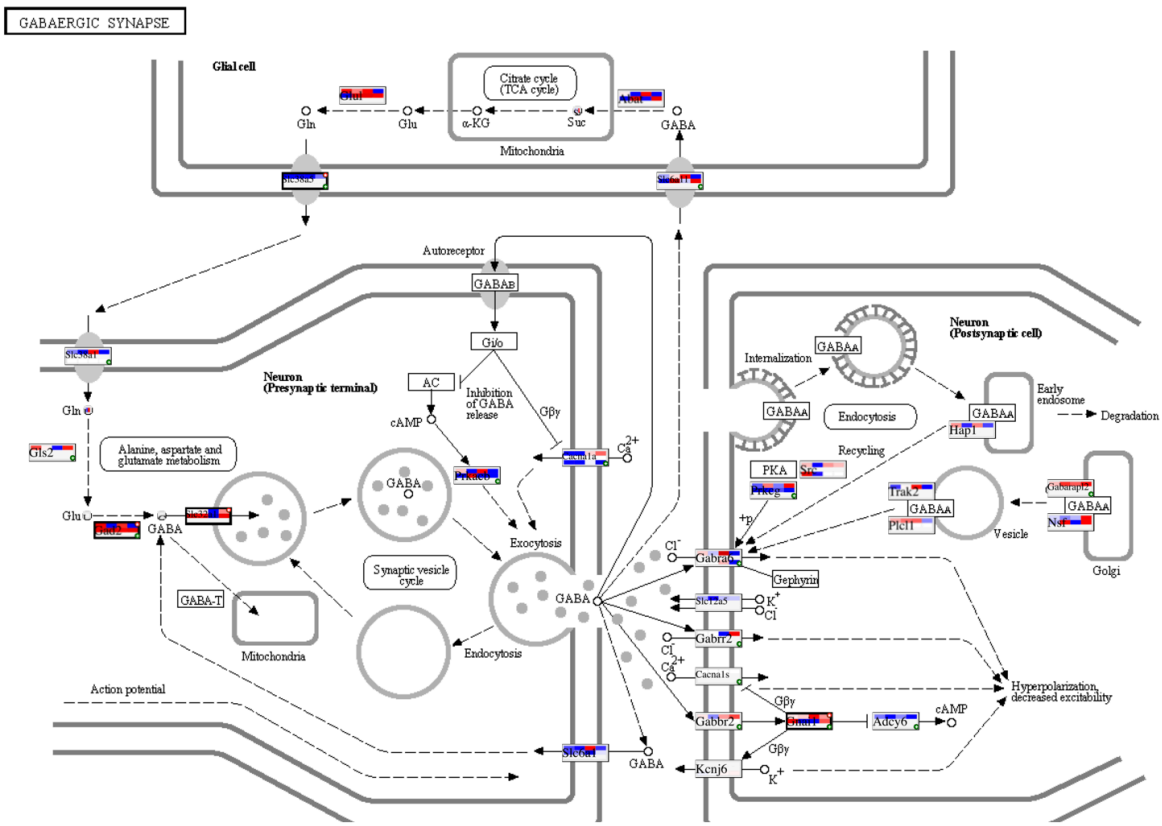


Figure 19. GABAergic synapse KEGG Pathway returned by Paintomics for the Cerebellum.

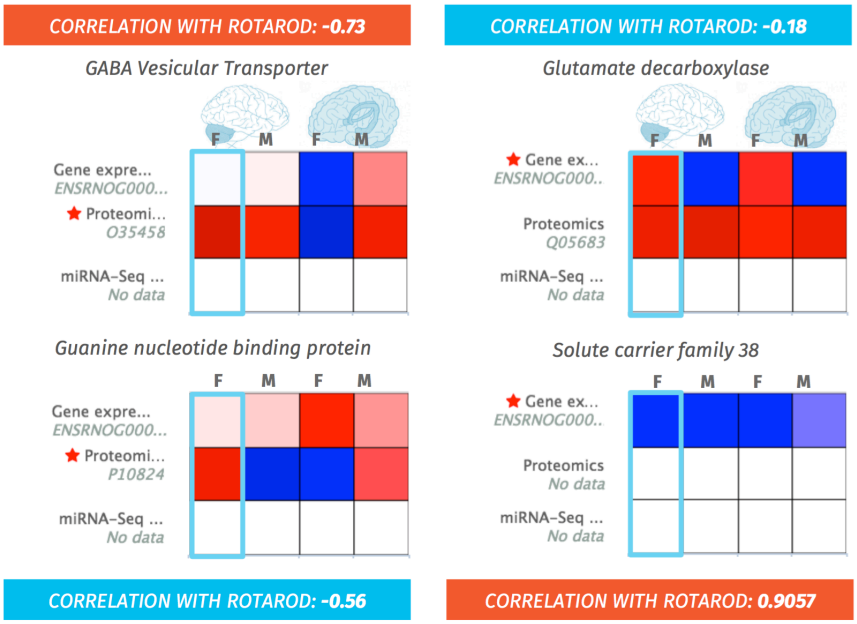


Figure 20. Expression profile of significant features in GABAergic Synapse pathway returned by Paintomics. Also shown is the Pearson correlation coefficient in regards to Rotarod test.

cGMP-PKG signaling pathway

As mentioned previously in section 4.3, a significant difference in learning and memory abilities was found between male rats treated with endosulfan and controls, while no significant effect was found in females. In tests of spatial ability, males outperform females both in rats and in humans, with previous studies concluding that this was due to reduced activation of soluble guanylate cyclase and the formation of cGMP, as well as the mechanisms that followed, in the Hippocampus (Monfort *et al.*, 2015).

Although the cGMP-PKG signaling pathway (Attachment XXII) was not found to be significant by Paintomics for any of the omics, the differences between males and females mapped visually onto this pathway were of interest to further validate previously made hypotheses or to understand the underlying molecular mechanism that differentiates behavior between the sexes.

When taking a closer look, we found that no significant features, which were made-up by those relevant for the effects of the pesticide or the interaction between the pesticide and the sex, were mapped in the pathway. Therefore no explanation for the difference between sexes in regards to their behavior towards endosulfan as shown by the learning tests could be found in this pathway. However, future studies are required in order to understand the role of the cGMP-PKG pathway in these events.

Parkinson's disease

Parkinson's disease (PD) is a progressive neurodegenerative disorder characterized by loss of dopaminergic neurons in the substantia nigra pars compacta (SNpc) of the midbrain. Parkinsonian

patients also exhibit symptoms and signs suggestive of hypothalamic dysfunction (such as dysautonomia, impaired heat tolerance). Lewy body formation has been demonstrated in every nucleus of the hypothalamus, specifically the tuberomamillary and posterior hypothalamic (Sandyk *et al.*, 1987). Furthermore, increasing evidence suggests that the cerebellum may have certain roles in the pathophysiology of Parkinson's disease. Anatomical studies identified reciprocal connections between the basal ganglia and cerebellum and Parkinson's disease-related pathological changes in this brain area (Wu & Hallet, 2013).

As mentioned in the introduction, developmental exposure to pesticides such as endosulfan has been suggested to contribute to neurotoxicity leading to neurodegenerative diseases such as Parkinson's disease. Our integrative efforts returned a total of 3 significant features mapped to the Parkinson's disease KEGG pathway for the cerebellum (Attachment XXIII), and 2 for the hippocampus.

In the cerebellum we found that the iron-sulfur protein subunit of succinate dehydrogenase (involved in the complex II of the mitochondrial electron transport chain and that is responsible for transferring electrons from succinate to ubiquinone) and the NADH dehydrogenase iron-sulfur protein 2 (core subunit of the mitochondrial membrane respiratory chain NADH dehydrogenase (Complex I) that is believed to belong to the minimal assembly required for catalysis) proteins were found to be DE, as well as adenosine.

In the hippocampus, protein deglycase DJ-1 (protein deglycase that repairs methylglyoxal- and glyoxal-glycated amino acids and proteins, and releases repaired proteins and lactate or glycolate, respectively) as well as the Cox6b2 gene (that codes for cytochrome c oxidase subunit VIb polypeptide 2, a subunit of the cytochrome c oxidase complex, also known as Complex IV, the last enzyme in the mitochondrial electron transport chain) were found to be DE.

Curiously, all genes and proteins found for both tissues are associated to mitochondrial function. Many lines of evidence suggest that mitochondrial dysfunction plays a central role in the pathogenesis of PD, starting in the early 1980s with the observation that an inhibitor (MPTP) of complex I of the electron transport chain can induce parkinsonism. Furthermore, recent findings have established that mitochondrial dysfunction is a common denominator of sporadic and familial PD, moving mitochondria to the forefront of PD research (Winklhofer & Haass, 2010).

As is the case with most features in this study, the effects of endosulfan on these proteins are not homogeneous, meaning that its effect varies depending on the sex and the tissue. This complicates the biological analysis of the molecular basis of this disease as well as of the neurotoxicity of this pesticide, reaching outside the boundaries of this effort. However, these proteins could potentially be targets for further studies in the future.

5. Conclusions

To summarize and conclude this effort, several observations regarding the whole of the project are compiled in this section.

- Great challenges were found when manipulating data not only from different omics disciplines, and thus different in nature and significance, but from different entities as well, as each team has different work disciplines and distance and time can get in the way of an efficient communication, thus complicating a study beyond its scientific efforts.

- Pre-processing of the omics data, while many times over-looked as following a protocol, was also proven essential, and can determine whether relevant conclusions are found or not. Pipelines have to be adjusted to each scenario, and many times different methods will have to be tried until finding the correct match in order to ensure the maximum representation of the biological case study as represented by the data to be analyzed. This work has served as an example of the difficulty that comes with the pre-processing stage preceding multi-omics integration, and that it's not always possible to obtain clean data due to the inherent noise associated to the omics disciplines.

- The data received was quite noisy, something that's been come to expect from most omics efforts. No significant patterns were found through the exploratory analysis performed on the individual omics. However, this could be explained by the fact that the effects of the low-concentration pesticides affect only a small set of features in the organism, not enough to completely digress from the norm when analyzed for example through a Principal Components Analysis.

- A differential expression analysis was performed on all the omics disciplines, obtaining a relatively small number of differentially expressed features, few with large fold changes. Expression profiles were created for these features so as to easily visualize changes between sexes, tissues and treatments. Frequent differences were found between males and females, as was to be expected and is hypothesized to be due to the estrogenic nature of the pesticide considered, as well as between the cerebellum and the hippocampus, due to differences in blood irrigation.

- miRNA target prediction and analysis continues to be a problem, with large numbers of predicted targets that create difficult interaction webs yet to be validated experimentally. This is especially true in regards to animal miRNAs.

- Differences in nomenclature standardization as well as in database content make contrasting information difficult, and part of the data is usually lost along the way.

- Multi-omics data integration has great potential. Each omic has its limits due to technological handicaps, but when integrated some of these can be overcome and a bigger

picture of the biological case study can be obtained, as was the case in this work, as exemplified by the Paintomics results, where in many cases only an omic was found to be relevant for a certain pathway, usually to lack of data for the rest of the disciplines. Furthermore, integrating through a visual representation, especially through biological pathways, has proven to be helpful and intuitive when analyzing and discussing omics efforts.

- Potential biomarkers and treatment targets for impaired motor coordination in female rats treated with endosulfan were found, the most promising being the Vesicular GABA Transporter. However, further experimental tests need to be carried out in order to test the veracity of said result, as well as further statistical analyses.

- Parkinson's disease was discussed in regards to our integration results, and several mitochondrial proteins were found to be of interest, further strengthening the role of this organelle in the pathogenesis of PD, as well as the involvement of the pesticide considered in neurodegenerative diseases.

- Bioinformatics goes hand in hand with biological advances and knowledge. This work was again an example of this, where without previous knowledge in neurobiology the analysis task would have proved impossible. Interdisciplinary efforts surround this field, and push it to its fullest potential.

Due to time constraints as well as other factors, this work marks just the beginning of what is likely to be a longer project in which many other scientists will get to collaborate in the future. There are still many things left to be done, and in the future the Genomics of Gene Expression team plans on analyzing our results in further detail, integrating more data that is yet to be received for a different set of rats, integrating each omics data with their corresponding clinical data to get a more profound look at the interaction between them using more complex models, and assessing the regulatory effect of miRNAs and transcription factors on gene expression by means of regression models.

6. Bibliography

ALTMAN, N. (1992). An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician*, 46(3), p.175.

ENSEMBL.ORG. (2016). [online] Available at: http://www.ensembl.org/biomart/ [Accessed 13 Jun. 2016].

BIOCONDUCTOR.ORG. (2016). *Bioconductor - Home*. [online] Available at: https://www.bioconductor.org/ [Accessed 13 Jun. 2016].

BIOINFO.CIPF.ES. (2016). *Paintomics v3.0*. [online] Available at: http://bioinfo.cipf.es/paintomics/ [Accessed 13 Jun. 2016].

BOIX, J., CAULI, O. AND FELIPO, V. (2010). Developmental exposure to polychlorinated biphenyls 52, 138 or 180 affects differentially learning or motor coordination in adult rats. mechanisms involved. *Neuroscience*, 167(4), pp.994-1003.

BRANN, D., DHANDAPANI, K., WAKADE, C., MAHESH, V. AND KHAN, M. (2007). Neurotrophic and neuroprotective actions of estrogen: Basic mechanisms and clinical implications. *Steroids*, 72(5), pp.381-405.

BRIZ, V., MOLINA-MOLINA, J., SANCHEZ-REDONDO, S., FERNANDEZ, M., GRIMALT, J., OLEA, N., RODRIGUEZ-FARRE, E. AND SUNOL, C. (2011). Differential Estrogenic Effects of the Persistent Organochlorine Pesticides Dieldrin, Endosulfan, and Lindane in Primary Neuronal Cultures. *Toxicological Sciences*, 120(2), pp.413-427.

BUESCHER, J. AND DRIGGERS, E. (2016). Integration of omics: more than the sum of its parts. *Cancer Metab*, 4(1).

CABALEIRO, T., CARIDE, A., ROMERO, A. AND LAFUENTE, A. (2008). Effects of in utero and lactational exposure to endosulfan in prefrontal cortex of male rats. *Toxicology Letters*, 176(1), pp.58-67.

CHAN, M., MORISAWA, S., NAKAYAMA, A., KAWAMOTO, Y. AND YONEDA, M. (2006). Development of anin vitro blood–brain barrier model to study the effects of endosulfan on the permeability of tight junctions and a comparative study of the cytotoxic effects of endosulfan on rat and human glial and neuronal cell cultures. *Environmental Toxicology*, 21(3), pp.223-235.

CHIU, C. (2005). GABA Transporter Deficiency Causes Tremor, Ataxia, Nervousness, and Increased GABA-Induced Tonic Conductance in Cerebellum. *Journal of Neuroscience*, 25(12), pp.3234-3245.

CHU, Y., & COREY, D. R. (2012). RNA Sequencing: Platform Selection, Experimental Design, and Data Interpretation. *Nucleic Acid Therapeutics*, *22*(4), 271–274. http://doi.org/10.1089/nat.2012.0367

COLE, L. AND CASIDA, J. (1986). Polychlorocycloalkane insecticide-induced convulsions in mice in relation to disruption of the GABA-regulated chloride ionophore. *Life Sciences*, 39(20), pp.1855-1862.

CUBELOS, B., GONZÁLEZ-GONZÁLEZ, I., GIMÉNEZ, C. AND ZAFRA, F. (2004). Amino acid transporter SNAT5 localizes to glial cells in the rat brain. *Glia*, 49(2), pp.230-244.

CRAN.R-PROJECT.ORG. (2016). *CRAN - Package corrplot*. [online] Available at: https://cran.r-project.org/web/packages/corrplot/index.html [Accessed 13 Jun. 2016].

DENAMIC-PROJECT.EU. (2016). *DENAMIC Developmental neurotoxicity assessment of mixtures in children*. [online] Available at: http://www.denamic-project.eu/ [Accessed 13 Jun. 2016].

FISHER, R. (1922). On the Interpretation of χ 2 from Contingency Tables, and the Calculation of P. *Journal of the Royal Statistical Society*, 85(1), p.87.

FISHER, R. (1938). *Statistical methods for research workers*. Edinburgh: Oliver and Boyd.

FLEISCHMANN, R., ADAMS, M., WHITE, O., CLAYTON, R., KIRKNESS, E., KERLAVAGE, A., BULT, C., TOMB, J., DOUGHERTY, B., MERRICK, J. AND AL., E. (1995). Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. *Science*, 269(5223), pp.496-512.

GARCIA-ALCALDE, F., GARCIA-LOPEZ, F., DOPAZO, J. AND CONESA, A. (2010). Paintomics: a web based tool for the joint visualization of transcriptomics and metabolomics data. *Bioinformatics*, 27(1), pp.137-139.

GIORDANO, G. AND COSTA, L. (2012). Developmental Neurotoxicity: Some Old and New Issues. *ISRN Toxicology*, 2012, pp.1-12.

GRIFFITHS, W. AND WANG, Y. (2009). Mass spectrometry: from proteomics to metabolomics and lipidomics. *Chemical Society Reviews*, 38(7), p.1882.

HANCHAR, H., DODSON, P., OLSEN, R., OTIS, T. AND WALLNER, M. (2005). Alcohol-induced motor impairment caused by increased extrasynaptic GABAA receptor activity. *Nature Neuroscience*, 8(3), pp.339-345.

HANIN, G., SHENHAR-TSARFATY, S., YAYON, N., YAU, Y., BENNETT, E., SKLAN, E., RAO, D., RANKINEN, T., BOUCHARD, C., GEIFMAN-SHOCHAT, S., SHIFMAN, S., GREENBERG, D. AND SOREQ, H. (2014). Competing targets of microRNA-608 affect anxiety and hypertension. *Human Molecular Genetics*, 23(24), pp.6694-6694.

HASTIE T, TIBSHIRANI R, NARASIMHAN B AND CHU G (2016). *impute: impute: Imputation for microarray data*. R package version 1.46.0.

HE, L. AND HANNON, G. (2004). MicroRNAs: small RNAs with a big role in gene regulation. *Nat Rev Genet*, 5(7), pp.522-531.

HUANG, L., LI, J., PANG, X., CHEN, C., XIANG, H., FENG, L., SU, S., LI, S., ZHANG, L. AND LIU, J. (2015). MicroRNA-29c Correlates with Neuroprotection Induced by FNS by Targeting Both Birc2 and Bak1 in Rat Brain after Stroke. *CNS Neuroscience & Therapeutics*, 21(6), pp.496-503.

IHAKA, R. AND GENTLEMAN, R. (1996). R: A Language for Data Analysis and Graphics. *Journal of Computational and Graphical Statistics*, 5(3), p.299.

JANG, T., JANG, J. AND LEE, K. (2016). Mechanism of acute endosulfan intoxication-induced neurotoxicity in Sprague-Dawley rats / Mehanizam akutne neurotoksičnosti u Sprague-Dawley štakora izazvane trovanjem endosulfanom. *Archives of Industrial Hygiene and Toxicology*, 67(1).

JONES, B. AND ROBERTS, D. (1968). The quantitative measurement of motor inco-ordination in naive mice using an accelerating rotarod. *Journal of Pharmacy and Pharmacology*, 20(4), pp.302-304.

KANEHISA, M. (2000). KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 28(1), pp.27-30.

LAFUENTE, A. AND PEREIRO, N. (2013). Neurotoxic effects induced by endosulfan exposure during pregnancy and lactation in female and male rat striatum. *Toxicology*, 311(1-2), pp.35-40.

LAW, C., CHEN, Y., SHI, W. AND SMYTH, G. (2014). voom: precision weights unlock linear model analysis tools for RNA-seq read counts. *Genome Biol*, 15(2), p.R29.

LLOP, S., JULVEZ, J., FERNANDEZ-SOMOANO, A., SANTA MARINA, L., VIZCAINO, E., IÑIGUEZ, C., LERTXUNDI, N., GASCÓN, M., REBAGLIATO, M. AND BALLESTER, F. (2013). Prenatal and postnatal insecticide use and infant neuropsychological development in a multicenter birth cohort study. *Environment International*, 59, pp.175-182.

LONG, J., RAY, B. AND LAHIRI, D. (2013). MicroRNA-339-5p Down-regulates Protein Expression of -Site Amyloid Precursor Protein-Cleaving Enzyme 1 (BACE1) in Human Primary Brain Cultures and Is Reduced in Brain Tissue Specimens of Alzheimer Disease Subjects. *Journal of Biological Chemistry*, 289(8), pp.5184-5198.

LOPEZ, J., LIM, R., CRUCEANU, C., CRAPPER, L., FASANO, C., LABONTE, B., MAUSSION, G., YANG, J., YERKO, V., VIGNEAULT, E., EL MESTIKAWY, S., MECHAWAR, N., PAVLIDIS, P. AND TURECKI, G. (2014). miR-1202 is a primate-specific and brain-enriched microRNA involved in major depression and antidepressant treatment. *Nature Medicine*, 20(7), pp.764-768.

MA, J., DUAN, M., SUN, L., YAN, M., LIU, T., WANG, Q., LIU, C., WANG, X., KANG, X., PEI, S., ZONG, D., CHEN, X., WANG, N. AND AI, J. (2015). Cardiac over-expression of microRNA-1 induces impairment of cognition in mice. *Neuroscience*, 299, pp.66-78.

MAYER, B. (2011). *Bioinformatics for omics data*. New York, NY: Humana.

MCINTYRE, L., LOPIANO, K., MORSE, A., AMIN, V., OBERG, A., YOUNG, L. AND NUZHDIN, S. (2011). RNA-seq: technical variability and sampling. *BMC Genomics*, 12(1), p.293.

MCKENNA, S., MEYER, M., GREGG, C. AND GERBER, S. (2016). s-CorrPlot: An Interactive Scatterplot for Exploring Correlation. *Journal of Computational and Graphical Statistics*, 25(2), pp.445-463.

MIRDB.ORG. (2016). *miRDB - MicroRNA Target Prediction And Functional Study Database*. [online] Available at: http://mirdb.org/miRDB/ [Accessed 13 Jun. 2016].

MONFORT, P., GOMEZ-GIMENEZ, B., LLANSOLA, M. AND FELIPO, V. (2015). Gender Differences in Spatial Learning, Synaptic Activity, and Long-Term Potentiation in the Hippocampus in Rats: Molecular Mechanisms. *ACS Chem. Neurosci.*, 6(8), pp.1420-1427.

NUEDA, M., FERRER, A. AND CONESA, A. (2011). ARSyN: a method for the identification and removal of systematic noise in multifactorial time course microarray experiments. *Biostatistics*, 13(3), pp.553-566.

OLTON, D. AND SAMUELSON, R. (1976). Remembrance of places passed: Spatial memory in rats. *Journal of Experimental Psychology: Animal Behavior Processes*, 2(2), pp.97-116.

PAUL, V., BALASUBRAMANIAM, E., JAYAKUMAR, A. AND KAZI, M. (1995). A sex-related difference in the neurobehavioral and hepatic effects following chronic endosulfan treatment in rats. *European Journal of Pharmacology: Environmental Toxicology and Pharmacology*, 293(4), pp.355-360.

PEARSON, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11), pp.559-572.

PICARDI, E. (2015). *RNA bioinformatics*. New York, NY: Humana Press.

R-PROJECT.ORG. (2016). *R: The R Project for Statistical Computing*. [online] Available at: https://www.r-project.org/ [Accessed 13 Jun. 2016].

RACINE, J. (2011). RStudio: A Platform-Independent IDE for R and Sweave. *J. Appl. Econ.*, 27(1), pp.167-172.

RITCHIE, M., PHIPSON, B., WU, D., HU, Y., LAW, C., SHI, W. AND SMYTH, G. (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Research*, 43(7), pp.e47-e47.

RODRÍGUEZ, A., ORTEGA, A., BERUMEN, L., GARCÍA-ALCOCER, M., GIMÉNEZ, C. AND ZAFRA, F. (2014). Expression of the System N transporter (SNAT5/SN2) during development indicates its plausible role in glutamatergic neurotransmission. *Neurochemistry International*, 73, pp.166-171.

SAITO, K., KAKIZAKI, T., KATAOKA, H., MISHINA, M. AND YANAGAWA, Y. (2010). Motor dysfunctions of striatal vesicular GABA transporter knockout mice. *Neuroscience Research*, 68, p.e145.

SANDYK, R., IACONO, R. AND BAMFORD, C. (1987). The hypothalamus in Parkinson Disease. *The Italian Journal of Neurological Sciences*, 8(3), pp.227-234.

SHEN, L., SUN, C., LI, Y., LI, X., SUN, T., LIU, C., ZHOU, Y. AND DU, Z. (2015). MicroRNA-199a-3p suppresses glioma cell proliferation by regulating the AKT/mTOR signaling pathway. *Tumor Biol.*, 36(9), pp.6929-6938.

SONESON, C. AND DELORENZI, M. (2013). A comparison of methods for differential expression analysis of RNA-seq data. *BMC Bioinformatics*, 14(1), p.91.

STATEGRA.EU. (2016). *STATegra*. [online] Available at: http://stategra.eu [Accessed 13 Jun. 2016].

TARAZONA, S., FURIÓ-TARÍ, P., TURRÀ, D., PIETRO, A., NUEDA, M., FERRER, A. AND CONESA, A. (2015). Data quality aware analysis of differential expression in RNA-seq with NOISeq R/Bioc package. *Nucleic Acids Res*, p.gkv711.

TARAZONA, S., GARCÍA, F., FERRER, A., DOPAZO, J. AND CONESA, A. (2012). NOIseq: a RNA-seq differential expression method robust for sequencing depth biases. *EMBnet.journal*, 17(B), p.18.

VAN ASSCHE, R., BROECKX, V., BOONEN, K., MAES, E., DE HAES, W., SCHOOFS, L. AND TEMMERMAN, L. (2015). Integrating -Omics: Systems Biology as Explored Through C. elegans Research. *Journal of Molecular Biology*, 427(21), pp.3441-3451.

WANG, Z., GERSTEIN, M. AND SNYDER, M. (2009). RNA-Seq: a revolutionary tool for transcriptomics. *Nat Rev Genet*, 10(1), pp.57-63.

WILSON, W., SHAPIRO, L., BRADNER, J. AND CAUDLE, W. (2014). Developmental exposure to the organochlorine insecticide endosulfan damages the nigrostriatal dopamine system in male offspring. *NeuroToxicology*, 44, pp.279-287.

WINKLHOFER, K. AND HAASS, C. (2010). Mitochondrial dysfunction in Parkinson's disease. *Biochimica et Biophysica Acta (BBA) - Molecular Basis of Disease*, 1802(1), pp.29-44.

WU, T. AND HALLETT, M. (2013). The cerebellum in Parkinson's disease. *Brain*, 136(3), pp.696-709.

XING, F., SHARMA, S., LIU, Y., MO, Y., WU, K., ZHANG, Y., POCHAMPALLY, R., MARTINEZ, L., LO, H. AND WATABE, K. (2015). miR-509 suppresses brain metastasis of breast cancer cells by modulating RhoC and TNF-α. *Oncogene*, 34(37), pp.4890-4900.

YANG, Z., ZHONG, L., XIAN, R. AND YUAN, B. (2015). MicroRNA-223 regulates inflammation and brain injury via feedback to NLRP3 inflammasome after intracerebral hemorrhage. *Molecular Immunology*, 65(2), pp.267-276.

ZOU, H., DING, Y., SHI, W., XU, X., GONG, A., ZHANG, Z. AND LIU, J. (2014). MicroRNA-29c/PTEN Pathway is Involved in Mice Brain Development and Modulates Neurite Outgrowth in PC12 Cells. *Cellular and Molecular Neurobiology*, 35(3), pp.313-322.

7. Attachments

7.1. Scripts

7.1.1. Attachment I: Master data set creation script

```
### IMPORTACIONES ###
library("XLConnect")
library("stats")
library("devtools")
library("FactoMineR")
library("RColorBrewer")
library("NOISeq")
library("openxlsx")

### DIRECTORIOS ###
dir_data_met <- "/Users/Elena/Dropbox/TFG/Analisis_WD/met/data"
dir_data_prot <- "/Users/Elena/Dropbox/TFG/Analisis_WD/prot/data"
dir_data_clinical <- "/Users/Elena/Dropbox/TFG/Analisis_WD/clinical"
dir_output <- "/Users/Elena/Dropbox/TFG/Analisis_WD"

### IMPORT METABOLOMIC DATA ###
setwd(dir_data_met)

## CYP ##
data_CYP <- readWorksheetFromFile("Individual data CYP.xlsx", sheet = 1)
rownames(data_CYP) <- data_CYP[,1] #Add names
data_CYP <- data_CYP[,4:length(colnames(data_CYP))] #Quitamos las columnas con los codigos de KEGG y
otros

## END_ST ##
data_END_ST <- readWorksheetFromFile("Individual data Endosulfan males_females_final.xlsx", sheet = 1,
startRow = 3)
rownames(data_END_ST) <- data_END_ST[,1] #Add names
data_END_ST <- data_END_ST[,2:length(colnames(data_END_ST))]

## END_HP ##
data_END_HP <- readWorksheetFromFile("Individual data Endosulfan males_females_final.xlsx", sheet = 2,
startRow = 3)
rownames(data_END_HP) <- data_END_HP[,1] #Add names
data_END_HP <- data_END_HP[,2:length(colnames(data_END_HP))]

## END_CB ##
data_END_CB <- readWorksheetFromFile("Individual data Endosulfan males_females_final.xlsx", sheet = 3,
startRow = 3)
rownames(data_END_CB) <- data_END_CB[,1] #Add names
data_END_CB <- data_END_CB[,2:length(colnames(data_END_CB))]

## END_CX ##
data_END_CX <- readWorksheetFromFile("Individual data Endosulfan males_females_final.xlsx", sheet = 4,
startRow = 3)
rownames(data_END_CX) <- data_END_CX[,1] #Add names
data_END_CX <- data_END_CX[,2:length(colnames(data_END_CX))]


### IMPORT PROTEOMIC DATA ###
setwd(dir_data_prot)
data_prot <- read.xlsx("DataMat_Denamic_Set1.xlsx", 1, startRow = 5, cols = c(c(1), c(5:103)), colNames
= TRUE, rowNames = TRUE)
data_prot <- as.data.frame(sapply(data_prot, as.numeric), row.names = rownames(data_prot))


### IMPORT CLINICAL DATA ###
```

```
setwd(dir_data_clinical)
data_clinical <- readWorksheetFromFile("Table Analysis Behaviour-Proteomics April 2015 (Ratas
analizadas proteomicamente).xlsx", sheet = 2)

### HOMOGENIZACION DE LOS NOMBRES ###
list_data_met <- list(data_CYP, data_END_CB, data_END_CX, data_END_HP, data_END_ST)
for (i in 1:length(list_data_met)) {
  names <- c()
  for (j in 1:length(colnames(list_data_met[[i]]))) {
    split <- unlist(strsplit(colnames(list_data_met[[i]])[j], split = "\\."))
    t <- split[1]
    s <- split[2]
    n <- split[3]
    tis <- substr(split[4], 1, 2)
    name <- paste(t, s, n, tis, sep = " ")
    names <- c(names, name)
  }
  colnames(list_data_met[[i]]) <- names
}

names_prot <- c()
treatment <- c()
sex <- c()
tissue <- c()
numb <- c()
for (i in 1:length(colnames(data_prot))) {
  split <- unlist(strsplit(colnames(data_prot)[i], split = "\\."))
  t <- split[1]
  s <- split[2]
  n <- paste(split[3], split[4], sep = "")
  tis <- split[6]
  treatment <- c(treatment, t)
  sex <- c(sex, s)
  numb <- c(numb, n)
  tissue <- c(tissue, tis)
  name <- paste(t, s, n, tis, sep = " ")
  names_prot <- c(names_prot, name)
}
colnames(data_prot) <- names_prot

names_clinical <- c()
for (i in 1:length(rownames(data_clinical))){
  s <- substr(data_clinical[i, 1], 1, 1)
  t <- data_clinical[i, 8]
  n <- data_clinical[i, 2]
  name <- paste(t, s, n, sep = " ")
  names_clinical <- c(names_clinical, name)
}
rownames(data_clinical) <- names_clinical
data_clinical <- data_clinical[,c(3:7)]


### NEW DATA FRAME ###
master_frame <- as.data.frame(t(data_prot)) #Proteomics added to master frame
#Now we add the metabolic contents to the new master frame
for (i in 1:length(list_data_met)) {
  for (j in 1:length(rownames(list_data_met[[i]]))) {
    metabolite <- rownames(list_data_met[[i]])[j]
    for (z in 1:length(colnames(list_data_met[[i]]))) {
      individual <- colnames(list_data_met[[i]])[z]
      master_frame[individual, metabolite] <- list_data_met[[i]][j, z]
    }
  }
}

# INFO REGARDING INDIVIDUALS
master_frame[, "Treatment"] <- treatment
master_frame[, "Sex"] <- sex
master_frame[, "Tissue"] <- tissue
master_frame[, "Number"] <- numb
```

```
# INFO CLINICAL DATA
for (i in 1:length(rownames(master_frame))) {
  for (j in 1:length(rownames(data_clinical))) {
    full_name <- rownames(master_frame)[i]
    name <- rownames(data_clinical)[j]
    if (grepl(name, full_name)) {
      master_frame[i,colnames(data_clinical)] <- data_clinical[j,]
    }
  }
}

### OUTPUT ###
setwd(dir_output)
write.table(master_frame, "master_frame.txt", sep="\t")
```

7.1.2. Attachment II: Pre-processing and exploration of Set 01 script

```
#Script para realizar PCA sobre datos pre-procesados
library(NOISeq)

### DIRECTORIOS ###
dir_analisis <- "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01"
dir_data <- "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/data"
dir_output <- "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_exp/"

### IMPORT DATA ###
setwd(dir_data)
metab = read.delim("metabolomics01.txt", header = TRUE, as.is = TRUE, row.names = 1, check.names =
FALSE, dec = ",")
metab2 = metab[-grep("END F I2 CX", rownames(metab)),]
proteom = read.delim("proteomics01.txt", header = TRUE, as.is = TRUE, row.names = 1,
                     check.names = FALSE, dec = ",")

metabTissue = lapply(c(" CX", " HP", " ST", " CB"), function (x) metab2[grep(x, rownames(metab2)),])
names(metabTissue) = c("CX", "HP", "ST", "CB")
proteomTissue = lapply(c(" CX", " HP", " ST", " CB"), function (x) proteom[grep(x,
rownames(proteom)),])
names(proteomTissue) = c("CX", "HP", "ST", "CB")

# Count and remove missing values ----------------------------------------

NAmetab = apply(metab2, 2, function(x) sum(is.na(x)))
NAproteom = apply(proteom, 2, function(x) sum(is.na(x)))

metab25 = metab2[,-which(NAmetab > 0.25*nrow(metab2))]; dim(metab25)  # 98 98
proteom25 = proteom[,-which(NAproteom > 0.25*nrow(proteom))]; dim(proteom25)  # 99 722

## Per tissue
NAmetabTissue = lapply(metabTissue, function (y) apply(y, 2, function(x) sum(is.na(x))))
NAproteomTissue = lapply(proteomTissue, function (y) apply(y, 2, function(x) sum(is.na(x))))

metabTissue25 = lapply(1:length(metabTissue),
                       function (i) metabTissue[[i]][,-which(NAmetabTissue[[i]] >
0.25*nrow(metabTissue[[i]]))])
proteomTissue25 = lapply(1:length(proteomTissue),
                         function (i) proteomTissue[[i]][,-which(NAproteomTissue[[i]] >
0.25*nrow(proteomTissue[[i]]))])
names(metabTissue25) = names(proteomTissue25) = names(metabTissue)

# Missing value imputation -----------------------------------------------
library(impute)

metabKNN = impute.knn(t(metab25), k = 2, rowmax = 0.5, colmax = 0.8, maxp = 1500,
rng.seed=362436069)$data
proteomKNN = impute.knn(t(proteom25), k = 2, rowmax = 0.5, colmax = 0.8, maxp = 1500,
rng.seed=362436069)$data

metabTissueKNN = lapply(metabTissue25,
                        function (x) impute.knn(t(x), k = 2, rowmax = 0.5, colmax = 0.8, maxp = 1500,
rng.seed=362436069)$data)
proteomTissueKNN = lapply(proteomTissue25,
                          function (x) impute.knn(t(x), k = 2, rowmax = 0.5, colmax = 0.8, maxp = 1500,
rng.seed=362436069)$data)

# Quantile normalization of metabolomics ---------------------------------

metabTissueKNN = lapply(metabTissueKNN, function(x) normalizeBetweenArrays(x, method="quantile"))

# Arsynseq ---------------------------------------------------------------

setwd(dir_data)
charac = read.delim("characteristics01.txt", header = TRUE, as.is = TRUE, row.names = 1, check.names =
FALSE)
charac[,"Number"] = sapply(charac[,"Number"], function (x) substr(x, start = 1, stop = nchar(x)-1))
```

```
charac[,"TSTN"] = apply(charac[,c("Treatment", "Sex", "Tissue", "Number")], 1, paste, collapse = "_")

charac_l_m = list(1:4)
for (i in 1:length(metabTissueKNN)) {
  charac_l_m[[i]] = charac[colnames(metabTissueKNN[[i]]),]
}
metabTissueNOISeq = list(1:length(metabTissueKNN))
for (i in 1:length(metabTissueKNN)) {
  metabTissueNOISeq[[i]] = readData(metabTissueKNN[[i]], charac_l_m[[i]])
}
metabTissueNoNoise = lapply(metabTissueNOISeq, function (x) ARSyNseq(x, factor = "TSTN", norm = "n",
logtransf = FALSE))
names(metabTissueNoNoise) = names(metabTissueKNN)
metabTissueNoNoise = lapply(metabTissueNoNoise, function(x) x@assayData$exprs)


charac_l_p = list(1:4)
for (i in 1:length(proteomTissueKNN)) {
  charac_l_p[[i]] = charac[colnames(proteomTissueKNN[[i]]),]
}
proteomTissueNOISeq = list(1:length(proteomTissueKNN))
for (i in 1:length(proteomTissueKNN)) {
  proteomTissueNOISeq[[i]] = readData(proteomTissueKNN[[i]], charac_l_p[[i]])
}
proteomTissueNoNoise = lapply(proteomTissueNOISeq, function (x) ARSyNseq(x, factor = "TSTN", norm =
"n", logtransf = TRUE))
names(proteomTissueNoNoise) = names(proteomTissueKNN)
proteomTissueNoNoise = lapply(proteomTissueNoNoise, function(x) x@assayData$exprs)


setwd(dir_data)
for (i in 1:length(proteomTissueNoNoise)) {
  write.table(proteomTissueNoNoise[[i]], paste(names(proteomTissueNoNoise)[i], "prot_NoNoiseData.txt",
sep = "_"), sep = "\t")
  write.table(metabTissueNoNoise[[i]], paste(names(metabTissueNoNoise)[i], "met_NoNoiseData.txt", sep =
"_"), sep = "\t")
}

# Graphical representations ------------------------------------------------
setwd(dir_output)
pdf(file = "heatmaps_norm.pdf", width = 3.5*4, height = 3.5*3)
par(mfcol = c(1,1))
for (i in 1:length(metabTissueKNN)) {
  heatmap(metabTissueKNN[[i]], main = paste("Metabolomics (", names(metabTissueKNN)[i], ")", sep =""),
          margins = c(7,3))
  heatmap(proteomTissueKNN[[i]], main = paste("Proteomics (", names(proteomTissueKNN)[i], ")", sep
=""),
          margins = c(7,3))
}
dev.off()

miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]
pdf(file = "boxplots_norm.pdf", width = 3.5*6, height = 3.5*3)
par(mfcol = c(2,4), mar = c(10,5,5,5))
for (i in 1:length(metabTissueKNN)) {
  sorted_m = metabTissueKNN[[i]][,order(colnames(metabTissueKNN[[i]]))]
  sorted_p = proteomTissueKNN[[i]][,order(colnames(proteomTissueKNN[[i]]))]
  if (i == 1) { #cortex has one less sample in metabolomics
    boxplot(sorted_m, main = paste("Metabolomics (", names(metabTissueKNN)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 10, 1, 1, 1, 1, 11, 11, 11, 11, 11, 2, 2, 2, 2, 9, 9, 9, 9,
7, 7, 7)])
  } else {
    boxplot(sorted_m, main = paste("Metabolomics (", names(metabTissueKNN)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 10, 1, 1, 1, 1, 11, 11, 11, 11, 11, 11, 2, 2, 2, 2, 9, 9, 9,
9, 7, 7, 7)])
  }
  boxplot(sorted_p, main = paste("Proteomics (", names(proteomTissueKNN)[i], ")", sep =""), las = 2,
          col = miscolores[c(10, 10, 10, 10, 1, 1, 1, 1, 11, 11, 11, 11, 11, 11, 2, 2, 2, 2, 9, 9, 9,
9, 7, 7, 7)])
```

```
}
dev.off()

# PCA ----------------------------------------------------------------------

### LOG + CENTER ###
#Per tissue
protTissue_data = lapply(proteomTissueNoNoise, function (x) scale(x, center = TRUE, scale = FALSE))
metabTissue_data = lapply(metabTissueNoNoise, function (x) scale(log(x), center = TRUE, scale = TRUE))
protTissue_data = lapply(protTissue_data, function (x) t(x))
metabTissue_data = lapply(metabTissue_data, function (x) t(x))

### APLICAMOS PCA ###
data2pca = list("Metabolomics (Cortex)" = metabTissue_data[[1]],
                "Metabolomics (Hippocampus)" = metabTissue_data[[2]],
                "Metabolomics (Striatum)" = metabTissue_data[[3]],
                "Metabolomics (Cerebellum)" = metabTissue_data[[4]],
                "Proteomics (Cortex)" = protTissue_data[[1]],
                "Proteomics (Hippocampus)" = protTissue_data[[2]],
                "Proteomics (Striatum)" = protTissue_data[[3]],
                "Proteomics (Cerebellum)" = protTissue_data[[4]])
pca.results = lapply(data2pca, PCA.GENES)


### EXPLAINED VARIANCE ###
setwd(dir_output)
samp <- c()
for (i in 1:length(data2pca)) {
  samp <- c(samp, length(rownames(data2pca[[i]])))
}

pdf(file = "explainedvariance_mother_arsynseq_norm_log.pdf", width = 3.5*4, height = 3.5*2)
par(mfcol = c(2,4))
for (i in 1:length(pca.results)) {
  barplot(pca.results[[i]]$var.exp[,1], names = 1:samp[i],
  xlab = "PC", ylab = "explained variance", ylim = c(0,0.7),
  main = names(pca.results)[i])
}
dev.off()

#### COLORS AND SHAPES ###
miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]
charac_l = list(1:length(data2pca))
for (i in 1:length(data2pca)) {
  charac_l[[i]] = charac[rownames(data2pca[[i]]),]
}

#Pesticides as colors
col.pest <- miscolores[1:3]
pesticides <- charac[,"Treatment"]
names(pesticides) <- rownames(charac)
names(col.pest) <- unique(pesticides)
mycol = col.pest[pesticides]

#Sex as shapes and tissues as filled-in shapes or empty shapes
myshapes = c(0, 15, 2, 17, 1, 16, 5, 18)
group_l <- list()
for (j in 1:length(data2pca)) {
  group <- c()
  for (i in 1:length(rownames(charac_l[[j]]))) {
    group <- c(group, paste(charac_l[[j]][i,"Sex"], charac_l[[j]][i, "Number"], sep = "-"))
  }
  group_l[[j]] <- group
}
mypch_l = list()
for (j in 1:length(data2pca)) {
  pch.group = myshapes[1:length(unique(group_l[[j]]))]
  names(pch.group) = unique(group_l[[j]])
  mypch = pch.group[group_l[[j]]]
  mypch_l[[j]] = mypch
}
```

```
setwd(dir_output)

### LOADINGS PLOT ###
pdf(file = "PCAloadings12_mother_arsynseq_norm_log.pdf", width = 3.5*4, height = 3.5*2)
par(mfcol = c(2,4))
for (i in 1:length(pca.results)) {
  plot(pca.results[[i]]$loadings[,1:2], col="white", cex = 0.5,
       xlab = paste("PCA 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0), "%", sep=""),
       ylab = paste("PCA 2 ", round(pca.results[[i]]$var.exp[2,1]*100,0), "%", sep=""),
       main = names(data2pca)[i],
       xlim = range(pca.results[[i]]$loadings[,1:2]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1),
       ylim = range(pca.results[[i]]$loadings[,1:2]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1))

  points(pca.results[[i]]$loadings[,1], pca.results[[i]]$loadings[,2], pch = 0)
}
dev.off()

pdf(file = "PCAloadings13_mother_arsynseq_norm_log.pdf", width = 3.5*4, height = 3.5*2)
par(mfcol = c(2,4))
for (i in 1:length(pca.results)) {
  plot(pca.results[[i]]$loadings[,1:3], col="white", cex = 0.5,
       xlab = paste("PCA 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0), "%", sep=""),
       ylab = paste("PCA 3 ", round(pca.results[[i]]$var.exp[2,1]*100,0), "%", sep=""),
       main = names(data2pca)[i],
       xlim = range(pca.results[[i]]$loadings[,1:3]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1),
       ylim = range(pca.results[[i]]$loadings[,1:3]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1))
  points(pca.results[[i]]$loadings[,1], pca.results[[i]]$loadings[,2], pch = 0)
}
dev.off()

### PCA SCORES PLOT (WITH COLORS AND SHAPES) ###
pdf("PCAscores12_mother_arsynseq_norm_log.pdf", width = 3.5*4, height = 3.5*2)
par(mfcol = c(2,4))
for (i in 1:length(pca.results)) {
  rango = diff(range(pca.results[[i]]$scores[,1:2]))

  plot(pca.results[[i]]$scores[,1:2], col = "white",
       xlab = paste("PC 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0),
                    "%", sep = ""),
       ylab = paste("PC 2 ", round(pca.results[[i]]$var.exp[2,1]*100,0),
                    "%", sep = ""),
       main = names(data2pca)[i],
       xlim = range(pca.results[[i]]$scores[,1:2]) + 0.02*rango*c(-1,1),
       ylim = range(pca.results[[i]]$scores[,1:2]) + 0.02*rango*c(-1,1))

  points(pca.results[[i]]$scores[,1], pca.results[[i]]$scores[,2],
         pch = mypch_l[[i]], col = mycol, cex = 1.5)
  legend("topright", c("END", "CYP", "VH"), col = col.pest, pch = 19, bty = "o", ncol = 2, box.col =
"black")
  legend("right", c("M", "F"), pch = c(0, 15), bty = "o", ncol = 2)
  if (i == 1 | i== 2 | i == 5 | i == 6) {
    legend("bottomright", c("II", "I", "III"), pch = c(0, 2, 1), bty = "o", ncol = 2)
  } else {
    legend("bottomright", c("I", "II", "III"), pch = c(0, 2, 1), bty = "o", ncol = 2)
  }
}
dev.off()

pdf("PCAscores13_mother_arsynseq_norm_log.pdf", width = 3.5*4, height = 3.5*2)
par(mfcol = c(2,4))
for (i in 1:length(pca.results)) {
  rango2 = diff(range(pca.results[[i]]$scores[,c(1,3)]))
  plot(pca.results[[i]]$scores[,c(1,3)], col = "white",
       xlab = paste("PC 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0),
                    "%", sep = ""),
       ylab = paste("PC 3 ", round(pca.results[[i]]$var.exp[3,1]*100,0),
```

```
                    "%", sep = ""),
        main = names(data2pca)[i],
        xlim = range(pca.results[[i]]$scores[,c(1,3)]) + 0.02*rango2*c(-1,1),
        ylim = range(pca.results[[i]]$scores[,c(1,3)]) + 0.02*rango2*c(-1,1))

  points(pca.results[[i]]$scores[,1], pca.results[[i]]$scores[,3],
         pch = mypch_l[[i]], col = mycol, cex = 1.5)
  legend("topright", c("END", "CYP", "VH"), col = col.pest, pch = 19, bty = "o", ncol = 2, box.col =
"black")
  legend("right", c("M", "F"), pch = c(0, 15), bty = "o", ncol = 2)
  if (i == 1 | i== 2 | i == 5 | i == 6) {
    legend("bottomright", c("II", "I", "III"), pch = c(0, 2, 1), bty = "o", ncol = 2)
  } else {
    legend("bottomright", c("I", "II", "III"), pch = c(0, 2, 1), bty = "o", ncol = 2)
  }
}
dev.off()
```

7.1.3. Attachment III: Differential expression analysis for Set 01 script

```
# Differential expression analysis for metabolomics and proteomics data of DENAMIC Set 01.
# Analysis by tissues.
# Only VH and END data will be considered, to compare with transcriptomics data.

library(plyr)

dir_analisis <- "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01"
dir_data <- "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/data"
dir_output <- "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_DE/"

# Data import ------------------------------------------------------------

#Import data as rows for prots/metabolites and columns for samples
setwd(dir_data)
met_CB = read.delim("CB_met_NoNoiseData_norm.txt", header = TRUE, as.is = TRUE, row.names = 1,
check.names = FALSE)
met_HP = read.delim("HP_met_NoNoiseData_norm.txt", header = TRUE, as.is = TRUE, row.names = 1,
check.names = FALSE)
prot_CB = read.delim("CB_prot_NoNoiseData.txt", header = TRUE, as.is = TRUE, row.names = 1, check.names
= FALSE)
prot_HP = read.delim("HP_prot_NoNoiseData.txt", header = TRUE, as.is = TRUE, row.names = 1, check.names
= FALSE)

met = list("CB" = met_CB, "HP" = met_HP)
prot = list("CB" = prot_CB, "HP" = prot_HP)

# Leave only END and VH --------------------------------------------------

met = lapply(met, function(x) x[,-grep("CYP", colnames(x))])
prot = lapply(prot, function(x) x[,-grep("CYP", colnames(x))])

# Model design -----------------------------------------------------------

charac = read.delim("characteristics01.txt", header = TRUE, as.is = TRUE, row.names = 1, check.names =
FALSE)
charac[,"Number"] = sapply(charac[,"Number"], function (x) substr(x, start = 1, stop = nchar(x)-1))

sex_m_l = list()
pest_m_l = list()
mother_m_l = list()
sex_p_l = list()
pest_p_l = list()
mother_p_l = list()
for (i in 1:2) {
  sex_m_l[[i]] = factor(charac[colnames(met[[i]]), "Sex"])
  pest_m_l[[i]] = factor(charac[colnames(met[[i]]), "Treatment"], levels = c("VH","END"))
  mother_m_l[[i]] = factor(charac[colnames(met[[i]]), "Number"])
  sex_p_l[[i]] = factor(charac[colnames(prot[[i]]), "Sex"])
  pest_p_l[[i]] = factor(charac[colnames(prot[[i]]), "Treatment"], levels = c("VH","END"))
  mother_p_l[[i]] = factor(charac[colnames(prot[[i]]), "Number"])
}

met_matrix = list()
prot_matrix = list()
for (i in 1:2) {
  met_matrix[[i]] = model.matrix(~ sex_m_l[[i]] + pest_m_l[[i]] + sex_m_l[[i]] * pest_m_l[[i]])
  prot_matrix[[i]] = model.matrix(~ sex_p_l[[i]] + pest_p_l[[i]] + sex_p_l[[i]] * pest_p_l[[i]])
}
names(met_matrix) = c("CB", "HP")
names(prot_matrix) = c("CB", "HP")

# Voom transformation ----------------------------------------------------
m_trans_l = list()
p_trans_l = prot
for (i in 1:2) {
  m_trans_l[[i]] <- voom(met[[i]], met_matrix[[i]],plot=TRUE)
}
```

```
# Limma pipeline ------------------------------------------------------------
fit_m = list()
fit_p = list()
for (i in 1:2) {
  fit_m[[i]] = lmFit(m_trans_l[[i]], met_matrix[[i]])
  fit_m[[i]] = eBayes(fit_m[[i]])
  fit_p[[i]] = lmFit(p_trans_l[[i]], prot_matrix[[i]])
  fit_p[[i]] = eBayes(fit_p[[i]])
}

# Venn diagrams -------------------------------------------------------------

###Adjusted p-value
setwd(dir_output)
miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]

for (i in 1:2) {
  results_m = decideTests(fit_m[[i]])[,c(3,4)]
  results_p = decideTests(fit_p[[i]])[,c(3,4)]

  pdf(paste(names(met_matrix)[i], "metab_Venn_adj.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_m, names = c("Intercept", "Sex", "Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()

  pdf(paste(names(met_matrix)[i], "proteom_Venn_adj.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_p, names = c("Intercept", "Sex", "Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()
}

###P-value without adjusting
for (i in 1:2) {
  results_m = decideTests(fit_m[[i]], adjust.method = "none", p.value=0.05)[,c(3,4)]
  results_p = decideTests(fit_p[[i]], adjust.method = "none", p.value=0.05)[,c(3,4)]

  pdf(paste(names(met_matrix)[i], "metab_Venn.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_m, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()

  pdf(paste(names(met_matrix)[i], "proteom_Venn.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_p, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()
}

### PDFs with everything
pdf("all_Venn_0.05.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2))
for (i in 1:2) {
  results_m = decideTests(fit_m[[i]], adjust.method = "none", p.value=0.05)[,c(3,4)]
  results_p = decideTests(fit_p[[i]], adjust.method = "none", p.value=0.05)[,c(3,4)]

  vennDiagram(results_m, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(met_matrix)[i], "(Metabolomics)", sep = " "), outer = FALSE)

  vennDiagram(results_p, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(met_matrix)[i], "(Proteomics)", sep = " "), outer = FALSE)
}
dev.off()

pdf("all_Venn_adj.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2))
for (i in 1:2) {
  results_m = decideTests(fit_m[[i]])[,c(3,4)]
  results_p = decideTests(fit_p[[i]])[,c(3,4)]
```

```
  vennDiagram(results_m, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(met_matrix)[i], "(Metabolomics)", sep = " "), outer = FALSE)

  vennDiagram(results_p, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(met_matrix)[i], "(Proteomics)", sep = " "), outer = FALSE)
}
dev.off()

# Create tables for future use ------------------------------------------

met_p_CB = list()
met_p_HP = list()
prot_p_CB = list()
prot_p_HP = list()

for (j in 1:4) {
    met_p_CB[[j]] = topTable(fit_m[[1]], coef = j, number = nrow(fit_m[[1]]))[,c("P.Value",
"adj.P.Val")]
    met_p_HP[[j]] = topTable(fit_m[[2]], coef = j, number = nrow(fit_m[[2]]))[,c("P.Value",
"adj.P.Val")]
    prot_p_CB[[j]] = topTable(fit_p[[1]], coef = j, number = nrow(fit_p[[1]]))[,c("P.Value",
"adj.P.Val")]
    prot_p_HP[[j]] = topTable(fit_p[[2]], coef = j, number = nrow(fit_p[[2]]))[,c("P.Value",
"adj.P.Val")]
}

names(met_p_CB) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")
names(met_p_HP) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")
names(prot_p_CB) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")
names(prot_p_HP) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")


setwd(dir_output)
for (i in 1:4) {
  write.table(met_p_CB[[i]], paste(names(met_p_CB)[i], "met_CB.txt", sep ="_"), sep = "\t", quote =
FALSE)
  write.table(met_p_HP[[i]], paste(names(met_p_HP)[i], "met_HP.txt", sep ="_"), sep = "\t", quote =
FALSE)
  write.table(prot_p_CB[[i]], paste(names(prot_p_CB)[i], "prot_CB.txt", sep ="_"), sep = "\t", quote =
FALSE)
  write.table(prot_p_HP[[i]], paste(names(prot_p_HP)[i], "prot_HP.txt", sep ="_"), sep = "\t", quote =
FALSE)
}

# Creation of average tables ------------------------------------------

met_averages = list()
prot_averages = list()

for (i in 1:2) { #1 is CB, 2 is HP
  VH_F_m = rowMeans(met[[i]][,grep("VH F ", colnames(met[[i]]))])
  END_F_m = rowMeans(met[[i]][,grep("END F ", colnames(met[[i]]))])
  VH_M_m = rowMeans(met[[i]][,grep("VH M ", colnames(met[[i]]))])
  END_M_m = rowMeans(met[[i]][,grep("END M ", colnames(met[[i]]))])
  met_averages[[i]] = data.frame(VH_F_m, END_F_m, VH_M_m, END_M_m)

  VH_F_p = rowMeans(prot[[i]][,grep("VH F ", colnames(prot[[i]]))])
  END_F_p = rowMeans(prot[[i]][,grep("END F ", colnames(prot[[i]]))])
  VH_M_p = rowMeans(prot[[i]][,grep("VH M ", colnames(prot[[i]]))])
  END_M_p = rowMeans(prot[[i]][,grep("END M ", colnames(prot[[i]]))])
  prot_averages[[i]] = data.frame(VH_F_p, END_F_p, VH_M_p, END_M_p)
}

colnames(met_averages[[1]]) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M")
colnames(met_averages[[2]]) = c("HP_VH_F", "HP_END_F", "HP_VH_M", "HP_END_M")
colnames(prot_averages[[1]]) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M")
colnames(prot_averages[[2]]) = c("HP_VH_F", "HP_END_F", "HP_VH_M", "HP_END_M")
```

```
#Export mean tables (individual tissues)
setwd(dir_output)
write.table(met_averages[[1]], "metabolomics_averages_CB.txt", sep = "\t")
write.table(met_averages[[2]], "metabolomics_averages_HP.txt", sep = "\t")
write.table(prot_averages[[1]], "proteomics_averages_CB.txt", sep = "\t")
write.table(prot_averages[[2]], "proteomics_averages_HP.txt", sep = "\t")

met_means = t(rbind.fill(as.data.frame(t(met_averages[[1]])), as.data.frame(t(met_averages[[2]]))))
prot_means = t(rbind.fill(as.data.frame(t(prot_averages[[1]])), as.data.frame(t(prot_averages[[2]]))))
colnames(met_means) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M", "HP_VH_F", "HP_END_F", "HP_VH_M",
"HP_END_M")
colnames(prot_means) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M", "HP_VH_F", "HP_END_F",
"HP_VH_M", "HP_END_M")

#Export mean tables (both tissues)
write.table(met_means, "metabolomics_averages.txt", sep = "\t", quote = FALSE)
write.table(prot_means, "proteomics_averages.txt", sep = "\t", quote = FALSE)

# Log2Ratio tables ----------------------------------------------------------


CB_met = transform(met_averages[[1]], log2_CB_F = log2(met_averages[[1]][,2]/met_averages[[1]][,1]),
                   log2_CB_M = log2(met_averages[[1]][,4]/met_averages[[1]][,3]))[,c(5,6)]
HP_met = transform(met_averages[[2]], log2_CB_F = log2(met_averages[[2]][,2]/met_averages[[2]][,1]),
                   log2_CB_M = log2(met_averages[[2]][,4]/met_averages[[2]][,3]))[,c(5,6)]

CB_prot = transform(prot_averages[[1]], log2_CB_F = prot_averages[[1]][,2]-prot_averages[[1]][,1],
                   log2_CB_M = prot_averages[[1]][,4]-prot_averages[[1]][,3])[,c(5,6)]
HP_prot = transform(prot_averages[[2]], log2_CB_F = prot_averages[[2]][,2]-prot_averages[[2]][,1],
                   log2_CB_M = prot_averages[[2]][,4]-prot_averages[[2]][,3])[,c(5,6)]

#Export for each individual tissue
write.table(CB_met, "metabolomics_log2_CB.txt", sep = "\t", quote = FALSE)
write.table(HP_met, "metabolomics_log2_HP.txt", sep = "\t", quote = FALSE)
write.table(CB_prot, "proteomics_log2_CB.txt", sep = "\t", quote = FALSE)
write.table(HP_prot, "proteomics_log2_HP.txt", sep = "\t", quote = FALSE)

met_log2 = t(rbind.fill(as.data.frame(t(CB_met)), as.data.frame(t(HP_met))))
prot_log2 = t(rbind.fill(as.data.frame(t(CB_prot)), as.data.frame(t(HP_prot))))
colnames(met_log2) = c("log2_CB_F", "log2_CB_M", "log2_HP_F", "log2_HP_M")
colnames(prot_log2) = c("log2_CB_F", "log2_CB_M", "log2_HP_F", "log2_HP_M")

write.table(met_log2, "metabolomics_log2.txt", sep = "\t", quote = FALSE)
write.table(prot_log2, "proteomics_log2.txt", sep = "\t", quote = FALSE)
```

7.1.4. Attachment IV: Initial formatting for transcriptomics script

```
dir_trans = "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Transcriptomics"
dir_data = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data"
dir_output = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_exp"

library(NOISeq)

# RNA-seq: genes? isoforms? ----------------------------------------------------------------
setwd(dir_trans)
rnaseqGenes = read.delim("Cuffdiff_RNA-seq/genes.read_group_tracking", header = TRUE, as.is = TRUE)  #
654408 rows
head(rnaseqGenes)
length(unique(rnaseqGenes$tracking_id))  # 18178

rnaseqIsoforms = read.delim("Cuffdiff_RNA-seq/isoforms.read_group_tracking", header = TRUE, as.is =
TRUE)  # 670572 rows
head(rnaseqIsoforms)
length(unique(rnaseqIsoforms$tracking_id))  # 18627

length(intersect(rnaseqGenes$tracking_id, rnaseqIsoforms$tracking_id)) # 18172

setdiff(rnaseqGenes$tracking_id, rnaseqIsoforms$tracking_id)

tail(setdiff(rnaseqIsoforms$tracking_id, rnaseqGenes$tracking_id), 50)

unique(rnaseqIsoforms$tracking_id[grep("NM_173300", rnaseqIsoforms$tracking_id)])
unique(rnaseqGenes$tracking_id[grep("NM_173300", rnaseqGenes$tracking_id)])
unique(rnaseqGenes$tracking_id[grep("_dup", rnaseqGenes$tracking_id)])


mirnaGenes = read.delim("Cuffdiff_miRNA/genes.read_group_tracking", header = TRUE, as.is = TRUE)
head(mirnaGenes)
length(unique(mirnaGenes$tracking_id))

mirnaIsoforms = read.delim("Cuffdiff_miRNA/isoforms.read_group_tracking", header = TRUE, as.is = TRUE)
head(mirnaIsoforms)
length(unique(mirnaIsoforms$tracking_id))
length(intersect(mirnaGenes$tracking_id, mirnaIsoforms$tracking_id))  # 0

# Long format --> Wide format ----------------------------------------------

######## RNA-Seq Genes
rnaseqGenes2 = rnaseqGenes[,c(1:3,7)]
pesti = sapply(rnaseqGenes2$condition, function (x) substr(x, start = 1, stop = nchar(x)-3))
sexo = sapply(rnaseqGenes2$condition, function (x) substr(x, start = nchar(x)-1, stop = nchar(x)-1))
tejido = sapply(rnaseqGenes2$condition, function (x) substr(x, start = nchar(x), stop = nchar(x)))
rnaseqGenes2 = data.frame(rnaseqGenes2, "pesti" = pesti, "sex" = sexo, "tissue" = tejido)
idSample = apply(rnaseqGenes2[,c("pesti", "sex", "tissue", "replicate")], 1, paste, collapse = "_")
rnaseqGenes2 = data.frame(rnaseqGenes2,"id" = idSample)
rnaseqGenes4reshape = rnaseqGenes2[,c("tracking_id", "id", "FPKM")]

#Genes en filas y FPKMs correspondientes a cada muestra en columnas
rnaseqGenesWide = reshape(data = rnaseqGenes4reshape, direction = "wide", timevar = c("id"),
                          idvar = c("tracking_id"))
rownames(rnaseqGenesWide) = rnaseqGenesWide[,"tracking_id"]
rnaseqGenesWide = rnaseqGenesWide[,2:length(colnames(rnaseqGenesWide))]


######## miRNA-Seq Genes
mirnaGenes2 = mirnaGenes[,c(1:3,7)]
pesti = sapply(mirnaGenes2$condition, function (x) substr(x, start = 1, stop = nchar(x)-3))
sexo = sapply(mirnaGenes2$condition, function (x) substr(x, start = nchar(x)-1, stop = nchar(x)-1))
tejido = sapply(mirnaGenes2$condition, function (x) substr(x, start = nchar(x), stop = nchar(x)))
mirnaGenes2 = data.frame(mirnaGenes2, "pesti" = pesti, "sex" = sexo, "tissue" = tejido)
idSample = apply(mirnaGenes2[,c("pesti", "sex", "tissue", "replicate")], 1, paste, collapse = "_")
mirnaGenes2 = data.frame(mirnaGenes2,"id" = idSample)
mirnaGenes4reshape = mirnaGenes2[,c("tracking_id", "id", "FPKM")]
```

```
#Genes en filas y FPKMs correspondientes a cada muestra en columnas
mirnaGenesWide = reshape(data = mirnaGenes4reshape, direction = "wide", timevar = c("id"), idvar =
c("tracking_id"))
rownames(mirnaGenesWide) = mirnaGenesWide[,"tracking_id"]
mirnaGenesWide = mirnaGenesWide[,2:length(colnames(mirnaGenesWide))]

# Export wide format ------------------------------------------------------
setwd(dir_output)
write.table(rnaseqGenesWide, "rnaseqGenesWide.txt", sep="\t")
write.table(mirnaGenesWide, "mirnaseqGenesWide.txt", sep = "\t")

# Imegen equivalents and sample characteristics ---------------------------

setwd(dir_trans)
rna_eq = read.delim("Cuffdiff_RNA-seq/read_groups.info", header = TRUE, as.is = TRUE)
mirna_eq = read.delim("Cuffdiff_miRNA/read_groups.info", header = TRUE, as.is = TRUE)

#RNA-Seq (que tambien vale para miRNA-Seq)
rna_s = sapply(rna_eq$file, function (x) strsplit(x, split = "/"))
rna_samp = sapply(rna_s, function(x) x[8])
rownames(rna_eq) = rna_samp
rna_eq = rna_eq[, 2:3]

pesti = sapply(rna_eq$condition, function (x) substr(x, start = 1, stop = nchar(x)-3))
sexo = sapply(rna_eq$condition, function (x) substr(x, start = nchar(x)-1, stop = nchar(x)-1))
tejido = sapply(rna_eq$condition, function (x) substr(x, start = nchar(x), stop = nchar(x)))

rna_eq = data.frame(rna_eq, "tissue" = tejido, "pesti" = pesti, "sex" = sexo)
id_rna = apply(rna_eq[,c("pesti", "sex", "tissue", "replicate_num")], 1, paste, collapse = "_")
rna_eq = data.frame(rna_eq, "id" = id_rna)

setwd(dir_output)
write.table(rna_eq, "imegen_equivalencies.txt", sep = "\t")

# Data preparation --------------------------------------------------------

library(NOISeq)
mirnaseq_all = as.data.frame(t(mirnaGenesWide))
mirna_all_names = sapply(rownames(mirnaseq_all), function (x) strsplit(x, split = "\\."))
mirna_all_names = sapply(mirna_all_names, function(x) x[2])
rownames(mirnaseq_all) = mirna_all_names

rnaseq_all = as.data.frame(t(rnaseqGenesWide))
rna_all_names = sapply(rownames(rnaseq_all), function (x) strsplit(x, split = "\\."))
rna_all_names = sapply(rna_all_names, function(x) x[2])
rownames(rnaseq_all) = rna_all_names

rnaseqTissue = lapply(c("_H", "_C"), function (x) rnaseq_all[grep(x, rownames(rnaseq_all)),])
names(rnaseqTissue) = c("Hippocampus", "Cerebellum")

mirnaseqTissue = lapply(c("_H", "_C"), function (x) mirnaseq_all[grep(x, rownames(mirnaseq_all)),])
names(mirnaseqTissue) = c("Hippocampus", "Cerebellum")

#Export data with no filter or arsynseq or norm
setwd(dir_data)
write.table(t(mirnaseq_all), "mirna_NonProcessed.txt", sep = "\t")
write.table(t(rnaseq_all), "rna_NonProcessed.txt", sep = "\t")
for (i in 1:2) {
  write.table(t(mirnaseqTissue[[i]]), paste(names(mirnaseqTissue)[i], "_mirna_NonProcessed.txt", sep =
"", sep = "\t")
  write.table(t(rnaseqTissue[[i]]), paste(names(mirnaseqTissue)[i], "_rna_NonProcessed.txt", sep = ""),
sep = "\t")
}
```

7.1.5. Attachment V: Transcriptomics initial formatting script

```
dir_trans = "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Transcriptomics"
dir_data = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data"
dir_miRNA = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/miRNA/"
dir_output = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data"

library(NOISeq)
library(plyr)


# Import data to format ------------------------------------------------------------

setwd(dir_data)

rnaseq_all = read.delim("rna_NonProcessed.txt", header = TRUE, as.is = TRUE)

rnaseqTissue = list("Hippocampus" = read.delim("Hippocampus_rna_NonProcessed.txt", header = TRUE, as.is
= TRUE),
                    "Cerebellum" = read.delim("Cerebellum_rna_NonProcessed.txt", header = TRUE, as.is =
TRUE))


# Import biomart -------------------------------------------------------------

biomart = read.delim("mart_export.txt")
colnames(biomart) = c("ENSEMBLEGeneID", "ENSEMBLTranscriptID", "RefSeqmRNAID",
                      "RefSeqPredictedmRNAID", "miRDBAccesionID")


# Import miRNA predictions ---------------------------------------------------

setwd(dir_miRNA)
mirna_predictions = read.delim("rat_predictions_80.txt")
mirna_predictions = mirna_predictions[which(mirna_predictions[,"score"] > 85),]


# Import miRBase -------------------------------------------------------------

setwd(dir_miRNA)
miRBase = read.delim("rno.gff3")
miRBase = miRBase[,9]
miRBase = sapply(as.character(miRBase), function (x) strsplit(as.character(x), split= "[;=]"))
miRBase = sapply(miRBase, function (x) x[c(4,6)])
miRBase = data.frame(t(miRBase))
colnames(miRBase) = c("Alias", "Name")
rownames(miRBase) = c()


# Change prediction table ----------------------------------------------------

# First, the miRNAs using miRBase

mirna_predictions = mirna_predictions[which(mirna_predictions[,"miRNA"] %in% miRBase[,"Name"]),]
mirna_predictions = data.frame("ID" = miRBase[mirna_predictions[,"miRNA"], "Alias"], mirna_predictions)
# 68952

# Second, the targets using Biomart

mirna_predictions3 = mirna_predictions[which(as.character(mirna_predictions[,"target"]) %in%
biomart[,3]),]
mirna_predictions4 = mirna_predictions[which(as.character(mirna_predictions[,"target"]) %in%
biomart[,4]),]

mirna_predictions3 = data.frame("Gene" = biomart[mirna_predictions3[,"target"], 1], mirna_predictions3)
mirna_predictions4 = data.frame("Gene" = biomart[mirna_predictions4[,"target"], 1], mirna_predictions4)

mirna_predictions = rbind.fill(mirna_predictions3, mirna_predictions4) # 41363
```

```
# Change RNA-seq IDs ------------------------------------------------------

# Using biomart

rnaseq_all = rnaseq_all[which(rownames(rnaseq_all) %in% biomart[,3]),] #Vamos de 18178 genes a 16619
biomart2 = biomart[which(biomart[,3] %in% rownames(rnaseq_all)),]
biomart2 = biomart2[!duplicated(biomart2[,3]),]
biomart2 = biomart2[order(biomart2[,3]),]
rnaseq_all = rnaseq_all[order(rownames(rnaseq_all)),]
rnaseq_all = data.frame("ENSEMBL" = as.character(biomart2[,1]), rnaseq_all)


# How many ENSEMBL IDs are unique?
ensembl = table(table(rnaseq_all[,"ENSEMBL"])) #Hay repeticiones, por tanto para ello sacamos mediana
rnaseq_all = aggregate(rnaseq_all[,2:ncol(rnaseq_all)], by = list("ENSEMBL" =
as.character(rnaseq_all[,"ENSEMBL"])), median)
rownames(rnaseq_all) = rnaseq_all[,"ENSEMBL"]
rnaseq_all = rnaseq_all[,-c(1)]


# Export same elements that were imported with the same name --------------

rnaseqTissue = lapply(c("_H", "_C"), function (x) rnaseq_all[,grep(x, colnames(rnaseq_all))])
names(rnaseqTissue) = c("Hippocampus", "Cerebellum")

setwd(dir_output)
write.table(rnaseq_all, "rna_NonProcessed.txt", sep = "\t")
for (i in 1:2) {
  write.table(rnaseqTissue[[i]], paste(names(rnaseqTissue)[i], "_rna_NonProcessed.txt", sep = ""), sep
= "\t")
}

# Export re-formatted miRNA predictions for later use ---------------------------------
setwd(dir_output)
write.table(mirna_predictions, "mirna_predictions.txt", sep = "\t")
```

7.1.6. Attachment VI: Pre-processing and exploration of transcriptomics script

```
dir_trans = "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Transcriptomics"
dir_data = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data"
dir_output = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_exp"

library(NOISeq)

# Import data
setwd(dir_data)

mirnaseq_all = read.delim("mirna_NonProcessed.txt", header = TRUE, as.is = TRUE)
rnaseq_all = read.delim("rna_NonProcessed.txt", header = TRUE, as.is = TRUE)
mirnaseqTissue = list("Hippocampus" = read.delim("Hippocampus_mirna_NonProcessed.txt", header = TRUE,
as.is = TRUE),
                      "Cerebellum" = read.delim("Cerebellum_mirna_NonProcessed.txt", header = TRUE,
as.is = TRUE))
rnaseqTissue = list("Hippocampus" = read.delim("Hippocampus_rna_NonProcessed.txt", header = TRUE, as.is
= TRUE),
                    "Cerebellum" = read.delim("Cerebellum_rna_NonProcessed.txt", header = TRUE, as.is =
TRUE))

rna_eq = read.delim("characteristics_transcriptomics.txt", header = TRUE, as.is = TRUE)


# First look at data ------------------------------------------------------

zero_mirna = apply(mirnaseq_all, 1, function(x) all(x == 0))
number_zero_mirna = sum(zero_mirna) #210 de 1133

zero_rna = apply(rnaseq_all, 1, function(x) all(x == 0))
number_zero_rna = sum(zero_rna) # 1657 de 15717


## Per tissue
zero_mirnaTissue = lapply(mirnaseqTissue, function (y) apply(y, 1, function(x) all(x==0)))
zero_rnaTissue = lapply(rnaseqTissue, function (y) apply(y, 1, function(x) all(x==0)))

names(zero_mirnaTissue) = c("Hippocampus", "Cerebellum")
names(zero_rnaTissue) = c("Hippocampus", "Cerebellum")

number_zero_mirnaTissue = lapply(zero_mirnaTissue, sum)
# HP: 240 de 1133
# CB: 241 de 1133
number_zero_rnaTissue = lapply(zero_rnaTissue, sum)
# HP: 1875 de 15717
# CB: 1931 de 15717

# Graphical representations (before arsynseq and filter and normalization) ----------------

setwd(dir_output)

# Demasiados datos para heatmap

miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]
pdf(file = "boxplots_noarsynseq_nofilter_nonorm.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2), mar = c(10,5,5,5))
for (i in 1:length(rnaseqTissue)) {
  sorted_mirna = mirnaseqTissue[[i]][,order(colnames(mirnaseqTissue[[i]]))]
  sorted_rna = rnaseqTissue[[i]][,order(colnames(rnaseqTissue[[i]]))]
  if (i == 1) {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7)])
  } else {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
```

```
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 7, 7, 7)])
  }
  boxplot(sorted_rna + 1, main = paste("RNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2, log =
"y",
          col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7, 7)])
}
dev.off()

# Filtros ----------------------------------------------------------------

#Quitamos todos los genes que tienen todo 0s y otras cosas con filtered.data de NOISeq

for (i in 1:2) {
  factor_rna = as.vector(sapply(colnames(rnaseqTissue[[i]]), function(x) substr(x, start = 1, stop =
3)))
  rnaseqTissue[[i]] = filtered.data(rnaseqTissue[[i]], factor_rna, norm = TRUE, method = 1, cv.cutoff =
500, cpm = 1)
  factor_mirna = as.vector(sapply(colnames(mirnaseqTissue[[i]]), function(x) substr(x, start = 1, stop
= 3)))
  mirnaseqTissue[[i]] = filtered.data(mirnaseqTissue[[i]], factor_mirna, norm = TRUE, method = 1,
cv.cutoff = 500, cpm = 1)
}


setwd(dir_data)
for (i in 1:2) {
  write.table(mirnaseqTissue[[i]], paste(names(mirnaseqTissue)[i], "_mirna_Filtered.txt", sep = ""),
sep = "\t")
  write.table(rnaseqTissue[[i]], paste(names(mirnaseqTissue)[i], "_rna_Filtered.txt", sep = ""), sep =
"\t")
}

# Graphical representations (before arsynseq, with filter, no norm) -----------------------

setwd(dir_output)

# Demasiados datos para heatmap

miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]
pdf(file = "boxplots_noarsynseq_filter_nonorm.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2), mar = c(10,5,5,5))
for (i in 1:length(rnaseqTissue)) {
  sorted_mirna = mirnaseqTissue[[i]][,order(colnames(mirnaseqTissue[[i]]))]
  sorted_rna = rnaseqTissue[[i]][,order(colnames(rnaseqTissue[[i]]))]
  if (i == 1) {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7)])
  } else {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 7, 7, 7)])
  }
  boxplot(sorted_rna + 1, main = paste("RNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2, log =
"y",
          col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7, 7)])
}
dev.off()

# Apply norm ----------------------------------------------------------------

library(limma)
for (i in 1:2) {
  mirnaseqTissue[[i]] = normalizeBetweenArrays(as.matrix(mirnaseqTissue[[i]]), method="quantile")
  rnaseqTissue[[i]] = normalizeBetweenArrays(as.matrix(rnaseqTissue[[i]]), method="quantile")
}


# Graphical representation (with filter and norm, no Arsynseq) -------------

miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]
```

```
pdf(file = "boxplots_noarsynseq_filter_norm.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2), mar = c(10,5,5,5))
for (i in 1:length(rnaseqTissue)) {
  sorted_mirna = mirnaseqTissue[[i]][,order(colnames(mirnaseqTissue[[i]]))]
  sorted_rna = rnaseqTissue[[i]][,order(colnames(rnaseqTissue[[i]]))]
  if (i == 1) {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7)])
  } else {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 7, 7, 7)])
  }
  boxplot(sorted_rna + 1, main = paste("RNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2, log =
"y",
          col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7, 7)])
}
dev.off()

# Arsynseq ------------------------------------------------------------------

rna_eq_a = data.frame(rna_eq[,c("tissue", "pesti", "sex")])
rna_eq[,"id"] = sapply(rna_eq[,"id"], as.character)
rownames(rna_eq_a) = rna_eq[,"id"]
rna_eq_a[,"TPS"] = sapply(rna_eq[,"id"], function (x) substr(x, start = 1, stop = nchar(x)-2))
mirna_eq = rna_eq_a[-c(nrow(rna_eq_a), nrow(rna_eq_a)-3),]


mirna_eq_Tissue = lapply(c("_H", "_C"), function (x) mirna_eq[grep(x, rownames(mirna_eq)),])
rna_eq_Tissue = lapply(c("_H", "_C"), function (x) rna_eq_a[grep(x, rownames(rna_eq_a)),])
names(rna_eq_Tissue) = c("H", "C")
names(mirna_eq_Tissue) = c("H", "C")


mirnaseqTissueNOISeq = list()
for (i in 1:length(mirnaseqTissue)) {
  mirnaseqTissueNOISeq[[i]] = readData(mirnaseqTissue[[i]], mirna_eq_Tissue[[i]])
}
mirnaseqTissueNoNoise = lapply(mirnaseqTissueNOISeq, function (x) ARSyNseq(x, factor = "TPS", norm =
"n", logtransf = FALSE))
names(mirnaseqTissueNoNoise) = names(mirnaseqTissue)
mirnaseqTissueNoNoise = lapply(mirnaseqTissueNoNoise, function(x) x@assayData$exprs)


rnaseqTissueNOISeq = list()
for (i in 1:length(rnaseqTissue)) {
  rnaseqTissueNOISeq[[i]] = readData(rnaseqTissue[[i]], rna_eq_Tissue[[i]])
}
rnaseqTissueNoNoise = lapply(rnaseqTissueNOISeq, function (x) ARSyNseq(x, factor = "TPS", norm = "n",
logtransf = FALSE))
names(rnaseqTissueNoNoise) = names(rnaseqTissue)
rnaseqTissueNoNoise = lapply(rnaseqTissueNoNoise, function(x) x@assayData$exprs)


setwd(dir_data)
for (i in 1:2) {
  write.table(mirnaseqTissue[[i]], paste(names(mirnaseqTissue)[i], "_mirna_NoNoiseData.txt", sep = ""),
sep = "\t")
  write.table(rnaseqTissue[[i]], paste(names(mirnaseqTissue)[i], "_rna_NoNoiseData.txt", sep = ""), sep
= "\t")
}

# Graphical representations (after arsynseq) ------------------------------------------------

setwd(dir_output)

miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]
pdf(file = "boxplots_arsynseq_filter_norm.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2), mar = c(10,5,5,5))
for (i in 1:length(rnaseqTissueNoNoise)) {
```

```
  sorted_mirna = mirnaseqTissueNoNoise[[i]][,order(colnames(mirnaseqTissueNoNoise[[i]]))]
  sorted_rna = rnaseqTissueNoNoise[[i]][,order(colnames(rnaseqTissueNoNoise[[i]]))]
  if (i == 1) {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7)])
  } else {
    boxplot(sorted_mirna + 1, main = paste("miRNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2,
log = "y",
            col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 7, 7, 7)])
  }
  boxplot(sorted_rna + 1, main = paste("RNA (", names(mirnaseqTissue)[i], ")", sep =""), las = 2, log =
"y",
          col = miscolores[c(10, 10, 10, 1, 1, 1, 11, 11, 11, 2, 2, 2, 9, 9, 9, 7, 7, 7)])
}
dev.off()


# PCA ---------------------------------------------------------------------

data2pca = list("miRNAseq (Hippocampus)" = log(t(mirnaseqTissueNoNoise[[1]])),
                "RNAseq (Hippocampus)" = log(t(rnaseqTissueNoNoise[[1]])),
                "miRNAseq (Cerebellum)" = log(t(mirnaseqTissueNoNoise[[2]])),
                "RNAseq (Cerebellum)" = log(t(rnaseqTissueNoNoise[[2]])))

pca.results = lapply(data2pca, PCA.GENES)


setwd(dir_output)
### Explained variance ###
samp <- c()
for (i in 1:length(data2pca)) {
  samp <- c(samp, length(rownames(data2pca[[i]])))
}
pdf(file = "explainedvariance_arsynseq.pdf", width = 3.5*2, height = 3.5*2)
par(mfcol = c(2,2))
for (i in 1:length(pca.results)) {
  barplot(pca.results[[i]]$var.exp[,1], names = 1:samp[i],
          xlab = "PC", ylab = "explained variance", ylim = c(0,0.7),
          main = names(pca.results)[i])
}
dev.off()

#### COLORS AND SHAPES ###
miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]
#Pesticides as colors
col.pest <- miscolores[1:3]
pesticides <- rna_eq[,"pesti"]
names(pesticides) <- rna_eq[,"id"]
names(col.pest) <- unique(pesticides)
mycol = col.pest[pesticides]

#Sex as shapes and tissues as filled-in shapes or empty shapes
myshapes = c(0, 15, 2, 17, 1, 16, 5, 18)
group_l <- list()
for (j in 1:length(data2pca)) {
  group <- c()
  for (i in 1:length(rownames(data2pca[[j]]))) {
    group <- c(group, paste(rna_eq[i,"sex"], rna_eq[i, "tissue"], sep = "-"))
  }
  group_l[[j]] <- group
}
mypch_l = list()
for (j in 1:length(data2pca)) {
  pch.group = myshapes[1:length(unique(group_l[[j]]))]
  names(pch.group) = unique(group_l[[j]])
  mypch = pch.group[group_l[[j]]]
  mypch_l[[j]] = mypch
}
```

```
### LOADINGS PLOT ###
pdf(file = "PCAloadings12_arsynseq.pdf", width = 3.5*3, height = 3.5*3)
par(mfcol = c(2,2))
for (i in 1:length(pca.results)) {
  plot(pca.results[[i]]$loadings[,1:2], col="white", cex = 0.5,
       xlab = paste("PCA 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0), "%", sep=""),
       ylab = paste("PCA 2 ", round(pca.results[[i]]$var.exp[2,1]*100,0), "%", sep=""),
       main = names(data2pca)[i],
       xlim = range(pca.results[[i]]$loadings[,1:2]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1),
       ylim = range(pca.results[[i]]$loadings[,1:2]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1))

  points(pca.results[[i]]$loadings[,1], pca.results[[i]]$loadings[,2], pch = 0)
}
dev.off()

pdf(file = "PCAloadings13_arsynseq.pdf", width = 3.5*3, height = 3.5*3)
par(mfcol = c(2,2))
for (i in 1:length(pca.results)) {
  plot(pca.results[[i]]$loadings[,1:3], col="white", cex = 0.5,
       xlab = paste("PCA 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0), "%", sep=""),
       ylab = paste("PCA 3 ", round(pca.results[[i]]$var.exp[2,1]*100,0), "%", sep=""),
       main = names(data2pca)[i],
       xlim = range(pca.results[[i]]$loadings[,1:3]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1),
       ylim = range(pca.results[[i]]$loadings[,1:3]) +
0.02*diff(range(pca.results[[i]]$loadings[,1:2]))*c(-1,1))
  points(pca.results[[i]]$loadings[,1], pca.results[[i]]$loadings[,2], pch = 0)
}
dev.off()


### PCA SCORES PLOT (WITH COLORS AND SHAPES) ###

pdf("PCAscores12_arsynseq.pdf", width = 3.5*3, height = 3.5*3)
par(mfcol = c(2,2))
for (i in 1:length(pca.results)) {
  rango = diff(range(pca.results[[i]]$scores[,1:2]))

  plot(pca.results[[i]]$scores[,1:2], col = "white",
       xlab = paste("PC 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0),
                    "%", sep = ""),
       ylab = paste("PC 2 ", round(pca.results[[i]]$var.exp[2,1]*100,0),
                    "%", sep = ""),
       main = names(data2pca)[i],
       xlim = range(pca.results[[i]]$scores[,1:2]) + 0.02*rango*c(-1,1),
       ylim = range(pca.results[[i]]$scores[,1:2]) + 0.02*rango*c(-1,1))

  points(pca.results[[i]]$scores[,1], pca.results[[i]]$scores[,2],
         pch = mypch_l[[i]], col = mycol, cex = 1.5)
  legend("topright", c("Endosulfan", "CYP+END", "Control"), col = col.pest, pch = 19, bty = "o", ncol =
2, box.col = "black")
  legend("right", c("M", "F"), pch = c(0, 15), bty = "o", ncol = 2)
}
dev.off()


pdf("PCAscores13_arsynseq.pdf", width = 3.5*3, height = 3.5*3)
par(mfcol = c(2,2))
for (i in 1:length(pca.results)) {
  rango2 = diff(range(pca.results[[i]]$scores[,c(1,3)]))
  plot(pca.results[[i]]$scores[,c(1,3)], col = "white",
       xlab = paste("PC 1 ", round(pca.results[[i]]$var.exp[1,1]*100,0),
                    "%", sep = ""),
       ylab = paste("PC 3 ", round(pca.results[[i]]$var.exp[3,1]*100,0),
                    "%", sep = ""),
       main = names(data2pca)[i],
       xlim = range(pca.results[[i]]$scores[,c(1,3)]) + 0.02*rango2*c(-1,1),
       ylim = range(pca.results[[i]]$scores[,c(1,3)]) + 0.02*rango2*c(-1,1))
```

```
  points(pca.results[[i]]$scores[,1], pca.results[[i]]$scores[,3],
         pch = mypch_l[[i]], col = mycol, cex = 1.5)
  legend("topright", c("Endosulfan", "CYP+END", "Control"), col = col.pest, pch = 19, bty = "o", ncol =
2, box.col = "black")
  legend("right", c("M", "F"), pch = c(0, 15), bty = "o", ncol = 2)
}
dev.off()
```

7.1.7. Attachment VII: miRDB bibliography check script

```
# Manual bibliography check ------------------------------------------------

#miRNA Bibliography check

setwd("~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/miRNA/")
predictions = read.delim("miRDB_v5.0_prediction_result.txt")

BACE1 = predictions[grep("NM_001207049", predictions[,"target"]),]
t_BACE1 = BACE1[grep("miR-339", BACE1[,"miRNA"]),]

NLRP3 = predictions[grep("NM_001243133", predictions[,"target"]),]
t_NLRP3 = NLRP3[grep("miR-223", NLRP3[,"miRNA"]),]

GRM4 = predictions[grep("NM_00125681", predictions[,"target"]),]
t_GRM4 = GRM4[grep("miR-1202", GRM4[,"miRNA"]),] #No matches for any splice variant

BDNF = predictions[grep("NM_001285422", predictions[,"target"]),]
t_BDNF = BDNF[grep("miR-1", BDNF[,"miRNA"]),]

RHOC = predictions[grep("NM_001106461", predictions[,"target"]),]
t_RHOC = RHOC[grep("miR-509", RHOC[,"miRNA"]),]

ACHE = predictions[grep("NM_001302621", predictions[,"target"]),]
t_ACHE = ACHE[grep("miR-608", RHOC[,"miRNA"]),] #No matches

mTOR = predictions[grep("NM_004958", predictions[,"target"]),]
t_mTOR = mTOR[grep("miR-199", mTOR[,"miRNA"]),]

APAF1 = predictions[grep("NM_013229", predictions[,"target"]),]
t_APAF1 = APAF1[grep("miR-23", APAF1[,"miRNA"]),]

# mirTarBase comparison ----------------------------------------------------

mirdb_rat = read.delim("mirna_predictions_80.txt")
mirtarbase = read.delim("mirTarBase_rat.txt")[,c(1:6)] #miRNA targets experimentally proven
biomart = read.delim("mart_export.txt")

#First of all, we add RefSeq IDs to the mirtarbase frame
names = sapply(mirtarbase[,5], function(x) as.character(biomart[grep(x, biomart[,4]),3]))
mirtarbase[,"RefSeq"] = sapply(names, function(x) max(x))
mirtarbase[,"ID"] = apply(mirtarbase[,c("miRNA","RefSeq")], 1, paste, collapse = "_")


#After that, we can compare it with the rat predictions and check scores
mirdb_rat[,"ID"] = apply(mirdb_rat[,c("miRNA","target")], 1, paste, collapse = "_")

rat_predictions = mirdb_rat[which(mirdb_rat[,"ID"] %in% mirtarbase[,"ID"]),]

write.table(rat_predictions, "mirna_mirtarbase_check.txt", sep = "\t", dec = ",")

# Analysis of miRDB --------------------------------------------------------

#Histograms
pdf("miRNAhistograms.pdf", width = 3.5*4, height = 3.5*2)
par(mfcol = c(1,2))
hist(table(mirdb_rat[,"miRNA"]), plot = TRUE,
     main = "Genes per miRNA",
     xlab = "Genes per miRNA",
     col = colors()[111],
     xlim = c(0,1700))
hist_gene = hist(table(mirdb_rat[,"target"]), plot = TRUE,
                 main = "miRNAs per gene",
                 xlab = "miRNAs per gene",
                 col = colors()[111],
                 xlim = c(0,150))
dev.off()
```

```
#Density plots
miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]

predictions_80 = mirdb_rat[which(mirdb_rat[,"score"] > 80),]
predictions_85 = mirdb_rat[which(mirdb_rat[,"score"] > 85),]
predictions_90 = mirdb_rat[which(mirdb_rat[,"score"] > 90),]
predictions_95 = mirdb_rat[which(mirdb_rat[,"score"] > 95),]

TT80_m = table(as.character(predictions_80[,"miRNA"]))
TT85_m = table(as.character(predictions_85[,"miRNA"]))
TT90_m = table(as.character(predictions_90[,"miRNA"]))
TT95_m = table(as.character(predictions_95[,"miRNA"]))

TT80_t = table(as.character(predictions_80[,"target"]))
TT85_t = table(as.character(predictions_85[,"target"]))
TT90_t = table(as.character(predictions_90[,"target"]))
TT95_t = table(as.character(predictions_95[,"target"]))

pdf("density_plots.pdf", width = 3.5*3, height = 3.5*1.5)
par(mfcol = c(1,2))
plot(density(TT80_m), lwd = 2, col = miscolores[1], main = "Density Plot", xlab = "Genes per miRNA",
     ylim = c(0, 0.03), xlim = c(0,600))
lines(density(TT85_m), lwd = 2, col = miscolores[2])
lines(density(TT90_m), lwd = 2, col = miscolores[3])
lines(density(TT95_m), lwd = 2, col = miscolores[4])

plot(density(TT80_t), lwd = 2, col = miscolores[1], main = "Density Plot", xlab = "miRNAs per gene",
     xlim = c(0,20), ylim = c(0,3))
lines(density(TT85_t), lwd = 2, col = miscolores[2])
lines(density(TT90_t), lwd = 2, col = miscolores[3])
lines(density(TT95_t), lwd = 2, col = miscolores[4])

legend("topright", c("80", "85", "90", "95"), col = miscolores[1:4], pch = 19,
       bty = "o", ncol = 2, box.col = "black")
dev.off()

# Median plot
x = c(80, 85, 90, 95)
y = c(median(predictions_80[,"score"]), median(predictions_85[,"score"]),
      median(predictions_90[,"score"]), median(predictions_95[,"score"]))
pdf("medians_score.pdf", width = 3.5*1.5, height = 3.5*1.5)
plot(x,y, pch = 19, main = "Median evolution with score cut-off")
dev.off()

# Summary table
breaks = c(80, 85, 90, 95)
medians_m = c(median(TT80_m), median(TT85_m), median(TT90_m), median(TT95_m))
medians_t = c(median(TT80_t), median(TT85_t), median(TT90_t), median(TT95_t))
miRNAs = c(length(unique(predictions_80[,"miRNA"])), length(unique(predictions_85[,"miRNA"])),
           length(unique(predictions_90[,"miRNA"])), length(unique(predictions_95[,"miRNA"])))
genes = c(length(unique(predictions_80[,"target"])), length(unique(predictions_85[,"target"])),
          length(unique(predictions_90[,"target"])), length(unique(predictions_95[,"target"])))

# We want to compare with our data, so we import it
setwd("~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data")
mirna_CB = read.delim("Cerebellum_mirna_NoNoiseData.txt")
rna_CB = read.delim("Cerebellum_rna_NoNoiseData.txt")
mirna_HP = read.delim("Hippocampus_mirna_NoNoiseData.txt")
rna_HP = read.delim("Hippocampus_rna_NoNoiseData.txt")


### POR AQUI VOY!

intersections_RNA_CB = c(nrow(predictions_80[which(unique(predictions_80[,"Gene"]) %in%
rownames(rna_CB)),]),
                         nrow(predictions_85[which(unique(predictions_85[,"Gene"]) %in%
rownames(rna_CB)),]),
                         nrow(predictions_90[which(unique(predictions_90[,"Gene"]) %in%
rownames(rna_CB)),]),
                         nrow(predictions_95[which(unique(predictions_95[,"Gene"]) %in%
rownames(rna_CB)),]))
```

```
intersections_RNA_HP = c(nrow(predictions_80[which(unique(predictions_80[,"Gene"]) %in%
rownames(rna_HP)),]),
                         nrow(predictions_85[which(unique(predictions_85[,"Gene"]) %in%
rownames(rna_HP)),]),
                         nrow(predictions_90[which(unique(predictions_90[,"Gene"]) %in%
rownames(rna_HP)),]),
                         nrow(predictions_95[which(unique(predictions_95[,"Gene"]) %in%
rownames(rna_HP)),]))

# We need to change miRNA names to MI format
setwd("~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/miRNA")
miRBase_rat = read.delim("miRDB_rat.txt")

miRBase_rat2 = read.delim("rno.gff3")
miRBase_rat2 = miRBase_rat2[,9]
miRBase_rat2 = sapply(as.character(miRBase_rat2), function (x) strsplit(as.character(x), split=
"[;=]"))
miRBase_rat2 = sapply(miRBase_rat2, function (x) x[c(4,6)])
miRBase_rat2_data = data.frame(t(miRBase_rat2))
colnames(miRBase_rat2_data) = c("Alias", "Name")

nice_names2 = miRBase_rat2_data[which(miRBase_rat2_data[,"Name"] %in% mirdb_rat[,"miRNA"]),]
nice_names2 = nice_names2[order(nice_names2[,2]),]
nice_names2 = nice_names2[!duplicated(nice_names2[,2]),] #752

mirna2 = mirdb_rat
mirna2 = mirna2[order(mirna2[,"miRNA"]),]
mirna2 = mirna2[which(mirna2[,"miRNA"] %in% nice_names2[,2]),]
mirna2 = mirna2[!duplicated(mirna2[,"miRNA"]),] #752 unique miRNAs

#Cambiamos nombres de mirna2 con equivalentes de nice_names2
rownames(mirna2) = nice_names2[,"Alias"]

#Hacemos nuevas tablas de predictions con mirna2
predictions_80_2 = mirna2[which(mirna2[,"score"] > 80),]
predictions_85_2 = mirna2[which(mirna2[,"score"] > 85),]
predictions_90_2 = mirna2[which(mirna2[,"score"] > 90),]
predictions_95_2 = mirna2[which(mirna2[,"score"] > 95),]

intersections_miRNA_CB = c(nrow(predictions_80[which(unique(predictions_80[,"ID"]) %in%
rownames(mirna_CB)),]),
                           nrow(predictions_85[which(unique(predictions_85[,"ID"]) %in%
rownames(mirna_CB)),]),
                           nrow(predictions_90[which(unique(predictions_90[,"ID"]) %in%
rownames(mirna_CB)),]),
                           nrow(predictions_95[which(unique(predictions_95[,"ID"]) %in%
rownames(mirna_CB)),]))
intersections_miRNA_HP = c(nrow(predictions_80[which(unique(predictions_80[,"ID"]) %in%
rownames(mirna_HP)),]),
                           nrow(predictions_85[which(unique(predictions_85[,"ID"]) %in%
rownames(mirna_HP)),]),
                           nrow(predictions_90[which(unique(predictions_90[,"ID"]) %in%
rownames(mirna_HP)),]),
                           nrow(predictions_95[which(unique(predictions_95[,"ID"]) %in%
rownames(mirna_HP)),]))

#Bringing it all together and creating the table
the_table = data.frame("Medians (genes per miRNA)" = medians_m,
                       "Medians (miRNAs per gene)" = medians_t,
                       "Num. miRNAs" = miRNAs,
                       "Num. genes" = genes,
                       "Intersect. miRNAs CB" = intersections_miRNA_CB,
                       "Intersect. miRNAs HP" = intersections_miRNA_HP,
                       "Intersect. Genes CB" = intersections_RNA_CB,
                       "Intersect. Genes HP" = intersections_RNA_HP)
rownames(the_table) = breaks
colnames(the_table) = c("Medians (genes per miRNA)", "Medians (miRNAs per gene)",
                        "Num. miRNAs", "Num. genes", "Intersect miRNAs CB",
                        "Intersect miRNAs HP", "Intersect Genes CB", "Intersect Genes HP")
```

7.1.8. Attachment VIII: Differential expression analysis for transcriptomics script

```
dir_trans = "/Users/Elena/Dropbox/Felipo_OmicsDENAMICdata/Transcriptomics"
dir_data = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data"
dir_output = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE"

library(plyr)

# Data import ------------------------------------------------------------

#Import data as rows for prots/metabolites and columns for samples
setwd(dir_data)
rna_HP = read.delim("Hippocampus_rna_NoNoiseData.txt", header = TRUE, as.is = TRUE, row.names = 1,
check.names = FALSE)
mirna_HP = read.delim("Hippocampus_mirna_NoNoiseData.txt", header = TRUE, as.is = TRUE, row.names = 1,
check.names = FALSE)
rna_CB = read.delim("Cerebellum_rna_NoNoiseData.txt", header = TRUE, as.is = TRUE, row.names = 1,
check.names = FALSE)
mirna_CB = read.delim("Cerebellum_mirna_NoNoiseData.txt", header = TRUE, as.is = TRUE, row.names = 1,
check.names = FALSE)


miRNA = list("CB" = mirna_CB, "HP" = mirna_HP)
RNA = list("CB" = rna_CB, "HP" = rna_HP)


# Only keep END and VH ---------------------------------------------------

miRNA = lapply(miRNA, function(x) x[,-grep("CYP_END", colnames(x))])
RNA = lapply(RNA, function(x) x[,-grep("CYP_END", colnames(x))])


# Model design -----------------------------------------------------------

charac = read.delim("characteristics_transcriptomics.txt", header = TRUE, as.is = TRUE, row.names = 1,
check.names = FALSE)
rna_eq = data.frame(charac, row.names = charac[,"id"])
mirna_eq = rna_eq[-c(3),]


sex_mirna_l = list()
pest_mirna_l = list()
sex_rna_l = list()
pest_rna_l = list()

for (i in 1:2) {
  sex_mirna_l[[i]] = factor(mirna_eq[colnames(miRNA[[i]]), "sex"])
  pest_mirna_l[[i]] = factor(mirna_eq[colnames(miRNA[[i]]), "pesti"], levels =
c("Control","Endosulfan"))
  sex_rna_l[[i]] = factor(rna_eq[colnames(RNA[[i]]), "sex"])
  pest_rna_l[[i]] = factor(rna_eq[colnames(RNA[[i]]), "pesti"], levels = c("Control","Endosulfan"))
}

mirna_matrix = list()
rna_matrix = list()
for (i in 1:2) {
  mirna_matrix[[i]] = model.matrix(~ sex_mirna_l[[i]] + pest_mirna_l[[i]] + sex_mirna_l[[i]] *
pest_mirna_l[[i]])
  rna_matrix[[i]] = model.matrix(~ sex_rna_l[[i]] + pest_rna_l[[i]] + sex_rna_l[[i]] * pest_rna_l[[i]])
}
names(mirna_matrix) = c("CB", "HP")
names(rna_matrix) = c("CB", "HP")

# Voom transformation ----------------------------------------------------
mirna_trans_l = list()
rna_trans_l = list()
for (i in 1:2) {
```

```
  mirna_trans_l[[i]] <- voom(miRNA[[i]], mirna_matrix[[i]],plot=TRUE)
  rna_trans_l[[i]] <- voom(RNA[[i]], rna_matrix[[i]],plot=TRUE)
}

# Limma pipeline ------------------------------------------------------------
fit_mirna = list()
fit_rna = list()
for (i in 1:2) {
  fit_mirna[[i]] = lmFit(mirna_trans_l[[i]], mirna_matrix[[i]])
  fit_mirna[[i]] = eBayes(fit_mirna[[i]])
  fit_rna[[i]] = lmFit(rna_trans_l[[i]], rna_matrix[[i]])
  fit_rna[[i]] = eBayes(fit_rna[[i]])
}

# Venn diagrams -------------------------------------------------------------

setwd(dir_output)
miscolores <- colors()[c(554, 89, 111, 512, 17, 586, 132, 428, 601, 568, 86, 390)]

###Adjusted p-values (BH)
for (i in 1:2) {
  results_mirna = decideTests(fit_mirna[[i]])[,c(3,4)]
  results_rna = decideTests(fit_rna[[i]])[,c(3,4)]

  pdf(paste(names(mirna_matrix)[i], "miRNA_Venn_adj.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_mirna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()

  pdf(paste(names(mirna_matrix)[i], "RNA_Venn_adj.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_rna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()
}

###P-values without adjusting
for (i in 1:2) {
  results_mirna = decideTests(fit_mirna[[i]], adjust.method = "none", p.value=0.01)[,c(3,4)]
  results_rna = decideTests(fit_rna[[i]], adjust.method = "none", p.value=0.01)[,c(3,4)]

  pdf(paste(names(mirna_matrix)[i], "miRNA_Venn.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_mirna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()

  pdf(paste(names(mirna_matrix)[i], "RNA_Venn.pdf", sep = "_"), width = 3.5*3, height = 3.5*3)
  vennDiagram(results_rna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)])
  dev.off()
}

pdf("all_Venn_0.05.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2))
for (i in 1:2) {
  results_mirna = decideTests(fit_mirna[[i]], adjust.method = "none", p.value=0.05)[,c(3,4)]
  results_rna = decideTests(fit_rna[[i]], adjust.method = "none", p.value=0.05)[,c(3,4)]

  vennDiagram(results_mirna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(mirna_matrix)[i], "(miRNA)", sep = " "), outer = FALSE)

  vennDiagram(results_rna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(mirna_matrix)[i], "(RNA)", sep = " "), outer = FALSE)
}
dev.off()

pdf("all_Venn_adj.pdf", width = 3.5*4, height = 3.5*4)
par(mfcol = c(2,2))
for (i in 1:2) {
  results_mirna = decideTests(fit_mirna[[i]])[,c(3,4)]
```

```
  results_rna = decideTests(fit_rna[[i]])[,c(3,4)]

  vennDiagram(results_mirna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(mirna_matrix)[i], "(miRNA)", sep = " "), outer = FALSE)

  vennDiagram(results_rna, names = c("Pesticide", "Sex∑Pesticide"),
              circle.col = miscolores[c(1,5,9,3)], mar = rep(0,4))
  title(main = paste(names(mirna_matrix)[i], "(RNA)", sep = " "), outer = FALSE)
}
dev.off()

# Create tables for future use ------------------------------------------


mirna_p_CB = list()
mirna_p_HP = list()
rna_p_CB = list()
rna_p_HP = list()

for (j in 1:4) {
  mirna_p_CB[[j]] = topTable(fit_mirna[[1]], coef = j, number = nrow(fit_mirna[[1]]))[,c("P.Value",
"adj.P.Val")]
  mirna_p_HP[[j]] = topTable(fit_mirna[[2]], coef = j, number = nrow(fit_mirna[[2]]))[,c("P.Value",
"adj.P.Val")]
  rna_p_CB[[j]] = topTable(fit_rna[[1]], coef = j, number = nrow(fit_rna[[1]]))[,c("P.Value",
"adj.P.Val")]
  rna_p_HP[[j]] = topTable(fit_rna[[2]], coef = j, number = nrow(fit_rna[[2]]))[,c("P.Value",
"adj.P.Val")]
}

names(mirna_p_CB) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")
names(mirna_p_HP) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")
names(rna_p_CB) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")
names(rna_p_HP) = c("Intercept", "Sex", "Pesti", "Pesti∑Sex")


setwd(dir_output)
for (i in 1:4) {
  write.table(mirna_p_CB[[i]], paste(names(mirna_p_CB)[i], "mirna_CB.txt", sep ="_"), sep = "\t")
  write.table(mirna_p_HP[[i]], paste(names(mirna_p_HP)[i], "mirna_HP.txt", sep ="_"), sep = "\t")
  write.table(rna_p_CB[[i]], paste(names(rna_p_CB)[i], "rna_CB.txt", sep ="_"), sep = "\t")
  write.table(rna_p_HP[[i]], paste(names(rna_p_HP)[i], "rna_HP.txt", sep ="_"), sep = "\t")
}

# Creation of average tables ------------------------------------------

mirna_averages = list()
rna_averages = list()

for (i in 1:2) { #1 is CB, 2 is HP
  VH_F_m = rowMeans(miRNA[[i]][,grep("Control_F", colnames(miRNA[[i]]))])
  END_F_m = rowMeans(miRNA[[i]][,grep("Endosulfan_F", colnames(miRNA[[i]]))])
  VH_M_m = rowMeans(miRNA[[i]][,grep("Control_M", colnames(miRNA[[i]]))])
  END_M_m = rowMeans(miRNA[[i]][,grep("Endosulfan_M", colnames(miRNA[[i]]))])
  mirna_averages[[i]] = data.frame(VH_F_m, END_F_m, VH_M_m, END_M_m)

  VH_F_r = rowMeans(RNA[[i]][,grep("Control_F", colnames(RNA[[i]]))])
  END_F_r = rowMeans(RNA[[i]][,grep("Endosulfan_F", colnames(RNA[[i]]))])
  VH_M_r = rowMeans(RNA[[i]][,grep("Control_M", colnames(RNA[[i]]))])
  END_M_r = rowMeans(RNA[[i]][,grep("Endosulfan_M", colnames(RNA[[i]]))])
  rna_averages[[i]] = data.frame(VH_F_r, END_F_r, VH_M_r, END_M_r)
}

colnames(mirna_averages[[1]]) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M")
colnames(mirna_averages[[2]]) = c("HP_VH_F", "HP_END_F", "HP_VH_M", "HP_END_M")
colnames(rna_averages[[1]]) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M")
colnames(rna_averages[[2]]) = c("HP_VH_F", "HP_END_F", "HP_VH_M", "HP_END_M")


#Export mean tables (individual tissues)
```

```
setwd(dir_output)
write.table(mirna_averages[[1]], "transcriptomics_mirna_averages_CB.txt", sep = "\t")
write.table(mirna_averages[[2]], "transcriptomics_mirna_averages_HP.txt", sep = "\t")
write.table(rna_averages[[1]], "transcriptomics_rna_averages_CB.txt", sep = "\t")
write.table(rna_averages[[2]], "transcriptomics_rna_averages_HP.txt", sep = "\t")

mirna_means = t(rbind.fill(as.data.frame(t(mirna_averages[[1]])),
as.data.frame(t(mirna_averages[[2]]))))
rna_means = t(rbind.fill(as.data.frame(t(rna_averages[[1]])), as.data.frame(t(rna_averages[[2]]))))
colnames(mirna_means) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M", "HP_VH_F", "HP_END_F",
"HP_VH_M", "HP_END_M")
colnames(rna_means) = c("CB_VH_F", "CB_END_F", "CB_VH_M", "CB_END_M", "HP_VH_F", "HP_END_F", "HP_VH_M",
"HP_END_M")

#Export mean tables (both tissues)
write.table(mirna_means, "transcriptomics_mirna_averages.txt", sep = "\t")
write.table(rna_means, "transcriptomics_rna_averages.txt", sep = "\t")


# Log2Ratio tables ----------------------------------------------------------

CB_mirna = transform(mirna_averages[[1]], log2_CB_F =
log2((mirna_averages[[1]][,2]+1)/(mirna_averages[[1]][,1]+1)),
                     log2_CB_M = log2((mirna_averages[[1]][,4]+1)/(mirna_averages[[1]][,3]+1)))[,c(5,6)]
HP_mirna = transform(mirna_averages[[2]], log2_CB_F =
log2((mirna_averages[[2]][,2]+1)/(mirna_averages[[2]][,1]+1)),
                     log2_CB_M = log2((mirna_averages[[2]][,4]+1)/(mirna_averages[[2]][,3]+1)))[,c(5,6)]

CB_rna = transform(rna_averages[[1]], log2_CB_F = log2(rna_averages[[1]][,2]/rna_averages[[1]][,1]),
                       log2_CB_M = log2(rna_averages[[1]][,4]/rna_averages[[1]][,3]))[,c(5,6)]
HP_rna = transform(rna_averages[[2]], log2_CB_F = log2(rna_averages[[2]][,2]/rna_averages[[2]][,1]),
                       log2_CB_M = log2(rna_averages[[2]][,4]/rna_averages[[2]][,3]))[,c(5,6)]


#Export for each individual tissue
write.table(CB_mirna, "transcriptomics_mirna_log2_CB.txt", sep = "\t")
write.table(HP_mirna, "transcriptomics_mirna_log2_HP.txt", sep = "\t")
write.table(CB_rna, "transcriptomics_rna_log2_CB.txt", sep = "\t")
write.table(HP_rna, "transcriptomics_rna_log2_HP.txt", sep = "\t")

mirna_log2 = t(rbind.fill(as.data.frame(t(CB_mirna)), as.data.frame(t(HP_mirna))))
rna_log2 = t(rbind.fill(as.data.frame(t(CB_rna)), as.data.frame(t(HP_rna))))
colnames(mirna_log2) = c("log2_CB_F", "log2_CB_M", "log2_HP_F", "log2_HP_M")
colnames(rna_log2) = c("log2_CB_F", "log2_CB_M", "log2_HP_F", "log2_HP_M")

write.table(mirna_log2, "transcriptomics_mirna_log2.txt", sep = "\t")
write.table(rna_log2, "transcriptomics_rna_log2.txt", sep = "\t")
```

7.1.9. Attachment IX: Clinical variable analysis script

```
dir_data = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/data/"
dir_output = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_clinical"
dir_data_met = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_DE/average_tables/"
dir_data_trans =
"~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE/average_tables/"
dir_sig = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/paintomics/output"
dir_pval_trans =
"~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE/pvalue_tables/"

library(corrplot)

# Import data ------------------------------------------------------------

setwd(dir_data)
data = read.delim("clinical01.txt", header = TRUE, check.names = FALSE, row.names = 1, dec = ",")
data = data[grep(" ST", rownames(data)),] # Solo nos interesa un tejido (las ratas son las mismas)

data_END = data[grep("END", rownames(data)),]
data_VH = data[grep("VH", rownames(data)),]
data_END_M = data_END[grep("M", rownames(data_END)),]
data_END_F = data_END[grep("F", rownames(data_END)),]
data_VH_M = data_VH[grep("M", rownames(data_VH)),]
data_VH_F = data_VH[grep("F", rownames(data_VH)),]

# Wilcoxon ---------------------------------------------------------------

MWM_ENDvsVH = wilcox.test(data_END[,1], data_VH[,1])
MWM_ENDvsVH_M = wilcox.test(data_END_M[,1], data_VH_M[,1])
MWM_ENDvsVH_F = wilcox.test(data_END_F[,1], data_VH_F[,1])

Rotarod_ENDvsVH = wilcox.test(data_END[,2], data_VH[,2])
Rotarod_ENDvsVH_M = wilcox.test(data_END_M[,2], data_VH_M[,2])
Rotarod_ENDvsVH_F = wilcox.test(data_END_F[,2], data_VH_F[,2])

Beam_ENDvsVH = wilcox.test(data_END[,3], data_VH[,3])
Beam_ENDvsVH_M = wilcox.test(data_END_M[,3], data_VH_M[,3])
Beam_ENDvsVH_F = wilcox.test(data_END_F[,3], data_VH_F[,3])

RM_ENDvsVH = wilcox.test(data_END[,4], data_VH[,4])
RM_ENDvsVH_M = wilcox.test(data_END_M[,4], data_VH_M[,4])
RM_ENDvsVH_F = wilcox.test(data_END_F[,4], data_VH_F[,4])

RMT_ENDvsVH = wilcox.test(data_END[,5], data_VH[,5])
RMT_ENDvsVH_M = wilcox.test(data_END_M[,5], data_VH_M[,5])
RMT_ENDvsVH_F = wilcox.test(data_END_F[,5], data_VH_F[,5])

# Correlations -----------------------------------------------------------

mean_END_F = apply(data_END_F, 2, mean, na.rm = TRUE)
mean_END_M = apply(data_END_M, 2, mean, na.rm = TRUE)
mean_VH_F = apply(data_VH_F, 2, mean, na.rm = TRUE)
mean_VH_M = apply(data_VH_M, 2, mean, na.rm = TRUE)
mean_CV = data.frame("VH_F" = mean_VH_F, "END_F" = mean_END_F, "VH_M" = mean_VH_M, "END_M" =
mean_END_M)

setwd(dir_data_met)
met_av_CB = read.delim("metabolomics_averages_CB.txt")
met_av_HP = read.delim("metabolomics_averages_HP.txt")
prot_av_CB = read.delim("proteomics_averages_CB.txt")
prot_av_HP = read.delim("proteomics_averages_HP.txt")

setwd(dir_data_trans)
rna_av_CB = read.delim("transcriptomics_rna_averages_CB.txt")
rna_av_HP = read.delim("transcriptomics_rna_averages_HP.txt")
mirna_av_CB = read.delim("transcriptomics_mirna_averages_CB.txt")
mirna_av_HP = read.delim("transcriptomics_mirna_averages_HP.txt")
```

```
setwd(dir_sig)
sig_rna_HP = as.vector(read.delim("significant_features_rna_HP.txt")[,1])
sig_rna_CB = as.vector(read.delim("significant_features_rna_CB.txt")[,1])
sig_metab_HP = as.vector(read.delim("significant_features_metab_HP.txt")[,1])
sig_metab_CB = as.vector(read.delim("significant_features_metab_CB.txt")[,1])
sig_proteom_CB = as.vector(read.delim("significant_features_prot_CB.txt")[,1])
sig_proteom_HP = as.vector(read.delim("significant_features_prot_HP.txt")[,1])

setwd(dir_pval_trans)
p_val_mirna_pesti = read.delim("Pesti_mirna.txt")
P_val_mirna_sp = read.delim("Pesti∑Sex_mirna.txt")
test1 = rownames(p_val_mirna_pesti[which(p_val_mirna_pesti[,1] < 0.05),])
test2 = rownames(p_val_mirna_pesti[which(p_val_mirna_pesti[,3] < 0.05),])

setwd(dir_output)
pdf("correlation_plots_num.pdf", width = 3.5*8, height = 3.5*2)
cor_m_CB = t(cor(t(met_av_CB[sig_metab_CB,]), t(mean_CV)))
cor_m_HP = t(cor(t(na.omit(met_av_HP[sig_metab_HP,])), t(mean_CV)))
cor_p_CB = t(cor(t(prot_av_CB[sig_proteom_CB,]), t(mean_CV)))
cor_p_HP = t(cor(t(na.omit(prot_av_HP[sig_proteom_HP,])), t(mean_CV)))
cor_r_CB = t(cor(t(rna_av_CB[sig_rna_CB,]), t(mean_CV)))
cor_r_HP = t(cor(t(na.omit(rna_av_HP[sig_rna_HP,])), t(mean_CV)))
cor_mir_CB = t(cor(t(mirna_av_CB[test1,]), t(mean_CV)))
cor_mir_HP = t(cor(t(mirna_av_CB[test2,]), t(mean_CV)))

corrplot(cor_m_CB, method="number")
corrplot(cor_m_HP, method="number")
corrplot(cor_p_CB, method="number")
corrplot(cor_p_HP, method="number")
corrplot(cor_r_CB, method="number")
corrplot(cor_r_HP, method="number")
corrplot(cor_mir_CB, method="number")
corrplot(cor_mir_HP, method="number")
dev.off()
```

7.1.10. Attachment X: Data formatting for Paintomics script

```
# Script to prepare data for Paintomics.
# Metabolites are detected just with their name. No need to change IDs.
# Proteins must have Uniprot ID. They already have them, so again, no need for change.
# Transcriptomics must have ENSEMBL Gene ID. They come with RefSeq IDs.

dir_data_log2_trans =
"~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE/log2_tables/"
dir_data_log2_set01 = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_DE/log2_tables/"
dir_data_p_trans =
"~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE/pvalue_tables/"
dir_data_p_set01 = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_DE/pvalue_tables/"
dir_data_set01 = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/data/"
dir_data_trans = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data"
dir_miRNA = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/miRNA/"
dir_output = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/paintomics/output"

library(plyr)

# Import miRNA targets ----------------------------------------------------

setwd(dir_miRNA)
mirna_predictions = read.delim("mirna_predictions.txt")

# Import all data we want to re-format ------------------------------------

# Log2 tables
setwd(dir_data_log2_trans)
mirna_log2 = read.delim("transcriptomics_mirna_log2.txt")
rna_log2 = read.delim("transcriptomics_rna_log2.txt")

setwd(dir_data_log2_set01)
metab_log2 = read.delim("metabolomics_log2.txt")
proteom_log2 = read.delim("proteomics_log2.txt")

# P-value tables
setwd(dir_data_p_trans)
mirna_p_pesti_CB = read.delim("Pesti_mirna_CB.txt")
mirna_p_pesti_HP = read.delim("Pesti_mirna_HP.txt")
rna_p_pesti_CB = read.delim("Pesti_rna_CB.txt")
rna_p_pesti_HP = read.delim("Pesti_rna_HP.txt")
mirna_p_sp_CB = read.delim("Pesti∑Sex_mirna_CB.txt")
mirna_p_sp_HP = read.delim("Pesti∑Sex_mirna_HP.txt")
rna_p_sp_CB = read.delim("Pesti∑Sex_rna_CB.txt")
rna_p_sp_HP = read.delim("Pesti∑Sex_rna_HP.txt")

setwd(dir_data_p_set01)
metab_p_pesti_CB = read.delim("Pesti_met_CB.txt")
metab_p_pesti_HP = read.delim("Pesti_met_HP.txt")
proteom_p_pesti_CB = read.delim("Pesti_prot_CB.txt")
proteom_p_pesti_HP = read.delim("Pesti_prot_HP.txt")
metab_p_sp_CB = read.delim("Pesti∑Sex_met_CB.txt")
metab_p_sp_HP = read.delim("Pesti∑Sex_met_HP.txt")
proteom_p_sp_CB = read.delim("Pesti∑Sex_prot_CB.txt")
proteom_p_sp_HP = read.delim("Pesti∑Sex_prot_HP.txt")

# Bind p-values for CB and HP
mirna_p_pesti = as.data.frame(t(rbind.fill(as.data.frame(t(mirna_p_pesti_CB)),
as.data.frame(t(mirna_p_pesti_HP)))))
colnames(mirna_p_pesti) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")
rna_p_pesti = as.data.frame(t(rbind.fill(as.data.frame(t(rna_p_pesti_CB)),
as.data.frame(t(rna_p_pesti_HP)))))
colnames(rna_p_pesti) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")

mirna_p_sp = as.data.frame(t(rbind.fill(as.data.frame(t(mirna_p_sp_CB)),
as.data.frame(t(mirna_p_sp_HP)))))
colnames(mirna_p_sp) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")
rna_p_sp = as.data.frame(t(rbind.fill(as.data.frame(t(rna_p_sp_CB)), as.data.frame(t(rna_p_sp_HP)))))
```

```
colnames(rna_p_sp) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")

metab_p_pesti = as.data.frame(t(rbind.fill(as.data.frame(t(metab_p_pesti_CB)),
as.data.frame(t(metab_p_pesti_HP)))))
colnames(metab_p_pesti) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")
proteom_p_pesti = as.data.frame(t(rbind.fill(as.data.frame(t(proteom_p_pesti_CB)),
as.data.frame(t(proteom_p_pesti_HP)))))
colnames(proteom_p_pesti) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")

metab_p_sp = as.data.frame(t(rbind.fill(as.data.frame(t(metab_p_sp_CB)),
as.data.frame(t(metab_p_sp_HP)))))
colnames(metab_p_sp) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")
proteom_p_sp = as.data.frame(t(rbind.fill(as.data.frame(t(proteom_p_sp_CB)),
as.data.frame(t(proteom_p_sp_HP)))))
colnames(metab_p_sp) = c("P.val_CB", "Adj.P.val_CB", "P.val_HP", "Adj.P.val_HP")

# Export for future use
setwd(dir_data_p_set01)
write.table(metab_p_pesti, "Pesti_met.txt", sep = "\t")
write.table(proteom_p_pesti, "Pesti_prot.txt", sep = "\t")
write.table(metab_p_sp, "Pesti∑Sex_met.txt", sep = "\t")
write.table(proteom_p_sp, "Pesti∑Sex_prot.txt", sep = "\t")

setwd(dir_data_p_trans)
write.table(mirna_p_pesti, "Pesti_mirna.txt", sep = "\t")
write.table(rna_p_pesti, "Pesti_rna.txt", sep = "\t")
write.table(mirna_p_sp, "Pesti∑Sex_mirna.txt", sep = "\t")
write.table(rna_p_sp, "Pesti∑Sex_rna.txt", sep = "\t")


# Import non-processed data ------------------------------------------------

setwd(dir_data_trans)
rna_nonp = read.delim("rna_NonProcessed.txt")
mirna_nonp = read.delim("mirna_NonProcessed.txt")

setwd(dir_data_set01)
metab_nonp = as.data.frame(t(read.delim("metabolomics01.txt", header = TRUE, as.is = TRUE, row.names =
1,
             check.names = FALSE, dec = ",")))
proteom_nonp = as.data.frame(t(read.delim("proteomics01.txt", header = TRUE, as.is = TRUE, row.names =
1,
               check.names = FALSE, dec = ",")))


# Prepare log2 RNA-seq/metab/proteom table ---------------------------------------------

# Replace NAs with zeroes
rna_log2[is.na(rna_log2)] = 0
metab_log2[is.na(metab_log2)] = 0
proteom_log2[is.na(proteom_log2)] = 0

# We import original data and add those filtered genes/metabolites/prots to tables
elim_rna = setdiff(rownames(rna_nonp), rownames(rna_log2))
rna = data.frame(rna_log2)
rna[elim_rna,] = rep(0, 4)

elim_metab = setdiff(rownames(metab_nonp), rownames(metab_log2))
metab = data.frame(metab_log2)
metab[elim_metab,] = rep(0, 4)

elim_proteom = setdiff(rownames(proteom_nonp), rownames(proteom_log2))
proteom = data.frame(proteom_log2)
proteom[elim_proteom,] = rep(0, 4)

setwd(dir_output)
write.table(rna, "rnaseqlog2.txt", sep = "\t", quote = FALSE)
write.table(metab, "metabolomicslog2.txt", sep = "\t", quote = FALSE)
write.table(proteom, "proteomicslog2.txt", sep = "\t", quote = FALSE)
```

```
# Prepare log2 miRNA-seq table ----------------------------------------------------

list_target_frames = list()
for (i in 1:nrow(mirna_log2)) {
  Genes = as.character(mirna_predictions[which(mirna_predictions["ID"] == rownames(mirna_log2)[i]),
"Gene"])
  log2_CB_F = rep(mirna_log2[i, 1], length(Genes))
  log2_CB_M = rep(mirna_log2[i, 2], length(Genes))
  log2_HP_F = rep(mirna_log2[i, 3], length(Genes))
  log2_HP_M = rep(mirna_log2[i, 4], length(Genes))
  list_target_frames[[i]] = data.frame("log2_CB_F" = log2_CB_F,
                                       "log2_CB_M" = log2_CB_M,
                                       "log2_HP_F" = log2_HP_F,
                                       "log2_HP_M" = log2_HP_M)
  list_target_frames[[i]] = data.frame("Genes" = Genes , list_target_frames[[i]])
}
names(list_target_frames) = rownames(mirna_log2)
target_log2 = rbind.fill(list_target_frames)
target_log2 = aggregate(target_log2[,c(2:5)], by = list("Genes" = target_log2[,"Genes"]), sum)
rownames(target_log2) = target_log2[,1]
target_log2 = target_log2[,-c(1)]

# Now we add 0s where there's NAs

target_log2[is.na(target_log2)] = 0

# Export

setwd(dir_output)
write.table(target_log2, "mirnaseq_targets_log2.txt", sep = "\t", quote = FALSE)


# Prepare significant features from p-value tables for RNAseq/metab/prot and for each tissue ----------

rna_sig_CB_pesti = rna_p_pesti[which(rna_p_pesti[,1] < 0.05),]
rna_sig_HP_pesti = rna_p_pesti[which(rna_p_pesti[,3] < 0.05),]
rna_sig_CB_sp = rna_p_sp[which(rna_p_pesti[,1] < 0.05),]
rna_sig_HP_sp = rna_p_sp[which(rna_p_pesti[,3] < 0.05),]

rna_sig_CB = unique(c(rownames(rna_sig_CB_pesti), rownames(rna_sig_CB_sp)))
rna_sig_HP = unique(c(rownames(rna_sig_HP_pesti), rownames(rna_sig_HP_sp)))

metab_sig_CB_pesti = metab_p_pesti[which(metab_p_pesti[,1] < 0.05),]
metab_sig_HP_pesti = metab_p_pesti[which(metab_p_pesti[,3] < 0.05),]
metab_sig_CB_sp = metab_p_sp[which(metab_p_pesti[,1] < 0.05),]
metab_sig_HP_sp = metab_p_sp[which(metab_p_pesti[,3] < 0.05),]

metab_sig_CB = unique(c(rownames(metab_sig_CB_pesti), rownames(metab_sig_CB_sp)))
metab_sig_HP = unique(c(rownames(metab_sig_HP_pesti), rownames(metab_sig_HP_sp)))

proteom_sig_CB_pesti = proteom_p_pesti[which(proteom_p_pesti[,1] < 0.05),]
proteom_sig_HP_pesti = proteom_p_pesti[which(proteom_p_pesti[,3] < 0.05),]
proteom_sig_CB_sp = proteom_p_sp[which(proteom_p_pesti[,1] < 0.05),]
proteom_sig_HP_sp = proteom_p_sp[which(proteom_p_pesti[,3] < 0.05),]

proteom_sig_CB = unique(c(rownames(proteom_sig_CB_pesti), rownames(proteom_sig_CB_sp)))
proteom_sig_HP = unique(c(rownames(proteom_sig_HP_pesti), rownames(proteom_sig_HP_sp)))

setwd(dir_output)
write.table(rna_sig_CB, "significant_features_rna_CB.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
write.table(rna_sig_HP, "significant_features_rna_HP.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
write.table(metab_sig_CB, "significant_features_metab_CB.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
write.table(metab_sig_HP, "significant_features_metab_HP.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
write.table(proteom_sig_CB, "significant_features_prot_CB.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
write.table(proteom_sig_HP, "significant_features_prot_HP.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
```

```
# Significant features for miRNAseq -------------------------------------

# Pesticide
list_target_frames_pesti = list()
for (i in 1:nrow(mirna_p_pesti)) {
  Genes = as.character(mirna_predictions[which(mirna_predictions["ID"] == rownames(mirna_p_pesti)[i]),
"Gene"])
  Pval_CB = rep(mirna_p_pesti[i, 1], length(Genes))
  Pval_HP = rep(mirna_p_pesti[i, 3], length(Genes))
  list_target_frames_pesti[[i]] = data.frame("Pval_CB" = Pval_CB, "Pval_HP" = Pval_HP)
  list_target_frames_pesti[[i]] = data.frame("Genes" = Genes , list_target_frames_pesti[[i]])
}
names(list_target_frames_pesti) = rownames(mirna_p_pesti)
target_p = rbind.fill(list_target_frames_pesti)
target_p[is.na(target_p)] = 0
target_p = aggregate(target_p[,c(2:3)], by = list("Genes" = target_p[,"Genes"]), sum)
rownames(target_p) = target_p[,1]
target_p = target_p[,-c(1)]

mirna_sig_CB_Pesti = target_p[which(target_p[,1] < 0.05),]
mirna_sig_HP_Pesti = target_p[which(target_p[,2] < 0.05),]

# Pesticide ∑ Sex
list_target_frames_sp = list()
for (i in 1:nrow(mirna_p_sp)) {
  Genes = as.character(mirna_predictions[which(mirna_predictions["ID"] == rownames(mirna_p_sp)[i]),
"Gene"])
  Pval_CB = rep(mirna_p_sp[i, 1], length(Genes))
  Pval_HP = rep(mirna_p_sp[i, 3], length(Genes))
  list_target_frames_sp[[i]] = data.frame("Pval_CB" = Pval_CB, "Pval_HP" = Pval_HP)
  list_target_frames_sp[[i]] = data.frame("Genes" = Genes , list_target_frames_sp[[i]])
}
names(list_target_frames_sp) = rownames(mirna_p_sp)
target_sp = rbind.fill(list_target_frames_sp)
target_sp[is.na(target_sp)] = 0
target_sp = aggregate(target_sp[,c(2:3)], by = list("Genes" = target_sp[,"Genes"]), sum)
rownames(target_sp) = target_sp[,1]
target_sp = target_sp[,-c(1)]

mirna_sig_CB_sp = target_sp[which(target_sp[,1] < 0.05),]
mirna_sig_HP_sp = target_sp[which(target_sp[,2] < 0.05),]

mirna_sig_CB = unique(c(rownames(mirna_sig_CB_Pesti), rownames(mirna_sig_CB_sp)))
mirna_sig_HP = unique(c(rownames(mirna_sig_HP_Pesti), rownames(mirna_sig_HP_sp)))


setwd(dir_output)
write.table(mirna_sig_CB, "significant_features_mirna_CB.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
write.table(mirna_sig_HP, "significant_features_mirna_HP.txt", sep = "\n", quote = FALSE, row.names =
FALSE, col.names = FALSE)
```

7.1.11. Attachment XI: Expression profile script

```
dir_sig = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/paintomics/output"
dir_averages_01 = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_DE/average_tables/"
dir_averages_trans =
"~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE/average_tables/"
dir_output_01 = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/set01/output_DE/expression_profiles/"
dir_output_trans =
"~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE/expression_profiles/"
dir_pval_trans =
"~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/output_DE/pvalue_tables/"
dir_data_trans = "~/Dropbox/Felipo_OmicsDENAMICdata/Integration/trancriptomics/data/"

# Import data ------------------------------------------------------------

setwd(dir_sig)
sig_rna_HP = as.vector(read.delim("significant_features_rna_HP.txt")[,1])
sig_rna_CB = as.vector(read.delim("significant_features_rna_CB.txt")[,1])
sig_metab_HP = as.vector(read.delim("significant_features_metab_HP.txt")[,1])
sig_metab_CB = as.vector(read.delim("significant_features_metab_CB.txt")[,1])
sig_proteom_CB = as.vector(read.delim("significant_features_prot_CB.txt")[,1])
sig_proteom_HP = as.vector(read.delim("significant_features_prot_HP.txt")[,1])

setwd(dir_pval_trans)
p_val_mirna_pesti = read.delim("Pesti_mirna.txt")
P_val_mirna_sp = read.delim("Pesti∑Sex_mirna.txt")

setwd(dir_averages_01)
metab_av = read.delim("metabolomics_averages.txt")
proteom_av = read.delim("proteomics_averages.txt")

setwd(dir_averages_trans)
rna_av = read.delim("transcriptomics_rna_averages.txt")
mirna_av = read.delim("transcriptomics_mirna_averages.txt")

# Create expression profiles  ---------------------------------------------

expression_profile <- function(gene, matrix, type) {
  ylabs = c("Gene expression", "miRNA expression", "Metabolite amount", "Protein amount")
  plot(x = c(1,2,4,7), rnorm(4), col = "white", xlab = "Female | Male", ylab = ylabs[type],
       main = gene, xaxt = "n", ylim = c(min(matrix[gene,]),max(matrix[gene,])))
  axis(side = 1, at = c(2,3,5,6), labels = rep(c("VH", "END"),2))
  abline(v = 4, col = "grey", lty = 2)
  # lines(c(1,2), c(media vehiculo, media endosulfan))
  lines(c(2:3), c(matrix[gene,1], matrix[gene,2]), type = "b", lwd = 2, col = "coral1", pch = 19) #CB F
  lines(c(5:6), c(matrix[gene,3], matrix[gene,4]), type = "b", lwd = 2, col = "coral1", pch = 19) #CB M
  lines(c(2:3), c(matrix[gene,5], matrix[gene,6]), type = "b", lwd = 2, col = "darkcyan") #HP F
  lines(c(5:6), c(matrix[gene,7], matrix[gene,8]), type = "b", lwd = 2, col = "darkcyan") #HP M
}

# RNA
setwd(dir_output_trans)
rna_matrix = na.omit(rna_av[unique(c(sig_rna_CB, sig_rna_HP)),])
rna_type = 1
pdf("RNA_expression_profile.pdf", width = 3.5*4, height = 3.5*5)
par(mfcol = c(5, 4))
for (i in 1:nrow(rna_matrix)) {
  gene = rownames(rna_matrix)[i]
  expression_profile(gene, rna_matrix, rna_type)
}
dev.off()


# Metab
setwd(dir_output_01)
metab_matrix = na.omit(metab_av[unique(c(sig_metab_CB, sig_metab_HP)),])
metab_type = 3
pdf("metab_expression_profile.pdf", width = 3.5*4, height = 3.5*5)
par(mfcol = c(5, 4))
```

```
for (i in 1:nrow(metab_matrix)) {
  metab = rownames(metab_matrix)[i]
  expression_profile(metab, metab_matrix, metab_type)
}
dev.off()

# Proteom
setwd(dir_output_01)
proteom_matrix = na.omit(proteom_av[unique(c(sig_proteom_CB, sig_proteom_HP)),])
proteom_type = 4
pdf("proteom_expression_profile.pdf", width = 3.5*4, height = 3.5*5)
par(mfcol = c(5, 4))
for (i in 1:nrow(proteom_matrix)) {
  prot = rownames(proteom_matrix)[i]
  expression_profile(prot, proteom_matrix, proteom_type)
}
dev.off()

# miRNA
setwd(dir_output_trans)
test1 = rownames(p_val_mirna_pesti[which(p_val_mirna_pesti[,1] < 0.05),])
test2 = rownames(p_val_mirna_pesti[which(p_val_mirna_pesti[,3] < 0.05),])
sig_mirna = unique(c(test1, test2))
mirna_matrix = na.omit(mirna_av[sig_mirna,])
mirna_type = 2
pdf("mirna_expression_profile.pdf", width = 3.5*4, height = 3.5*5)
par(mfcol = c(5, 4))
for (i in 1:nrow(mirna_matrix)) {
  mirna = rownames(mirna_matrix)[i]
  expression_profile(mirna, mirna_matrix, mirna_type)
}
dev.off()


# Heatmaps for trans ----------------------------------------------------------

setwd(dir_data_trans)
rna_data_CB = read.delim("Cerebellum_rna_NoNoiseData.txt")
rna_data_HP = read.delim("Hippocampus_rna_NoNoiseData.txt")
rna_DE_data_CB = rna_data_CB[sig_rna_CB,]
rna_DE_data_HP = na.omit(rna_data_HP[sig_rna_HP,])

mirna_data_CB = read.delim("Cerebellum_mirna_NoNoiseData.txt")
mirna_data_HP = read.delim("Hippocampus_mirna_NoNoiseData.txt")
mirna_DE_data_CB = mirna_data_CB[test1,]
mirna_DE_data_HP = mirna_data_HP[test2,]

setwd(dir_output_trans)
pdf("heatmaps_DE.pdf", width = 3.5*3, height = 3.5*2)
heatmap(as.matrix(rna_DE_data_CB), main = "Differentially expressed RNAseq CB", margins = c(10,5))
heatmap(as.matrix(rna_DE_data_HP), main = "Differentially expressed RNAseq HP", margins = c(10,5))
heatmap(as.matrix(mirna_DE_data_CB), main = "Differentially expressed miRNAseq CB", margins = c(10,5))
heatmap(as.matrix(mirna_DE_data_HP), main = "Differentially expressed miRNAseq HP", margins = c(10,5))
dev.off()
```

7.2. Other supplementary material

7.2.1. Attachment XII: Set 03 and Set 10 details

Set 03

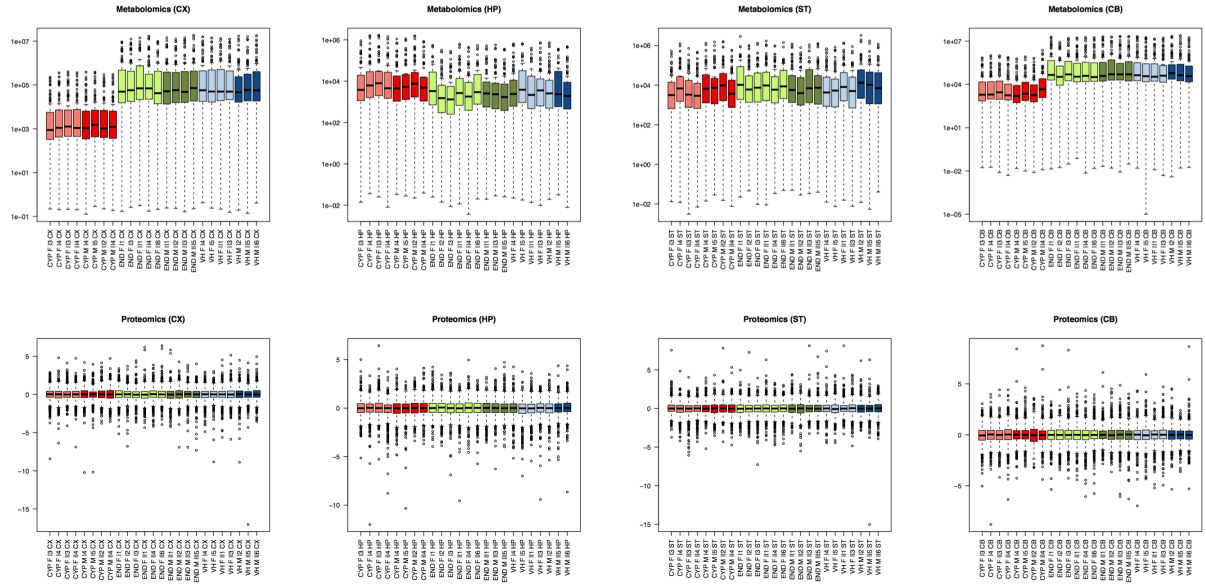
Treatment	Tissues: CB		HP		CX		ST		TOTAL / TISSUE
	F	M	F	M	F	M	F	M	
VH	3	2	3	2	3	2	3	2	5
CYP	1	0	1	0	1	0	1	0	1
CHLOR 0.1	4	1	4	1	4	1	4	1	5
CHLOR 0.3	4	4	4	4	4	4	4	4	8
CHLOR 1	2	2	2	2	2	2	2	2	4
CAR	3	2	3	2	3	2	3	2	5
							TOTAL RATS:		28

Set 10

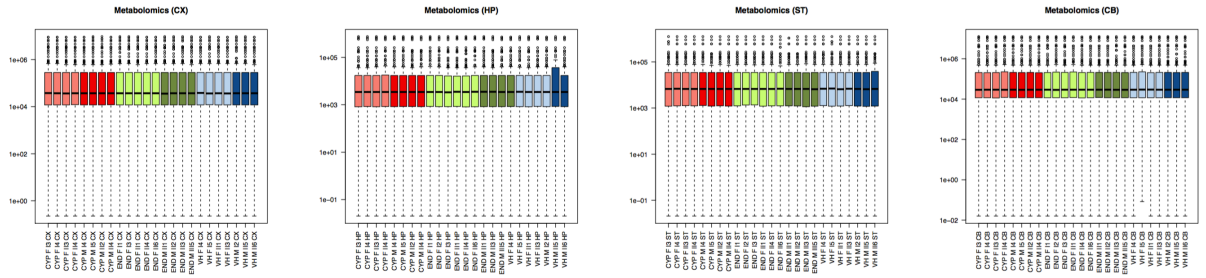
Treatment	Tissues: CB		HP		CX		ST		TOTAL / TISSUE
	F	M	F	M	F	M	F	M	
VH	3	3	3	3	3	3	3	3	6
CYP	2	2	2	2	2	2	2	2	4
CPF	3	3	3	3	3	3	3	3	6
CYP+END	3	3	3	3	3	3	3	3	6
END	3	3	3	3	3	3	3	3	6
							TOTAL RATS:		28

7.2.2. Attachment XIII: Set 01 Boxplots

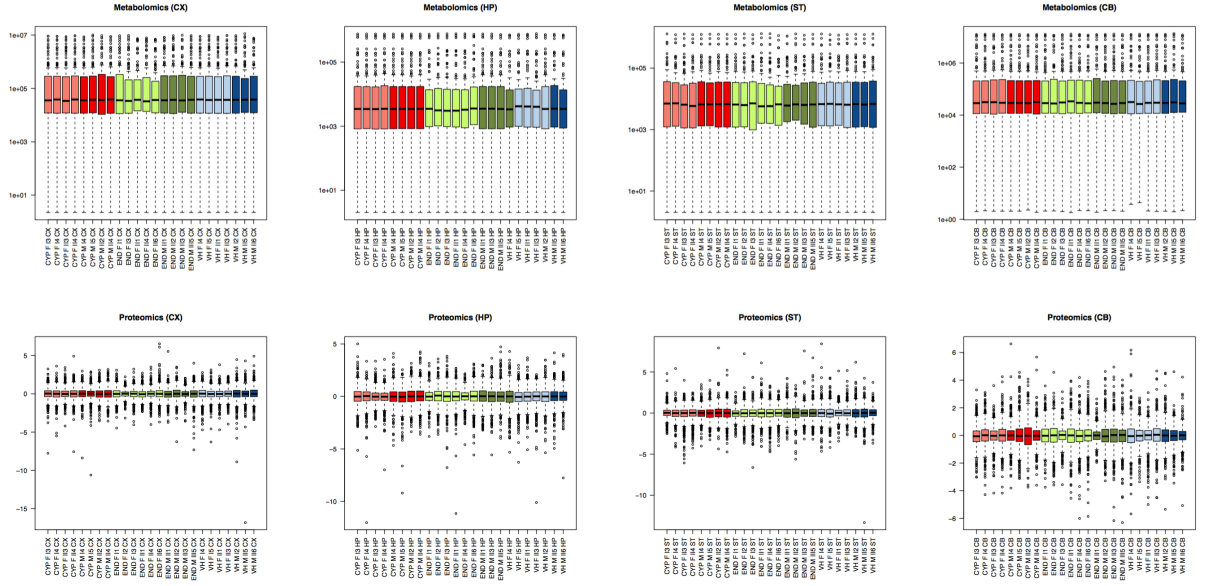
Before any transformation:



After normalization of metabolomics data:

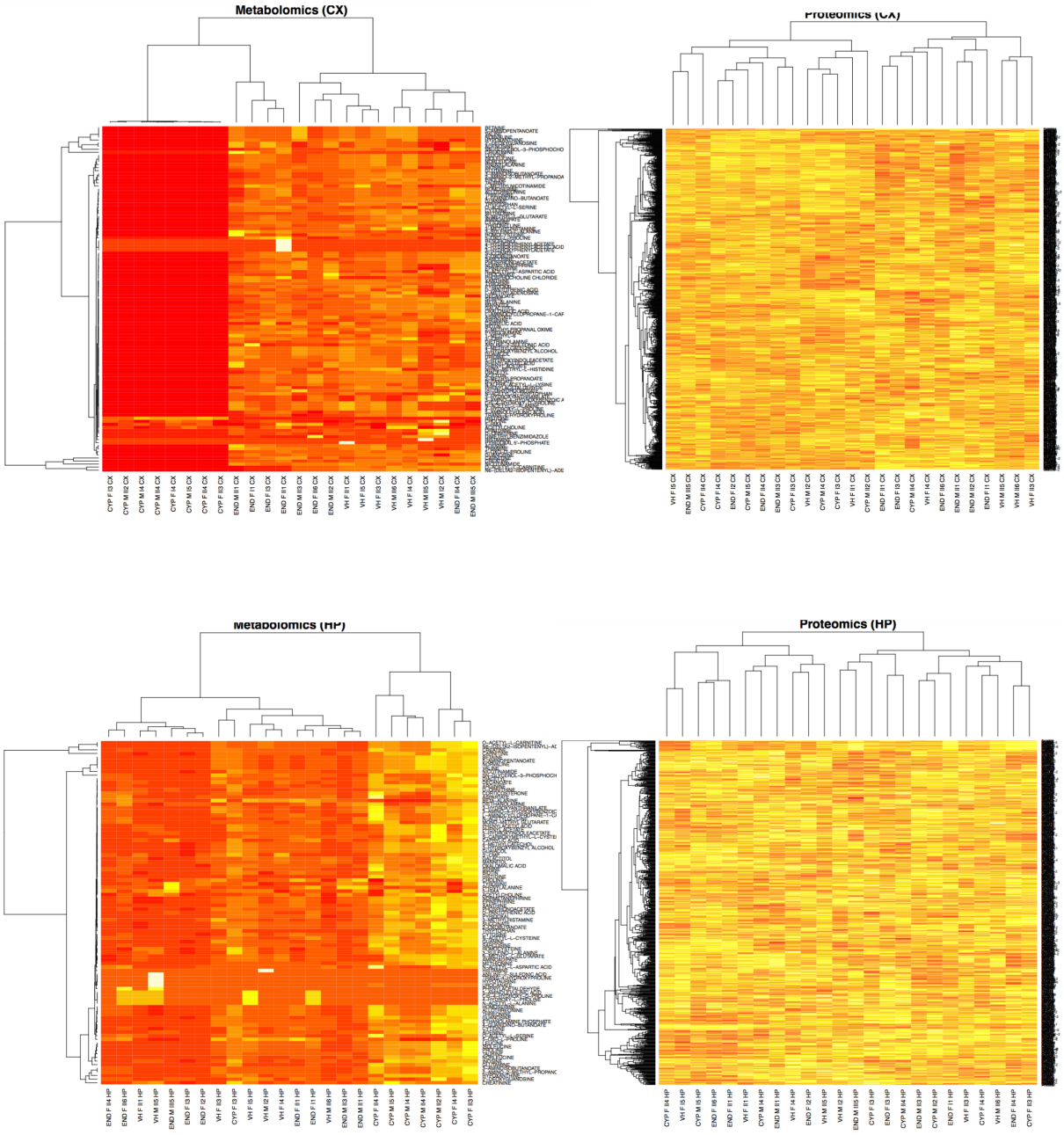


After normalization and arsynseq on metabolomics data and arsynseq on proteomics:

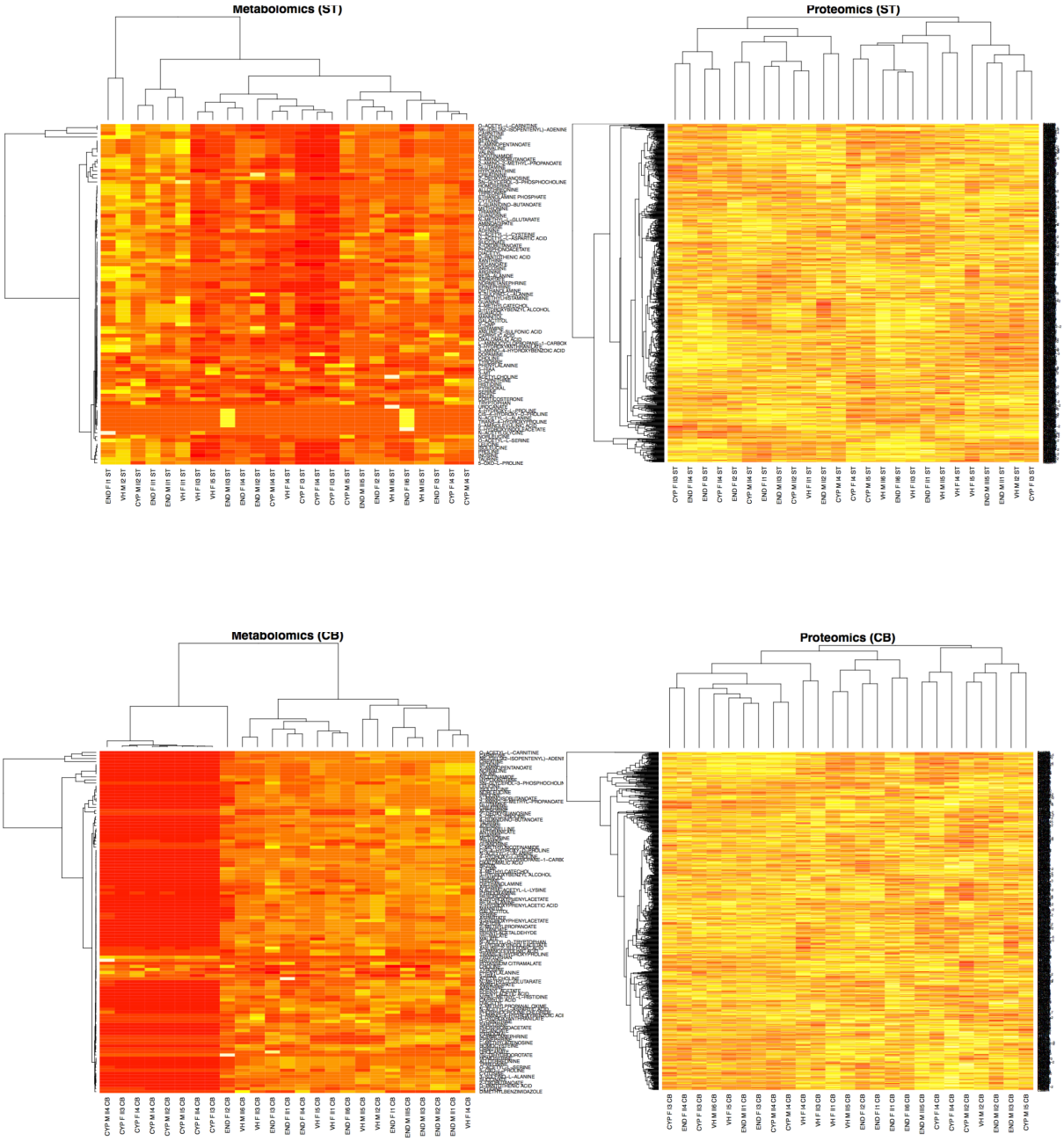


7.2.3. Attachment XIV: Set 01 Heatmaps

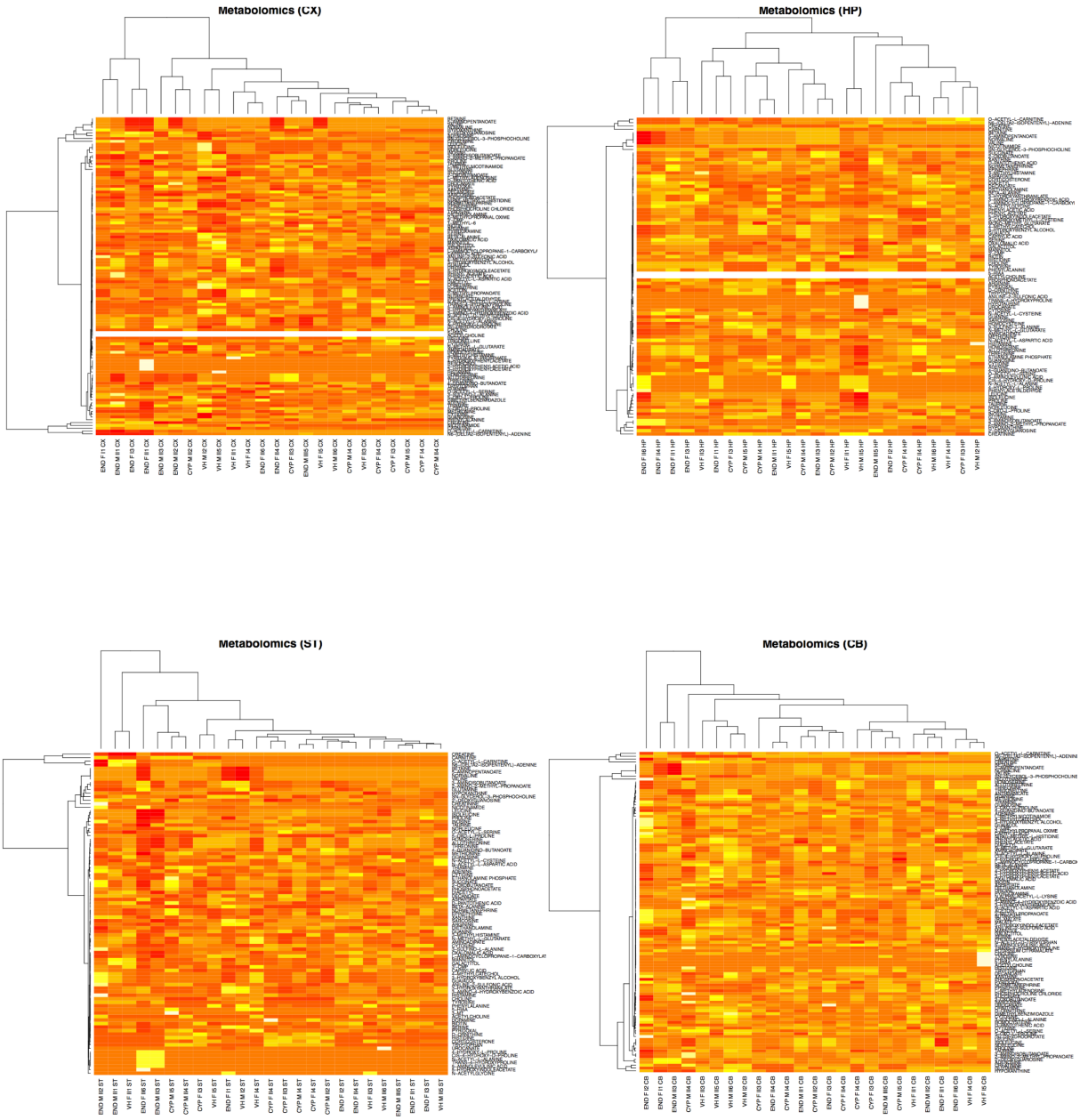
Before pre-processing:



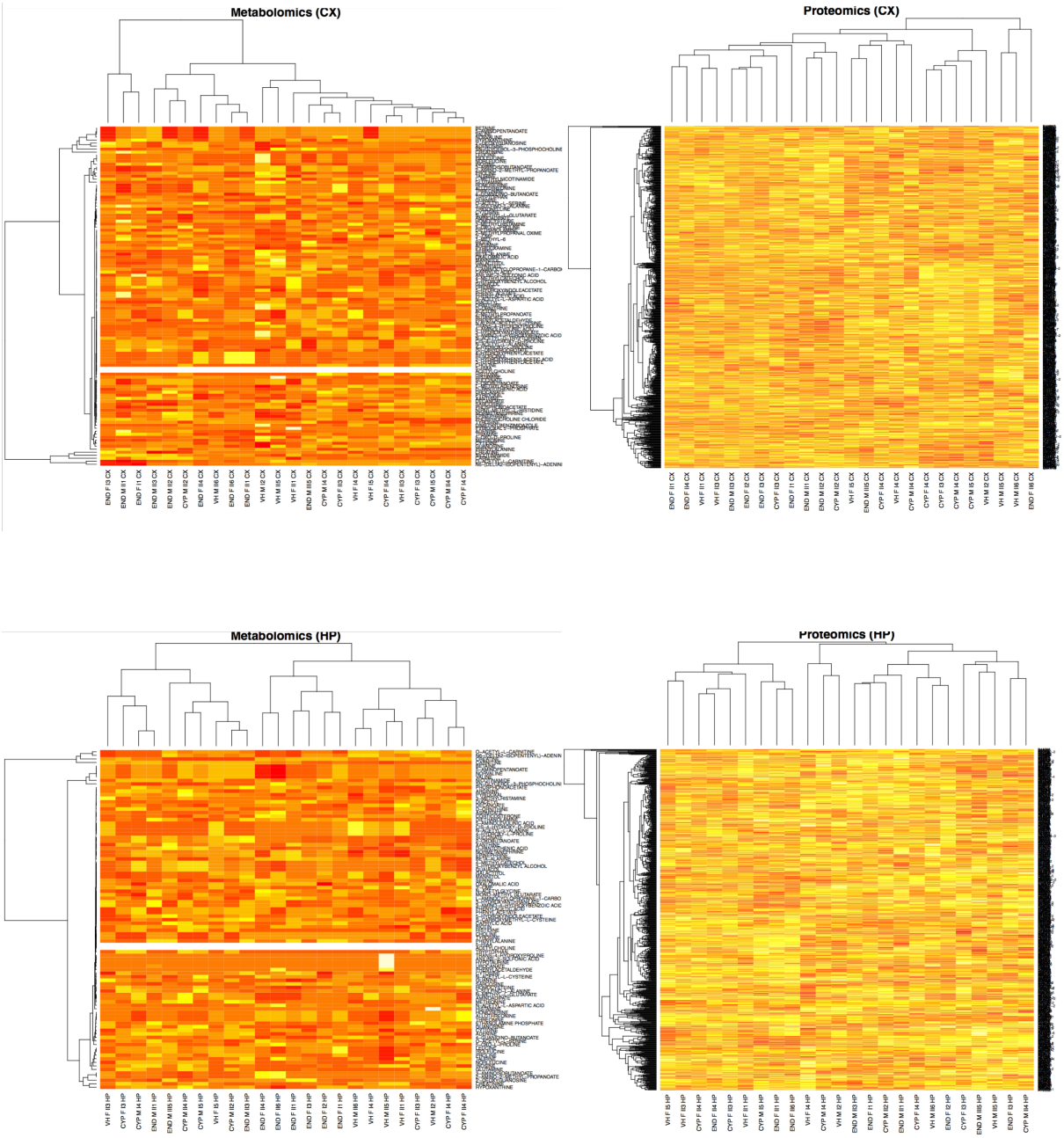
Before pre-processing:



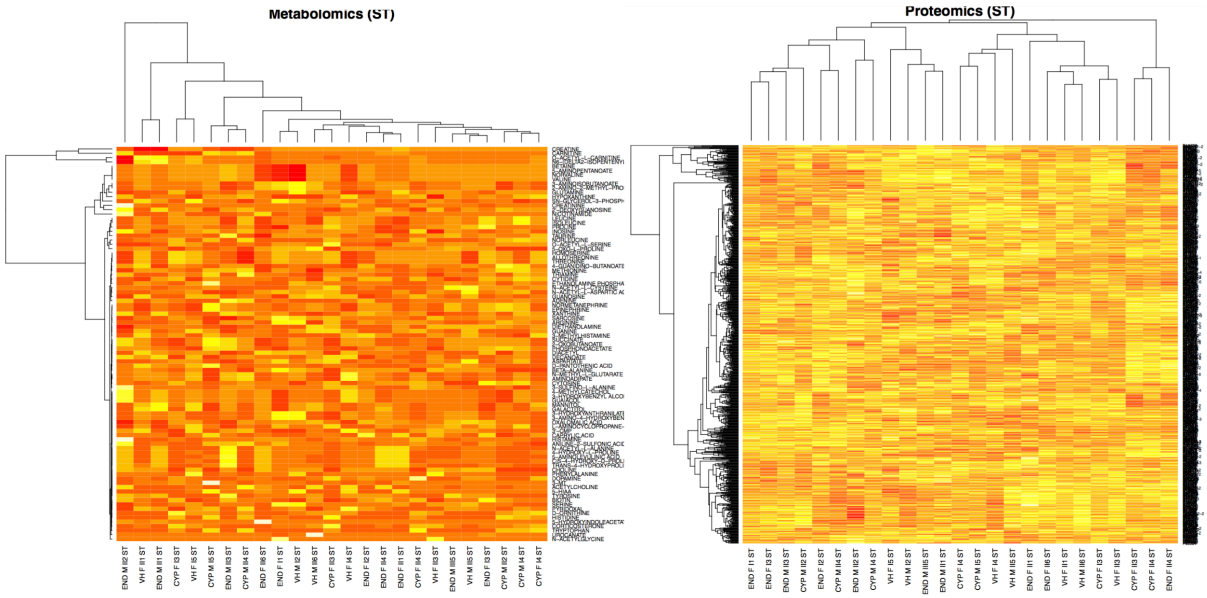
After normalization of metabolomics data:

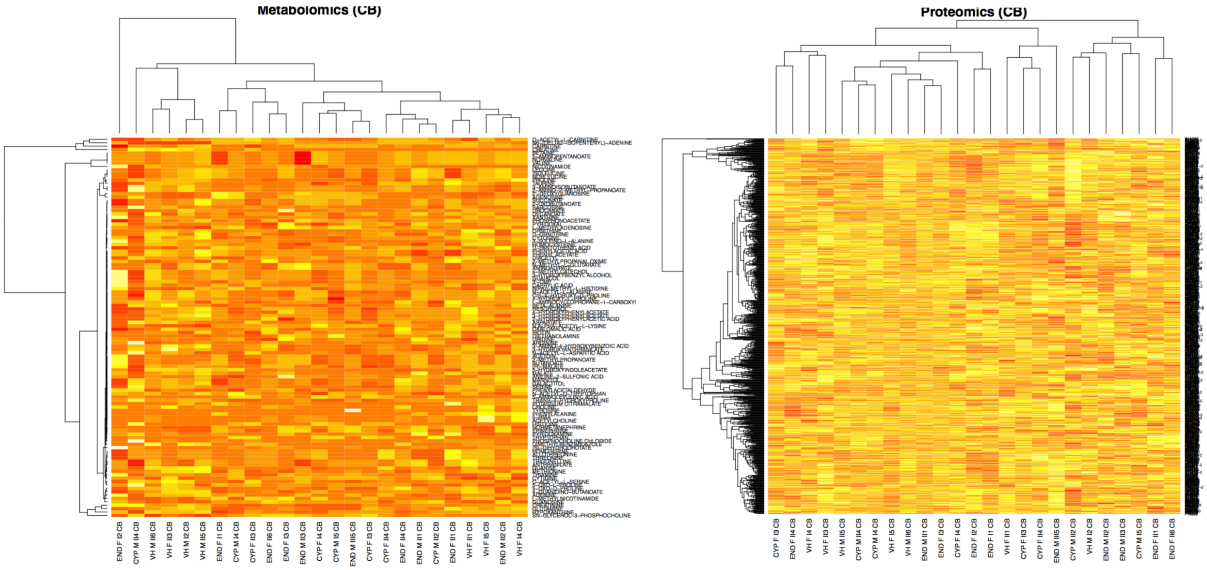


After normalization and arsynseq on of metabolomics data and arsynseq on proteomics:



After normalization and arsynseq on of metabolomics data and arsynseq on proteomics:

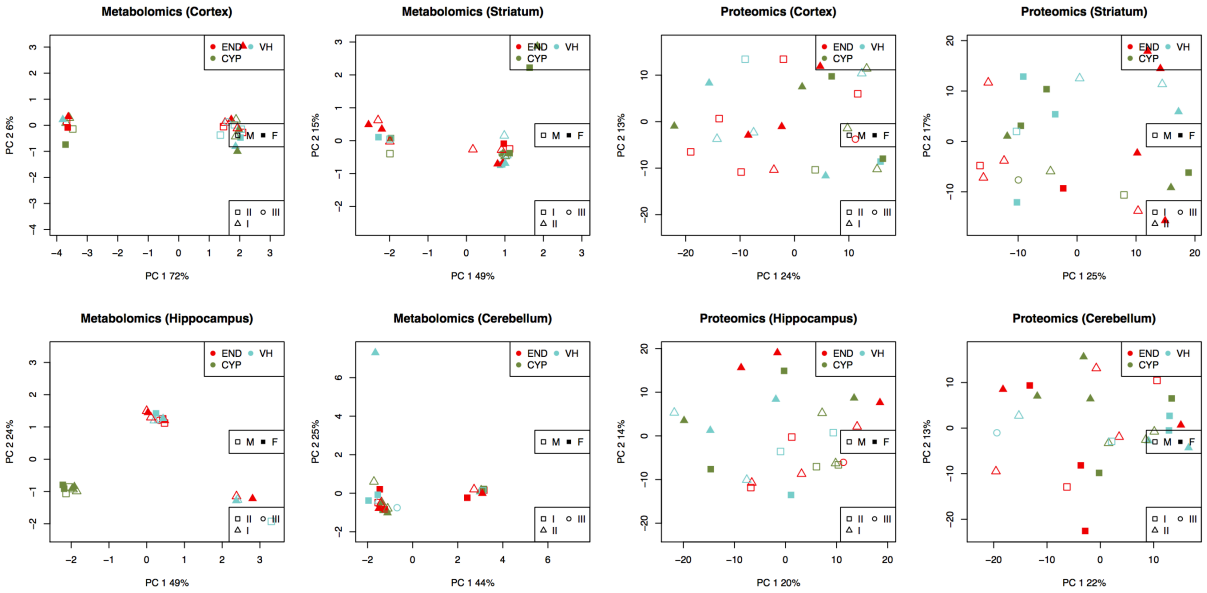




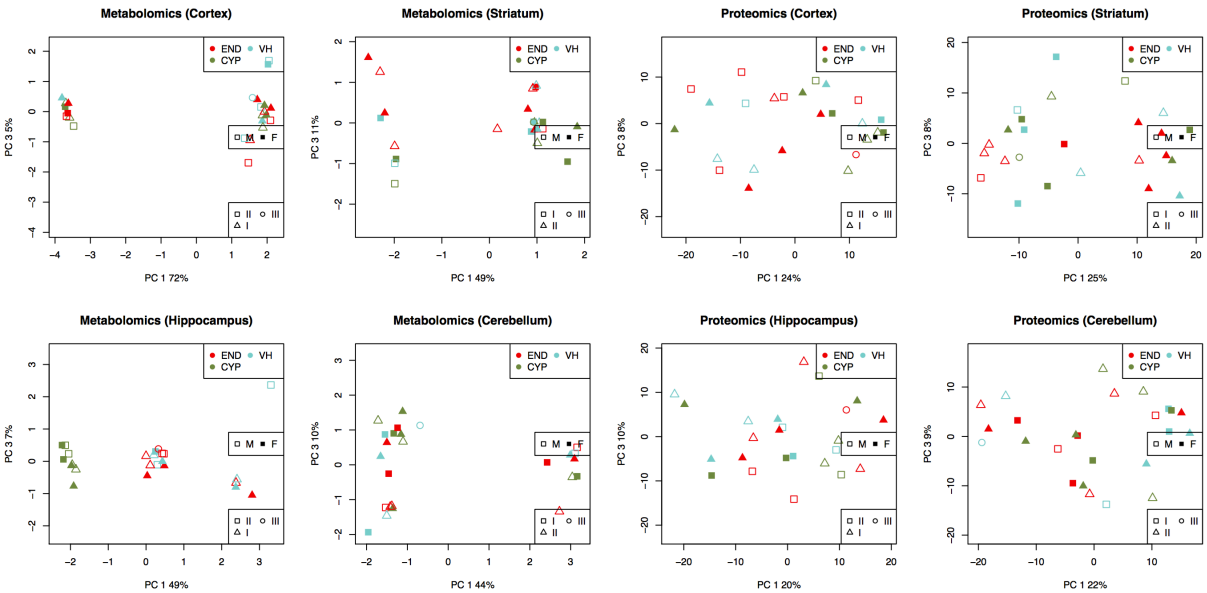
7.2.4. Attachment XV: PCA Set 01

Before pre-processing:

PCA 1 & 2

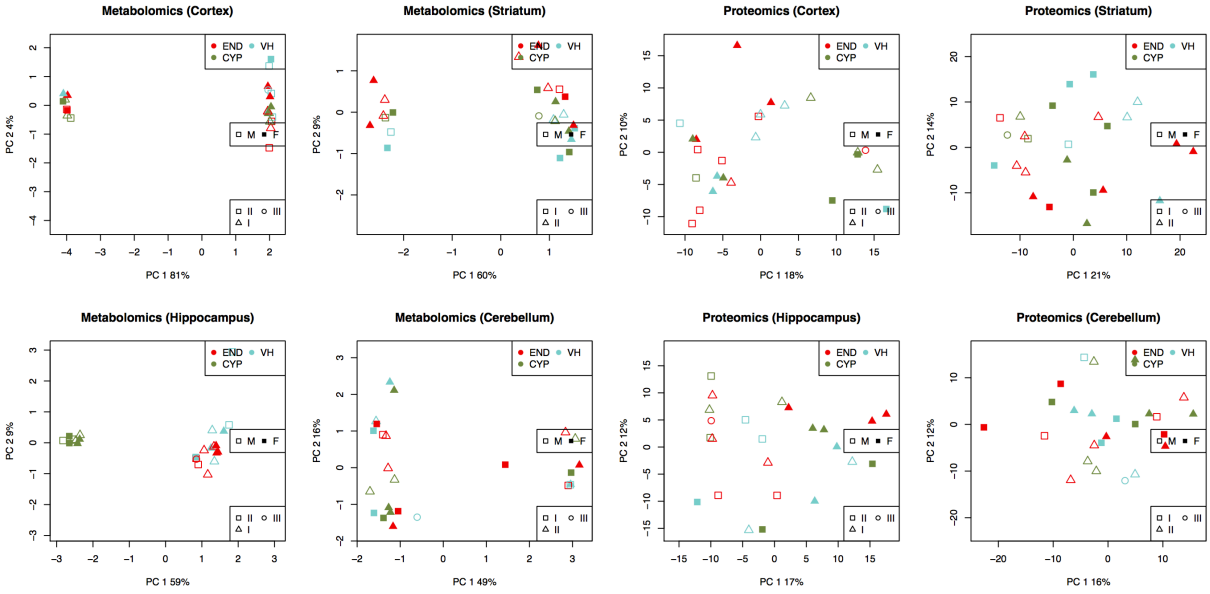


PCA 1 & 3

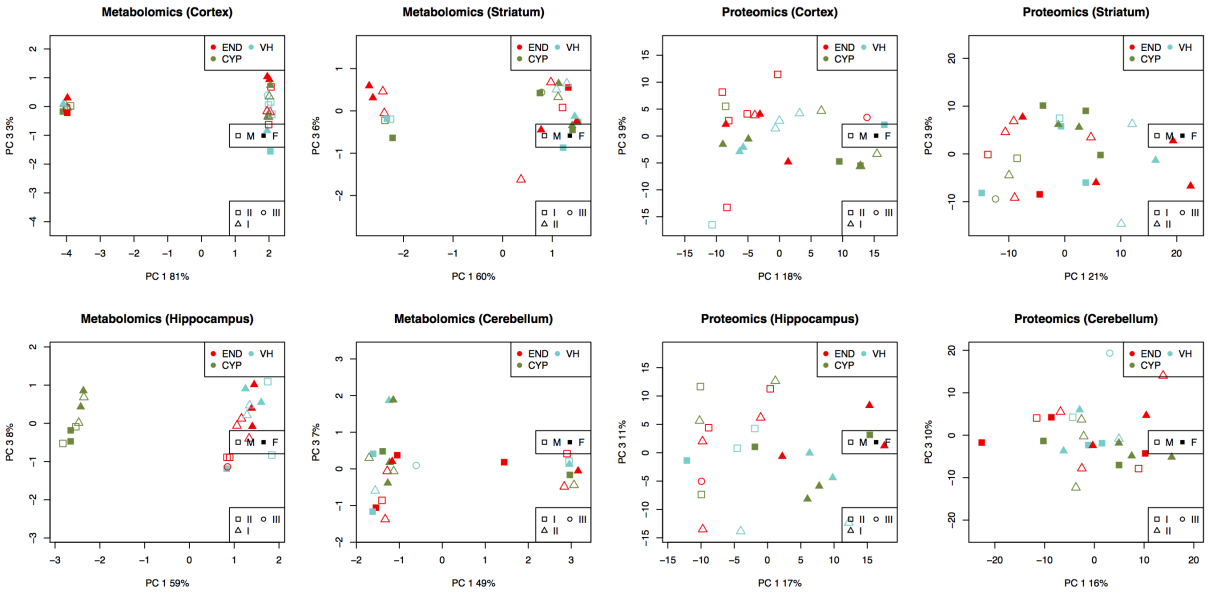


After pre-processing:

PCA 1 & 2

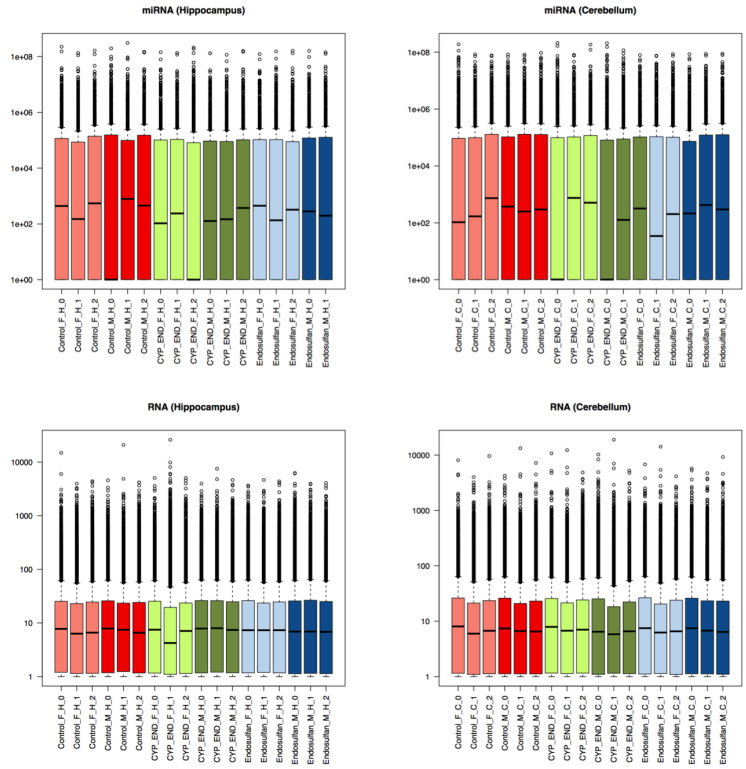


PCA 1 & 3

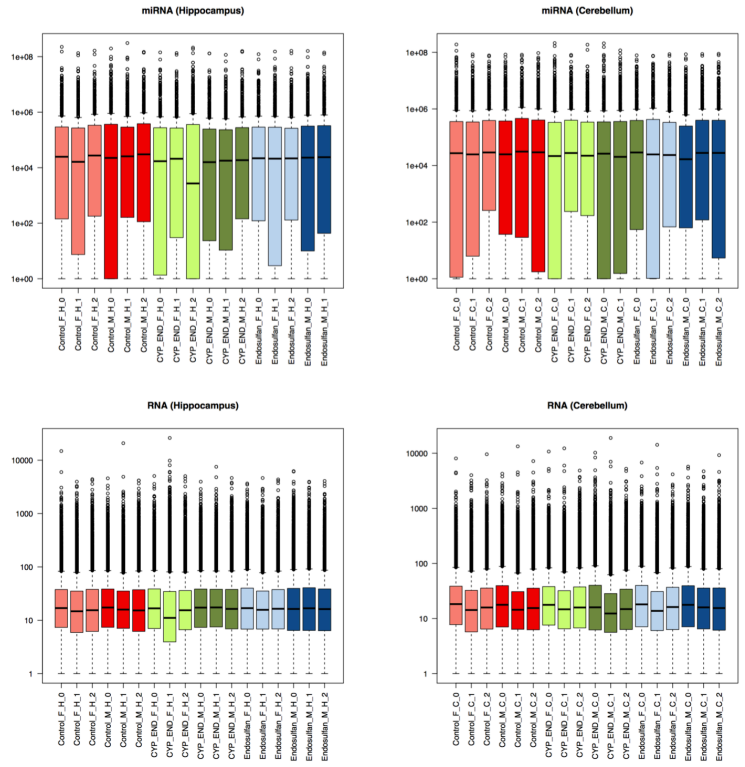


7.2.5. Attachment XVI: Transcriptomics boxplots

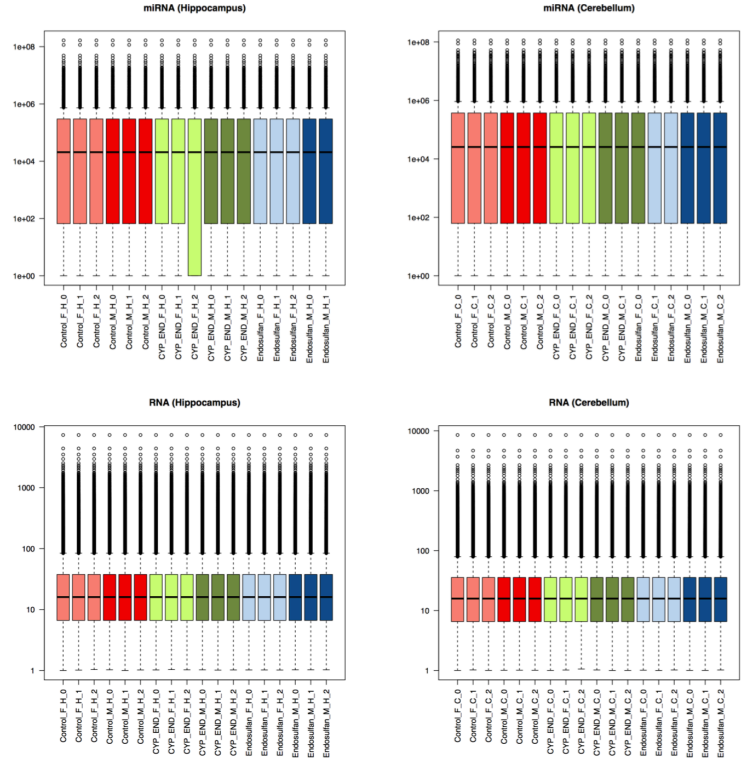
Before pre-processing:



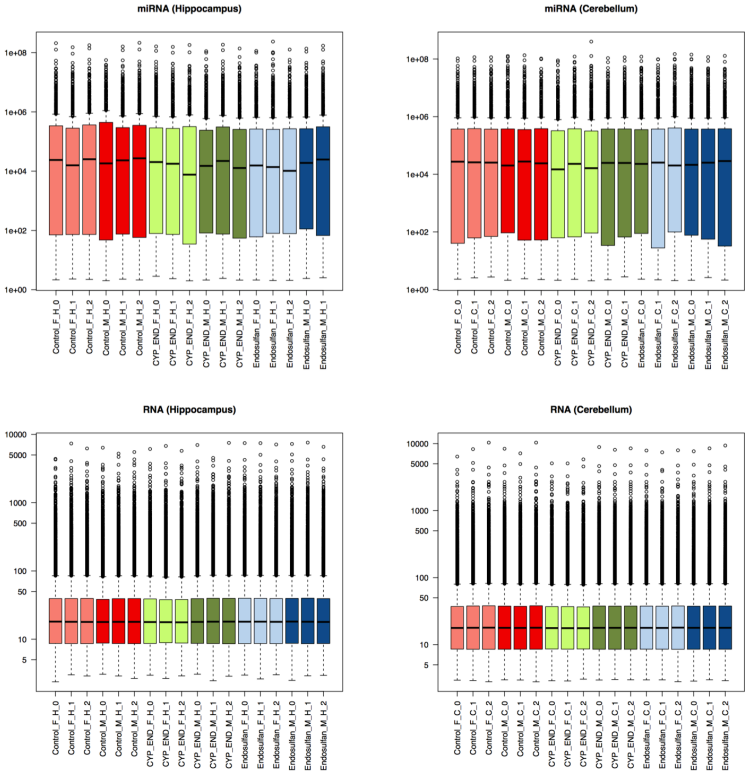
After filter:



After normalization:



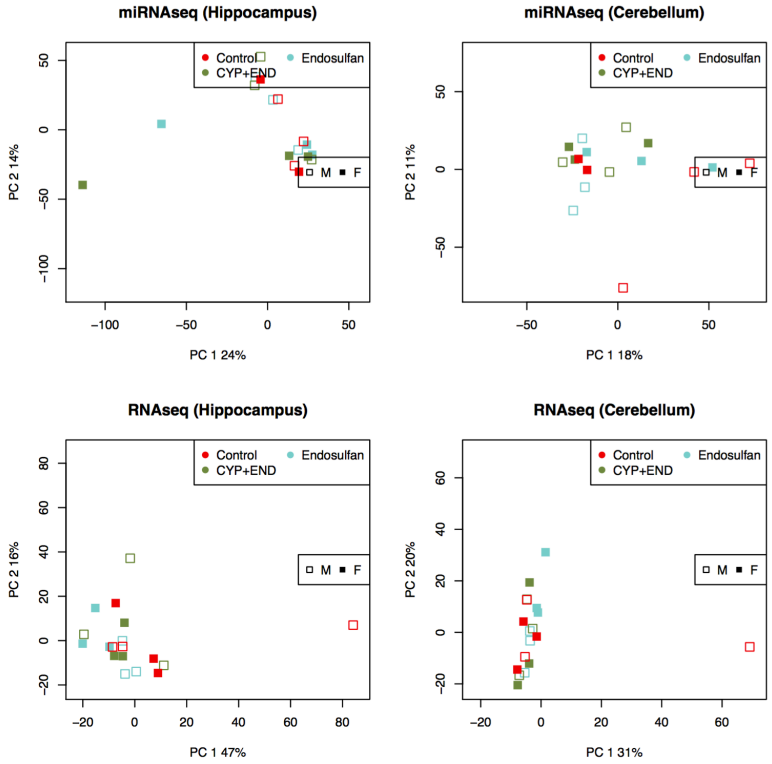
After arsynseq and normalization:



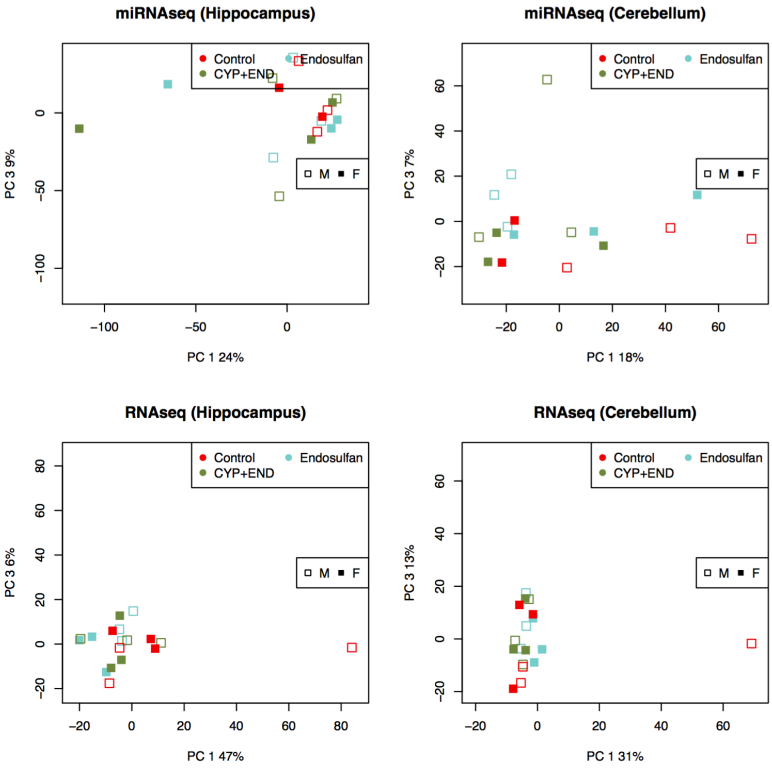
7.2.6. Attachment XVII: PCA transcriptomics

Before pre-processing:

PCA 1 & 2

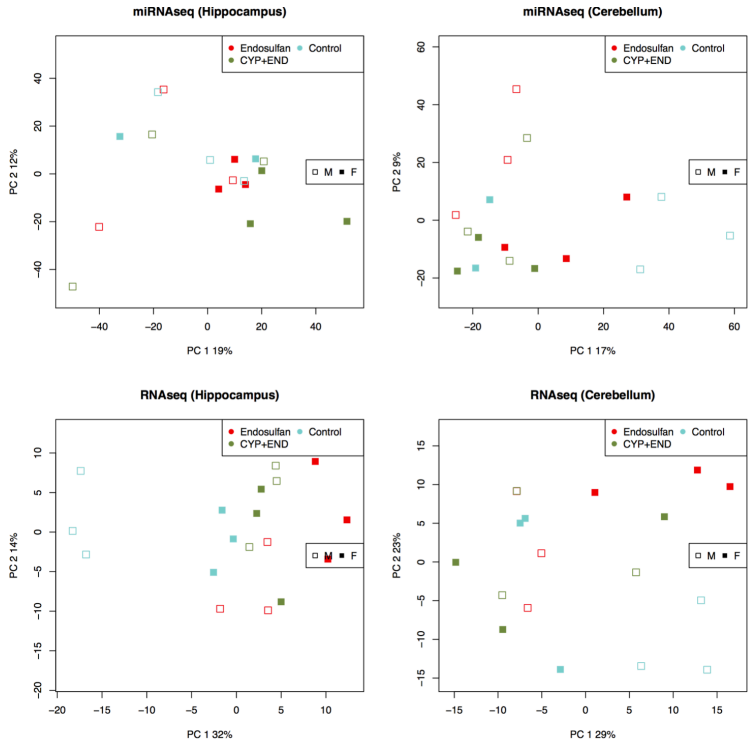


PCA 1 & 3

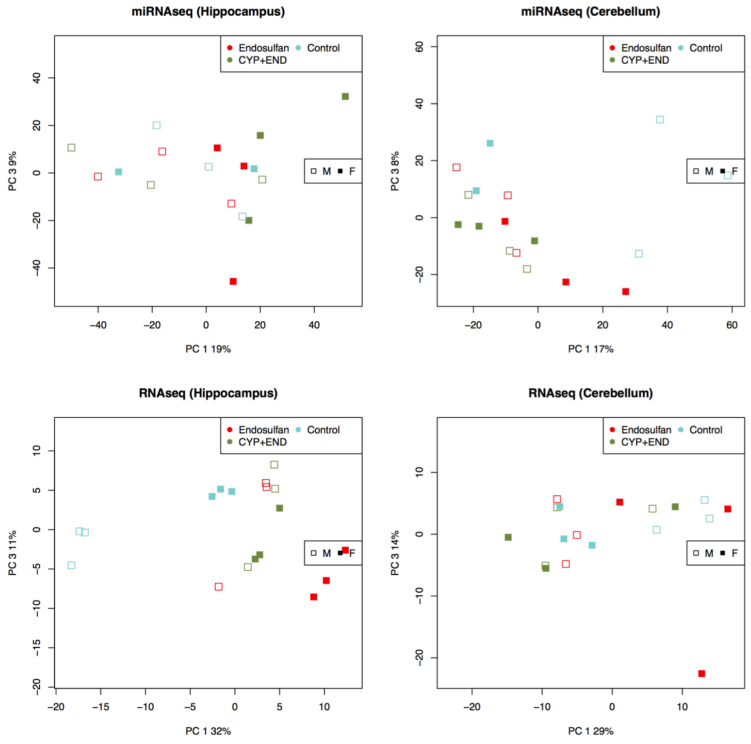


After pre-processing:

PCA 1 & 2

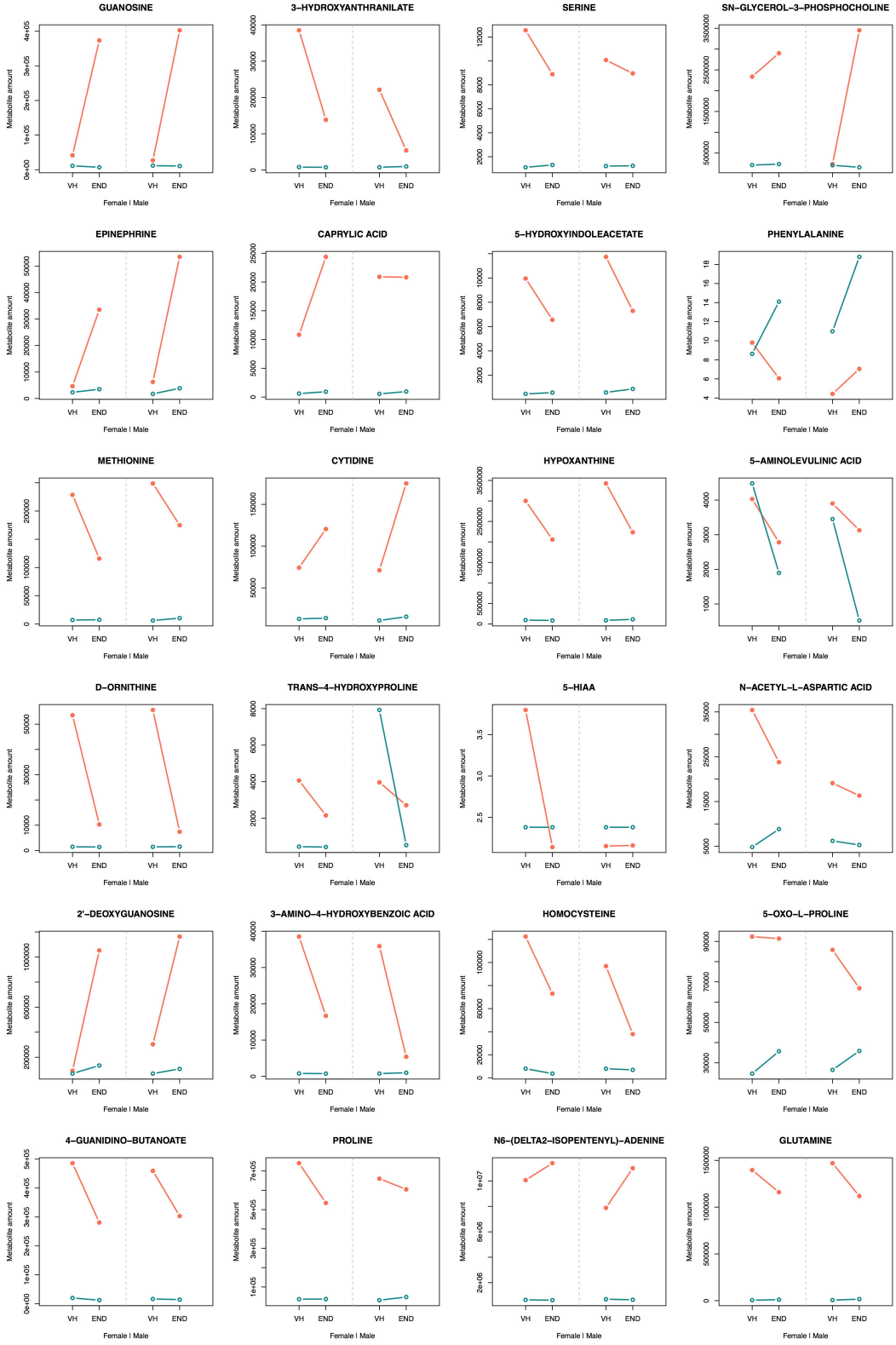


PCA 1 & 3

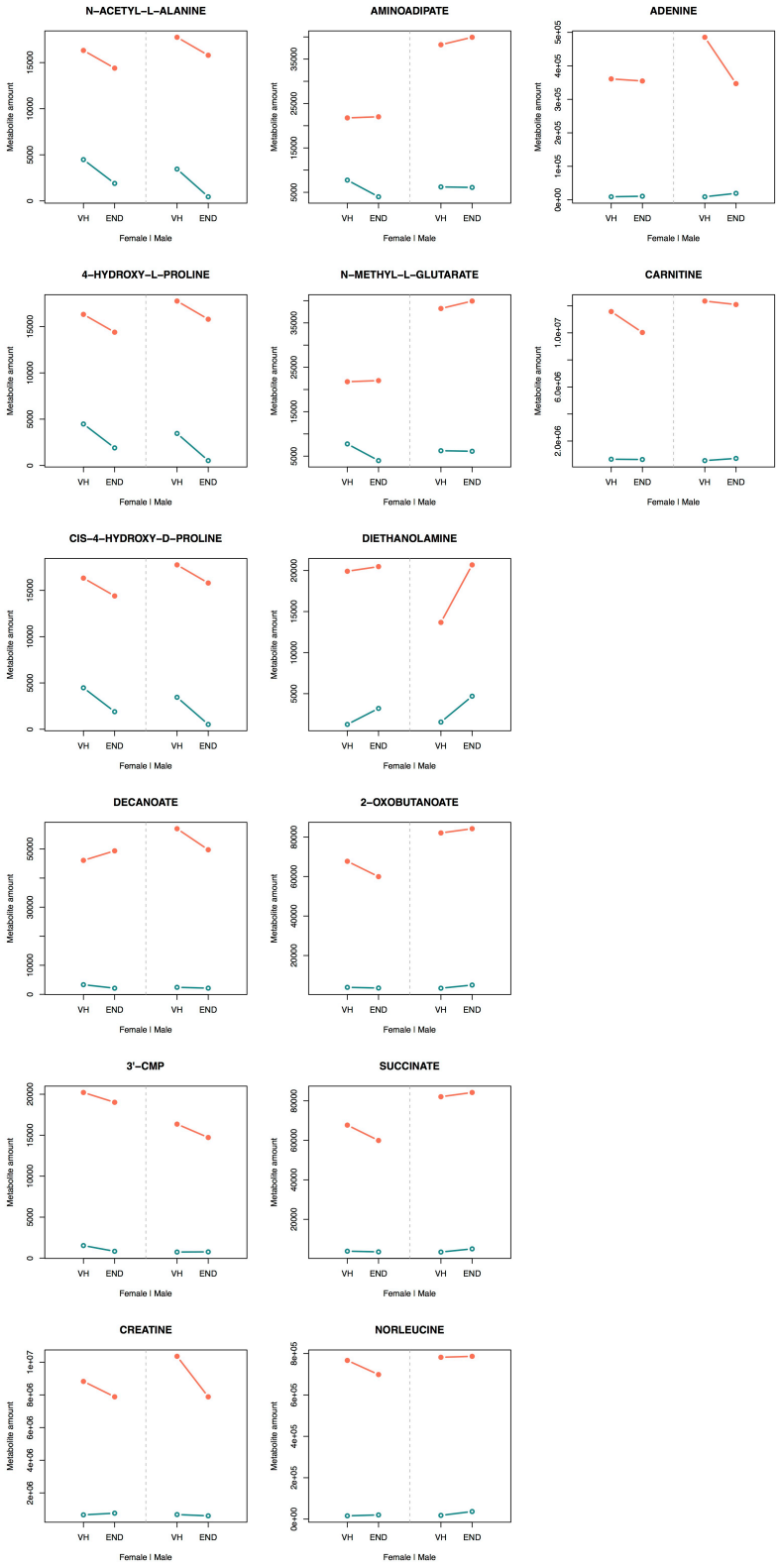


7.2.7. Attachment XVIII: Expression profiles

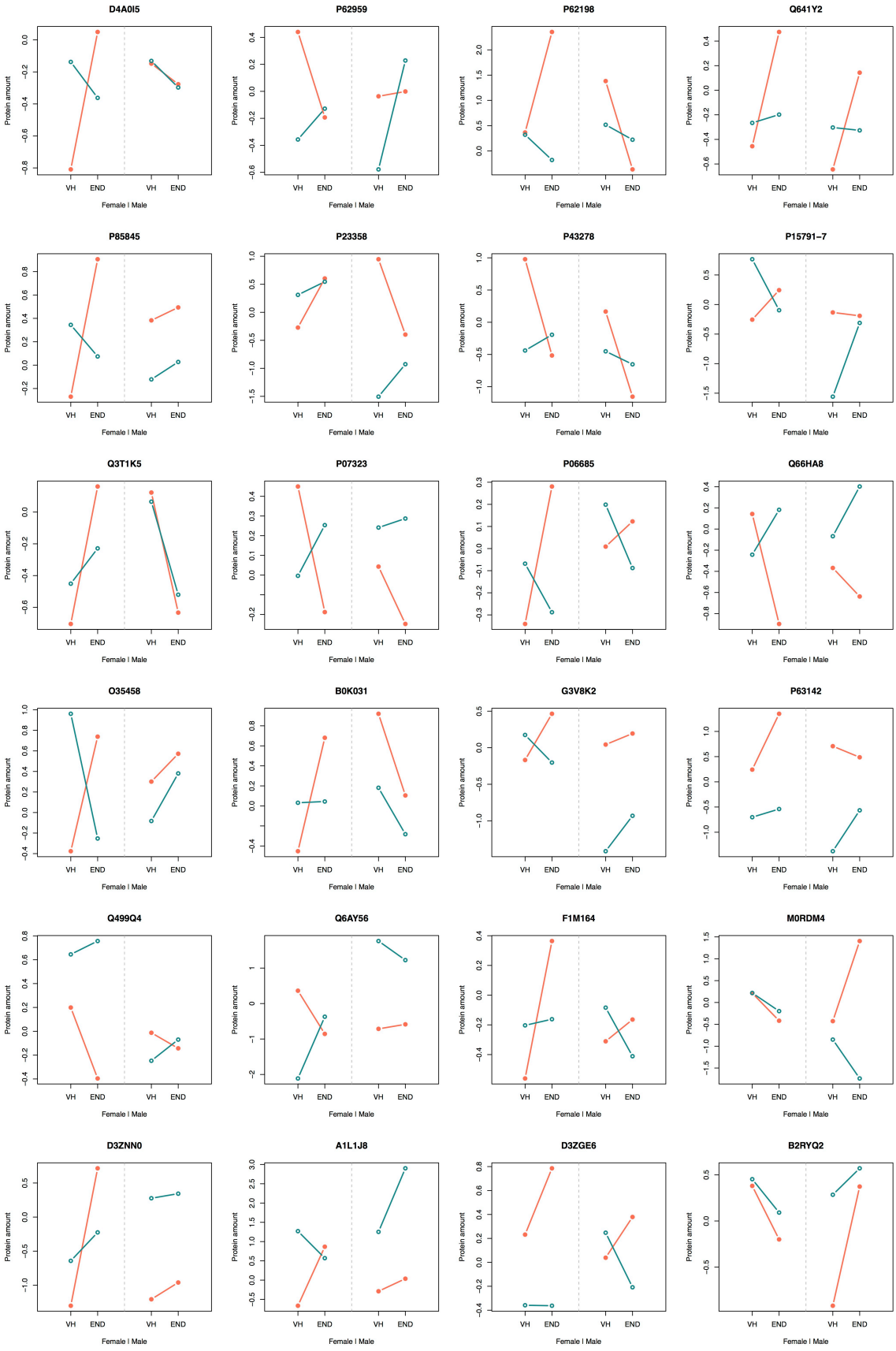
Metabolomics



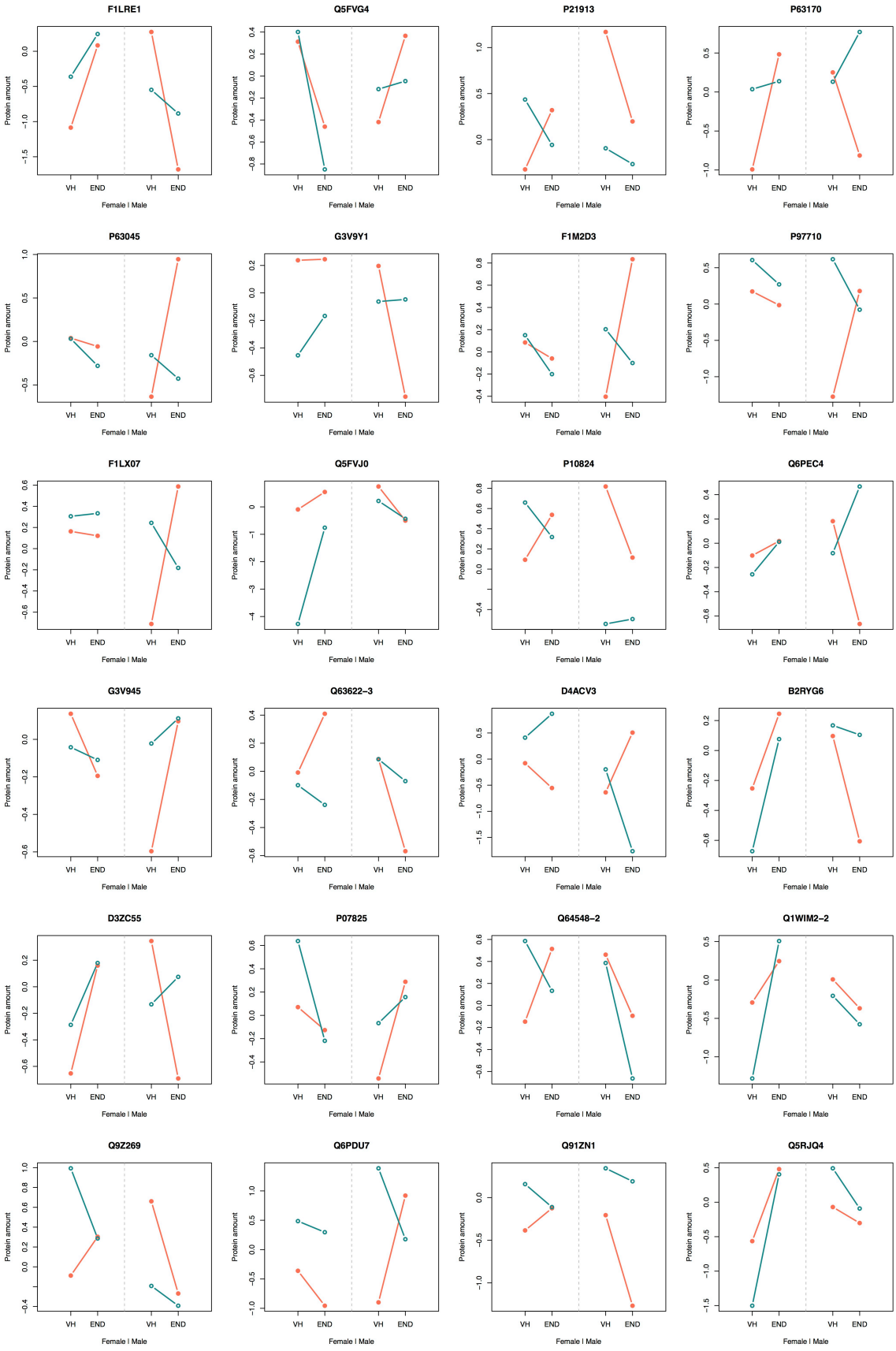
Metabolomics



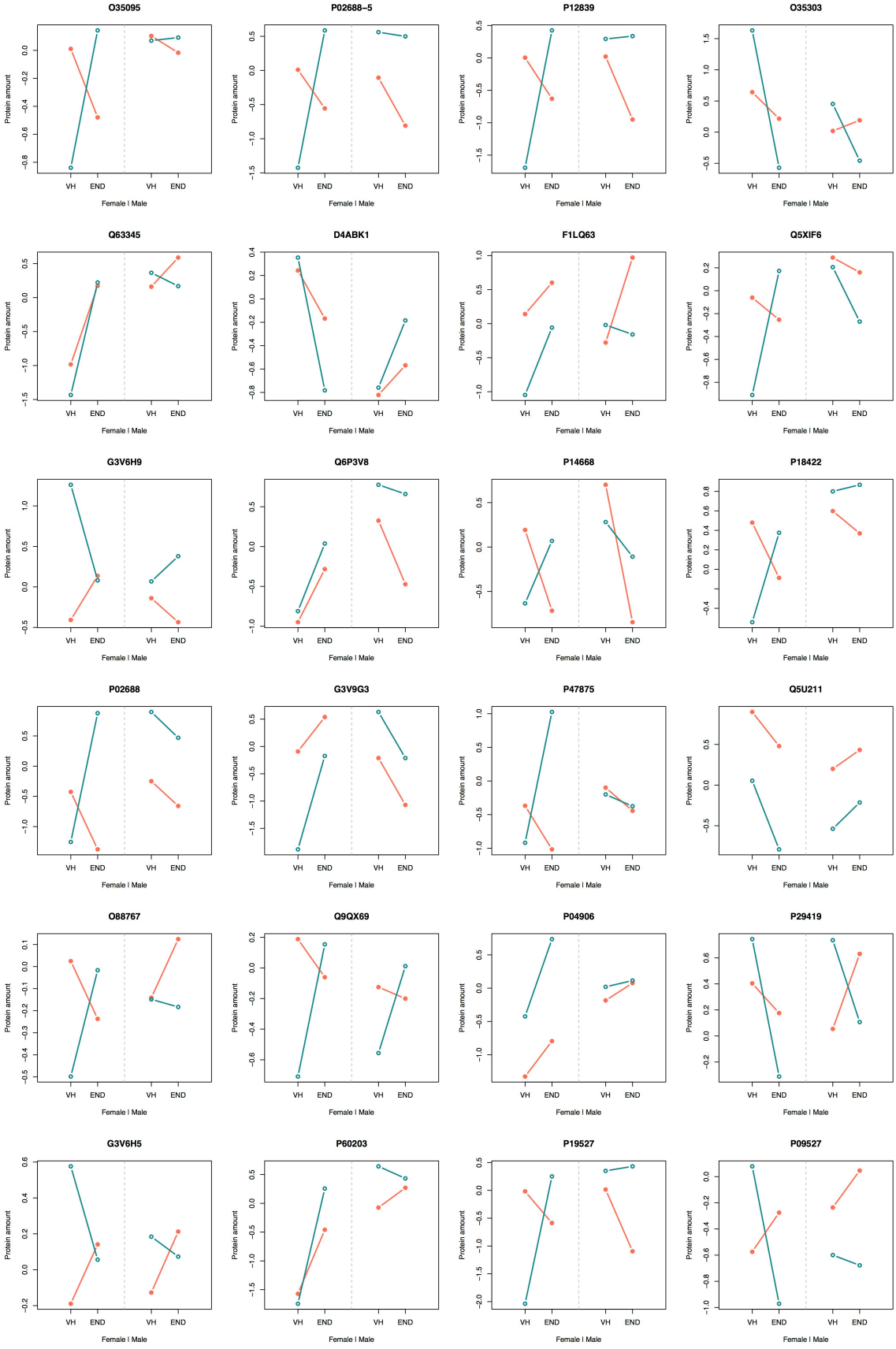
Proteomics


D4A0I5
P62959
P62198
Q641Y2
P85845
P23358
P43278
P15791−7
Q3T1K5
P07323
P06685
Q66HA8
O35458
B0K031
G3V8K2
P63142
Q499Q4
Q6AY56
F1M164
M0RDM4
D3ZNN0
A1L1J8
D3ZGE6
B2RYQ2
Protein amount
VH
END
Female | Male

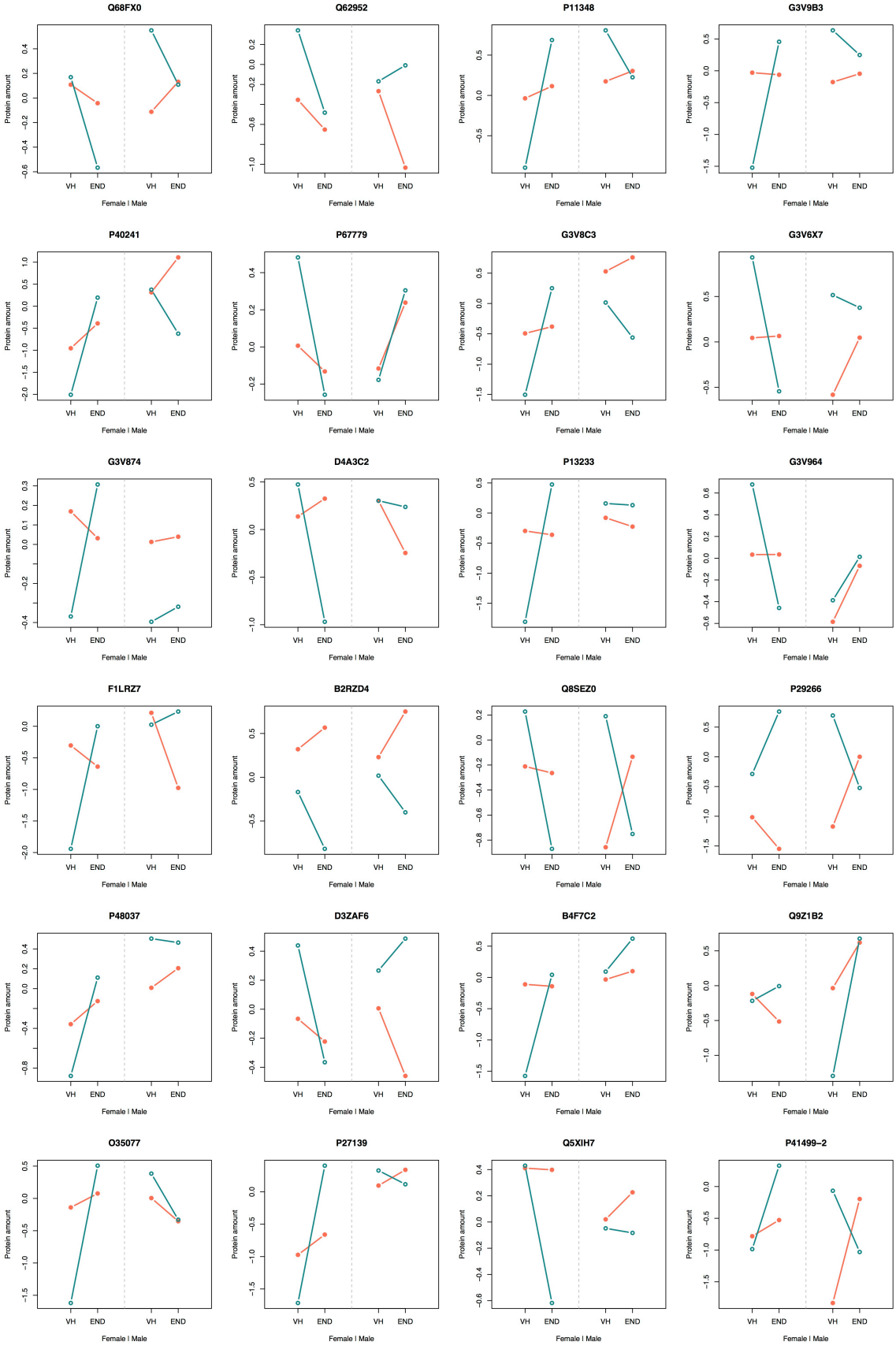
Proteomics


F1LRE1
Q5FVG4
P21913
P63170
P63045
G3V9Y1
F1M2D3
P97710
F1LX07
Q5FVJ0
P10824
Q6PEC4
G3V945
Q63622−3
D4ACV3
B2RYG6
D3ZC55
P07825
Q64548−2
Q1WIM2−2
Q9Z269
Q6PDU7
Q91ZN1
Q5RJQ4
Protein amount
VH
END
Female | Male

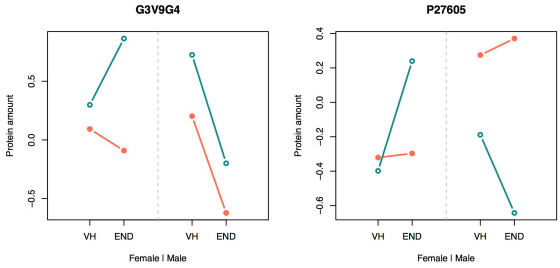
Proteomics


O35095
P02688–5
P12839
O35303
Q63345
D4ABK1
F1LQ63
Q5XIF6
G3V6H9
Q6P3V8
P14668
P18422
P02688
G3V9G3
P47875
Q5U211
O88767
Q9QX69
P04906
P29419
G3V6H5
P60203
P19527
P09527
Protein amount
VH
END
Female | Male

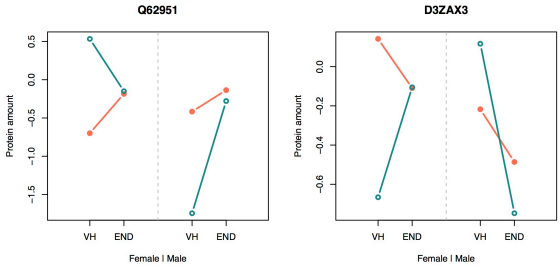
Proteomics


Q68FX0
Q62952
P11348
G3V9B3
P40241
P67779
G3V8C3
G3V6X7
G3V874
D4A3C2
P13233
G3V964
F1LRZ7
B2RZD4
Q8SEZ0
P29266
P48037
D3ZAF6
B4F7C2
Q9Z1B2
O35077
P27139
Q5XIH7
P41499-2
Protein amount
VH
END
Female | Male

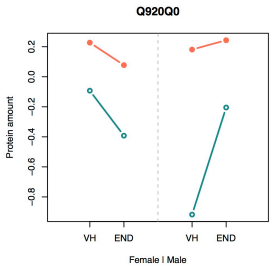
Proteomics

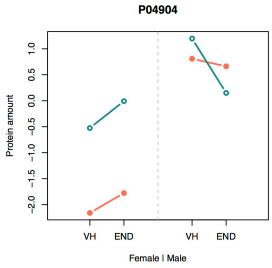

G3V9G4
Protein amount
0.5
0.0
−0.5
VH
END
VH
END
Female | Male

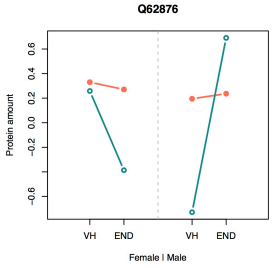
P27605
Protein amount
0.4
0.2
0.0
−0.2
−0.4
−0.6
VH
END
VH
END
Female | Male

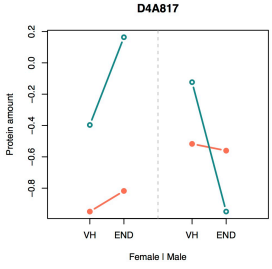

Q62951
Protein amount
0.5
0.0
−0.5
−1.0
−1.5
VH
END
VH
END
Female | Male

D3ZAX3
Protein amount
0.0
−0.2
−0.4
−0.6
VH
END
VH
END
Female | Male

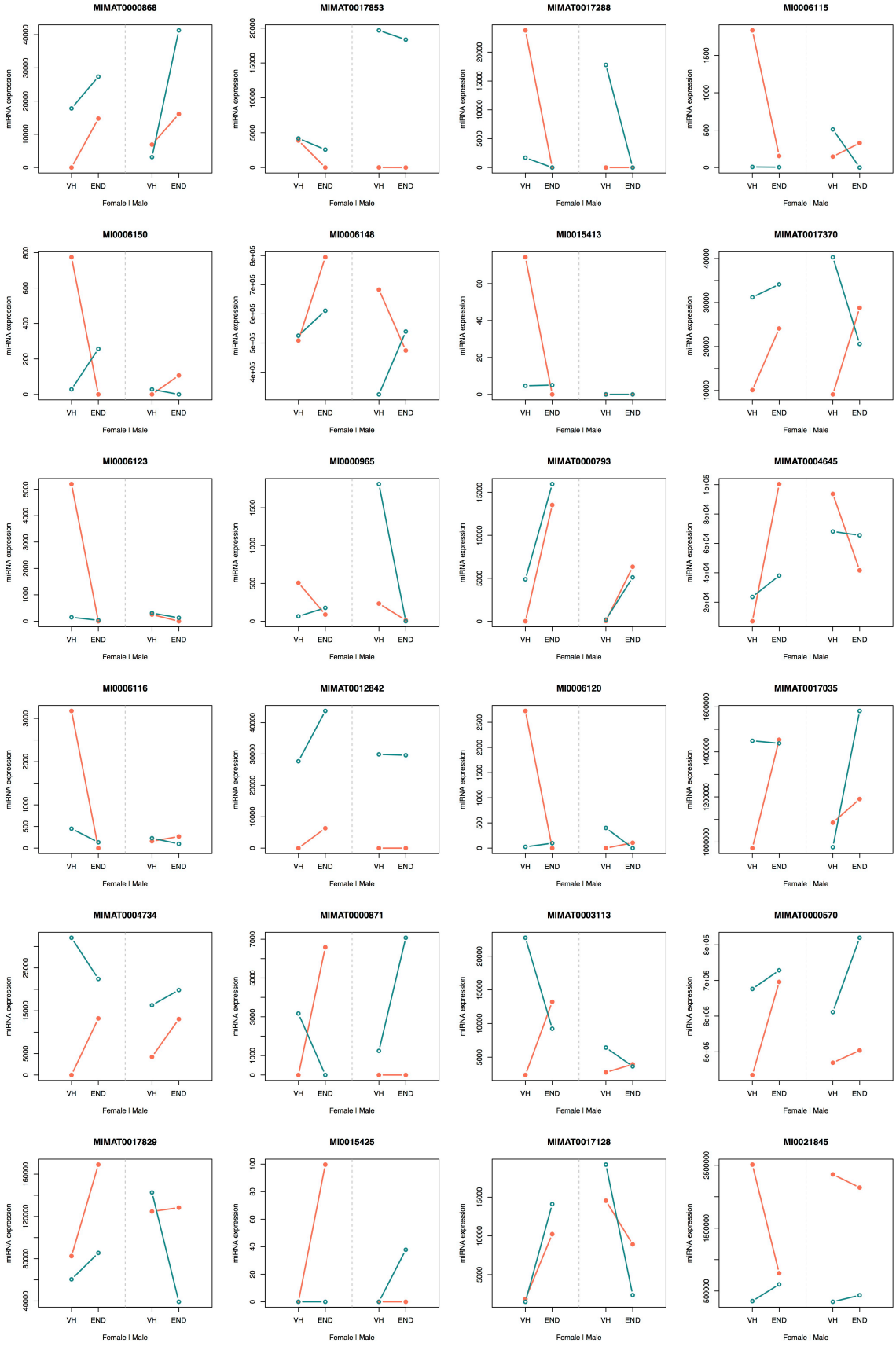

Q920Q0
Protein amount
0.2
0.0
−0.2
−0.4
−0.6
−0.8
VH
END
VH
END
Female | Male


P04904
Protein amount
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
VH
END
VH
END
Female | Male

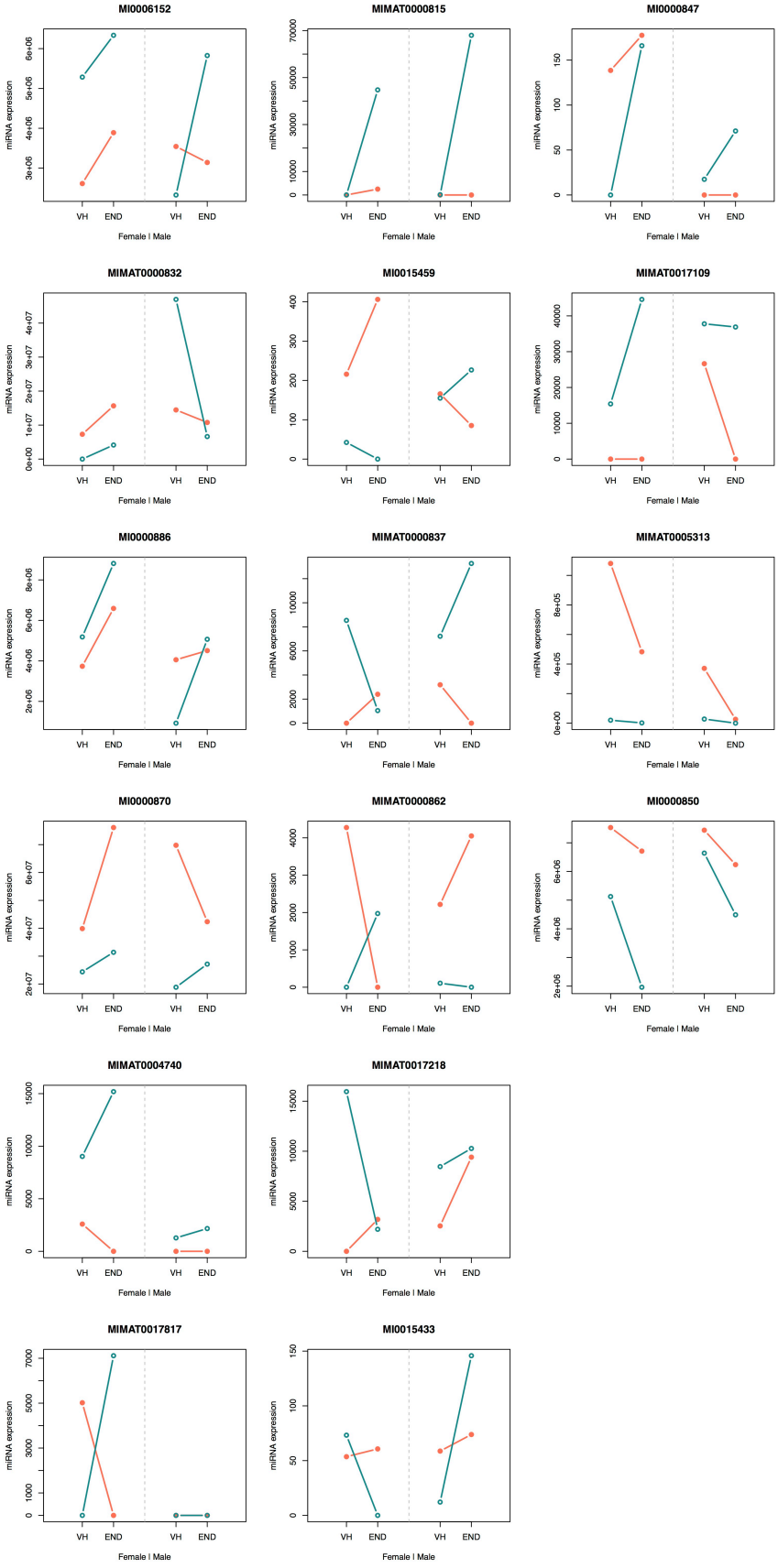

Q62876
Protein amount
0.6
0.4
0.2
0.0
−0.2
−0.6
VH
END
VH
END
Female | Male


D4A817
Protein amount
0.2
0.0
−0.2
−0.4
−0.6
−0.8
VH
END
VH
END
Female | Male

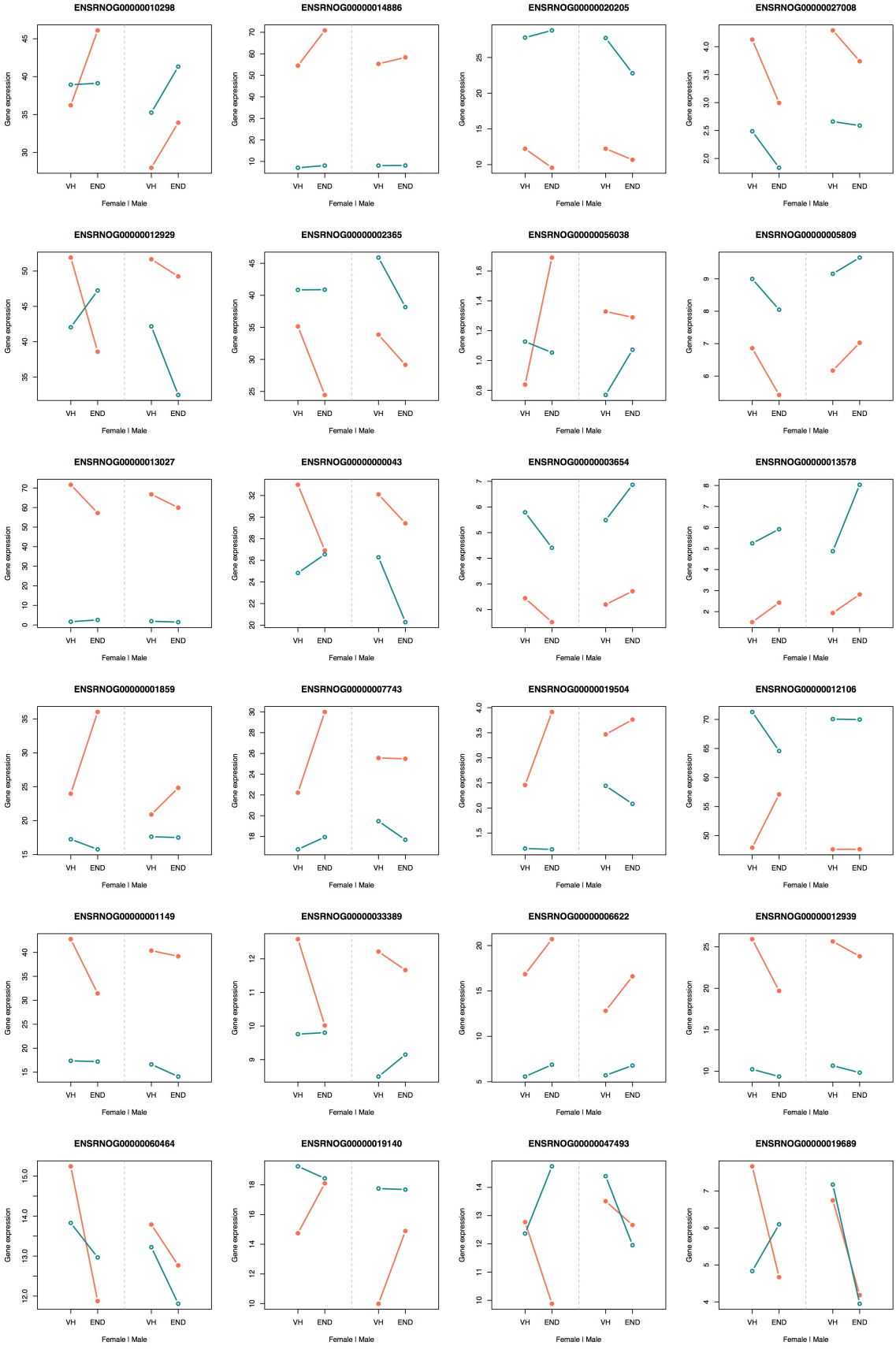
miRNA-seq


MIMAT0000868
MIMAT0017853
MIMAT0017288
MI0006115
MI0006150
MI0006148
MI0015413
MIMAT0017370
MI0006123
MI0000965
MIMAT0000793
MIMAT0004645
MI0006116
MIMAT0012842
MI0006120
MIMAT0017035
MIMAT0004734
MIMAT0000871
MIMAT0003113
MIMAT0000570
MIMAT0017829
MI0015425
MIMAT0017128
MI0021845
miRNA expression
VH
END
Female | Male

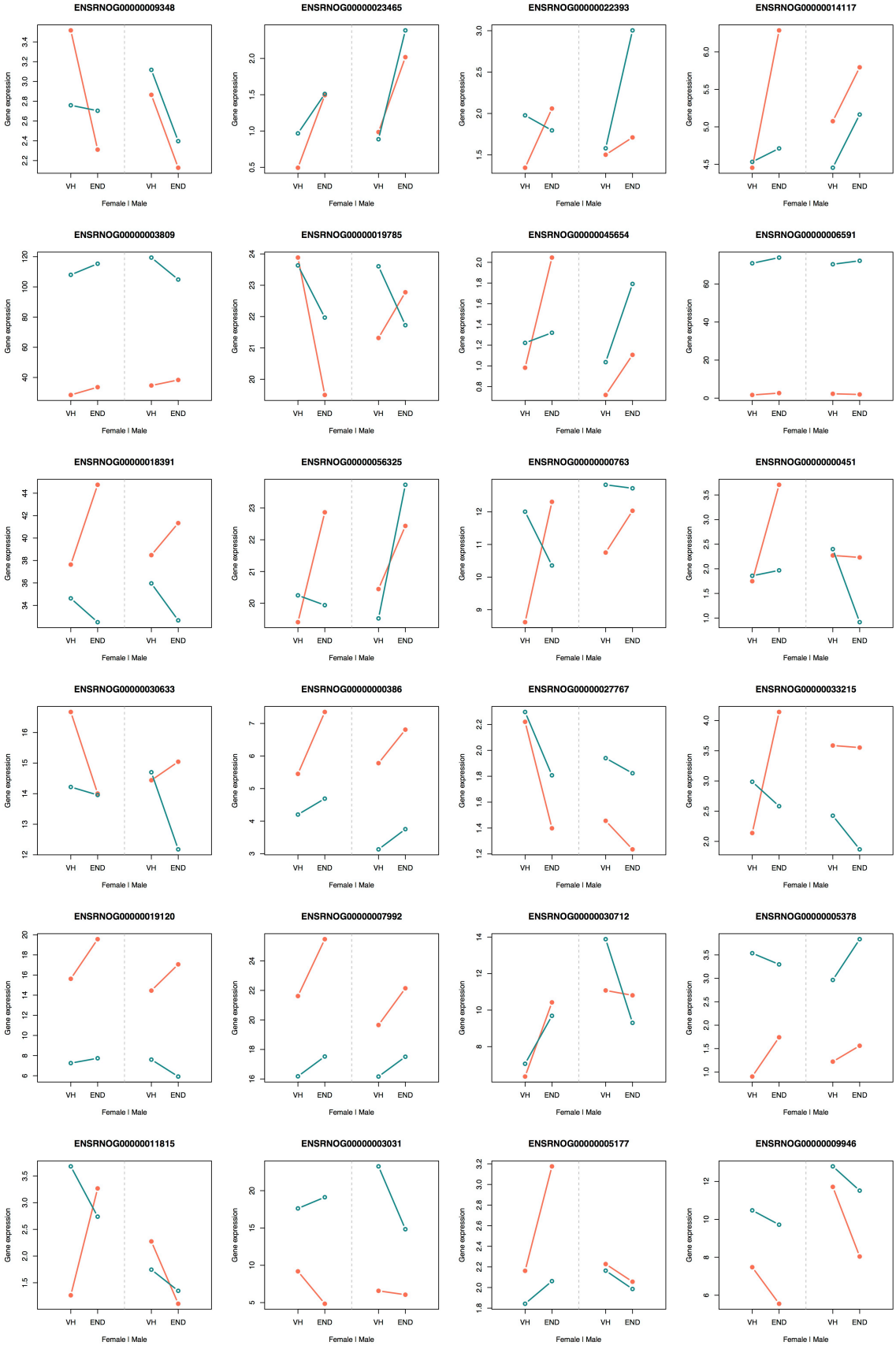
miRNA-seq


MI0006152
MIMAT0000815
MI0000847
MIMAT0000832
MI0015459
MIMAT0017109
MI0000886
MIMAT0000837
MIMAT0005313
MI0000870
MIMAT0000862
MI0000850
MIMAT0004740
MIMAT0017218
MIMAT0017817
MI0015433
miRNA expression
VH
END
Female | Male

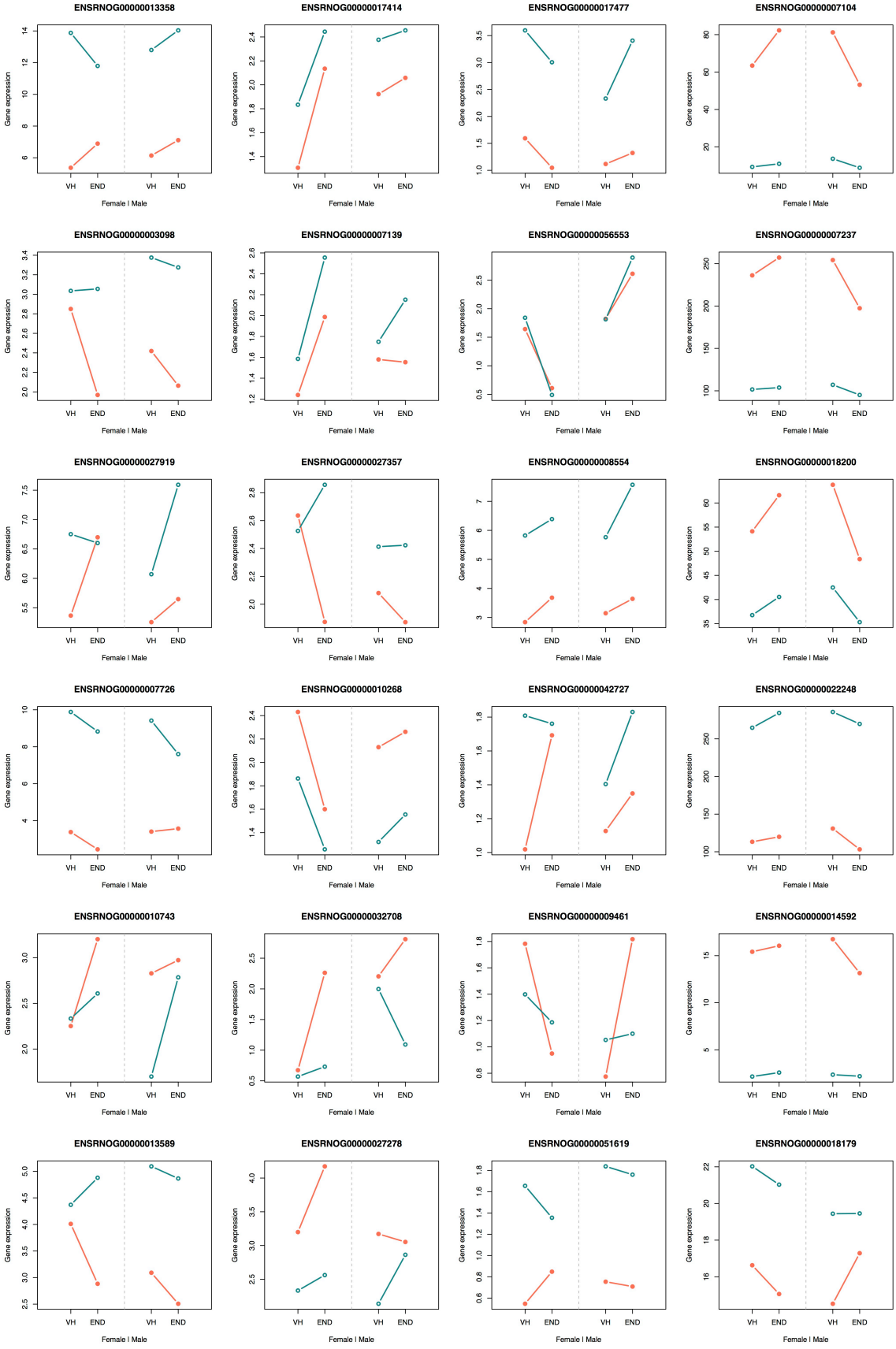
RNA-seq


ENSRNOG00000010298
ENSRNOG00000014886
ENSRNOG00000020205
ENSRNOG00000027008
ENSRNOG00000012929
ENSRNOG00000002365
ENSRNOG00000056038
ENSRNOG00000005809
ENSRNOG00000013027
ENSRNOG00000000043
ENSRNOG00000003654
ENSRNOG00000013578
ENSRNOG00000001859
ENSRNOG00000007743
ENSRNOG00000019504
ENSRNOG00000012106
ENSRNOG00000001149
ENSRNOG00000033389
ENSRNOG00000006622
ENSRNOG00000012939
ENSRNOG00000060464
ENSRNOG00000019140
ENSRNOG00000047493
ENSRNOG00000019689
Gene expression
VH END VH END
Female | Male

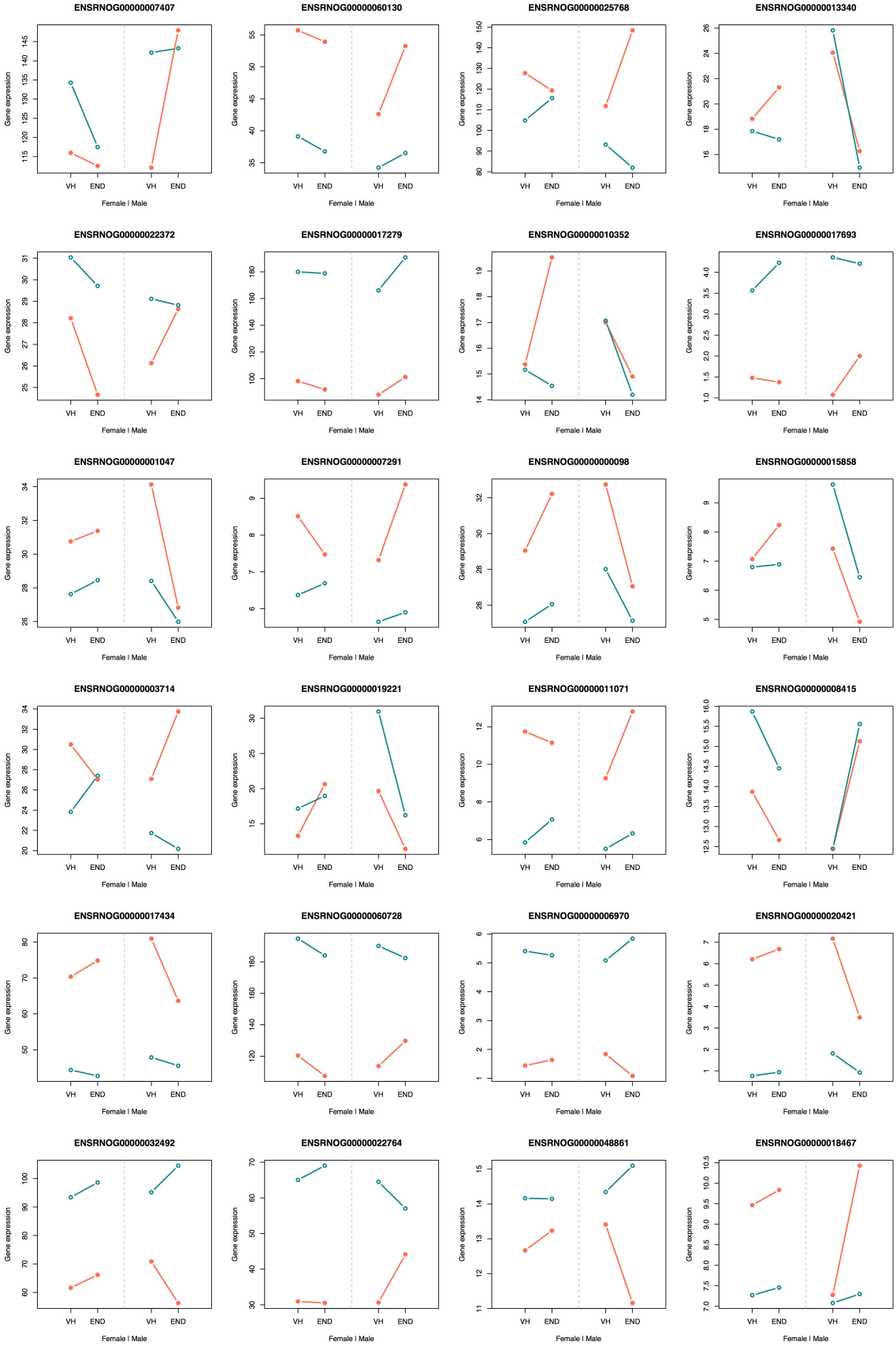
RNA-seq


ENSRNOG00000009348
ENSRNOG00000023465
ENSRNOG00000022393
ENSRNOG00000014117
ENSRNOG00000003809
ENSRNOG00000019785
ENSRNOG00000045654
ENSRNOG00000006591
ENSRNOG00000018391
ENSRNOG00000056325
ENSRNOG00000000763
ENSRNOG00000000451
ENSRNOG00000030633
ENSRNOG00000000386
ENSRNOG00000027767
ENSRNOG00000033215
ENSRNOG00000019120
ENSRNOG00000007992
ENSRNOG00000030712
ENSRNOG00000005378
ENSRNOG00000011815
ENSRNOG00000003031
ENSRNOG00000005177
ENSRNOG00000009946
Gene expression
VH END VH END
Female | Male

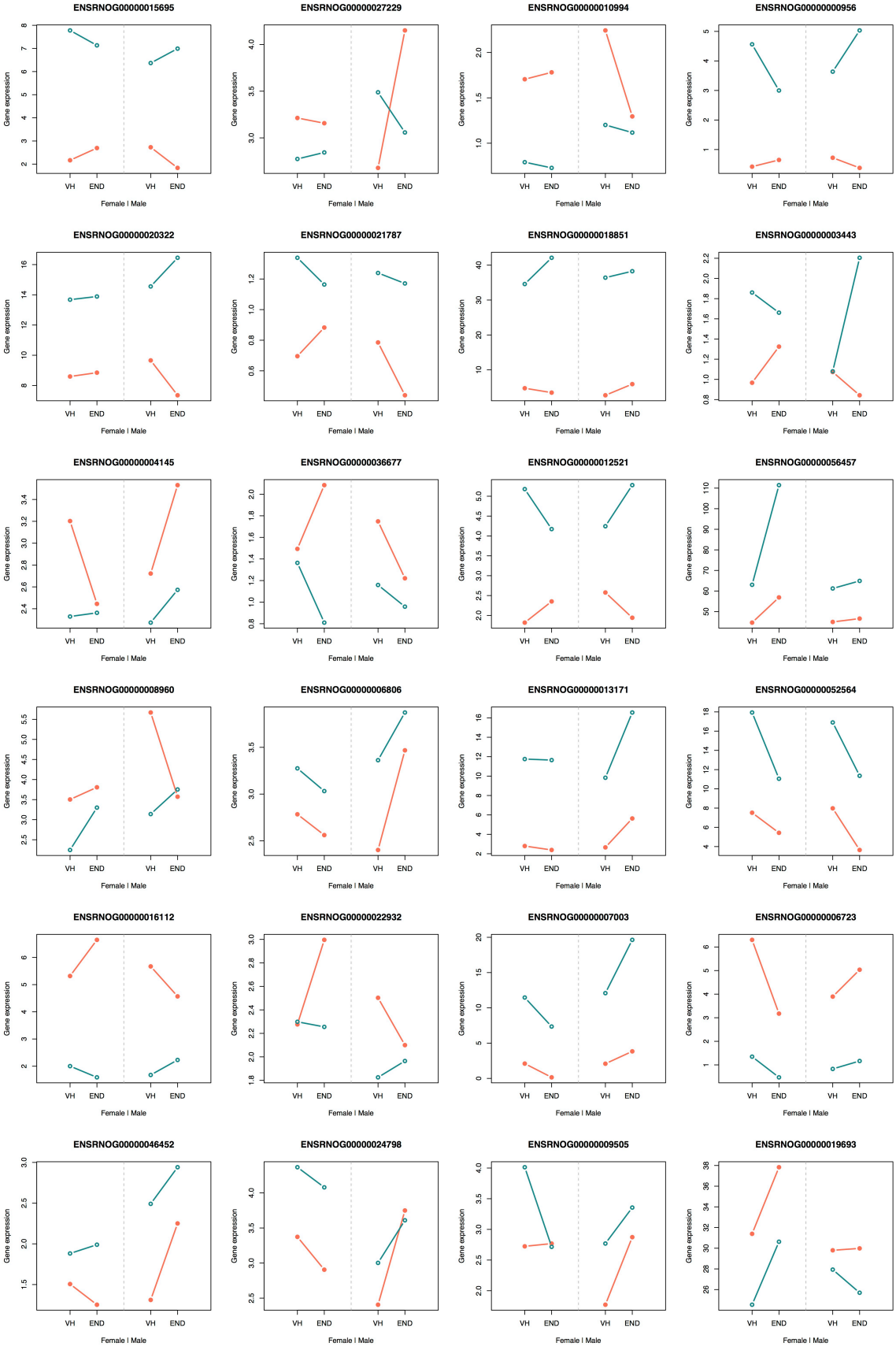
RNA-seq


ENSRNOG00000013358
ENSRNOG00000017414
ENSRNOG00000017477
ENSRNOG00000007104
ENSRNOG00000003098
ENSRNOG00000007139
ENSRNOG00000056553
ENSRNOG00000007237
ENSRNOG00000027919
ENSRNOG00000027357
ENSRNOG00000008554
ENSRNOG00000018200
ENSRNOG00000007726
ENSRNOG00000010268
ENSRNOG00000042727
ENSRNOG00000022248
ENSRNOG00000010743
ENSRNOG00000032708
ENSRNOG00000009461
ENSRNOG00000014592
ENSRNOG00000013589
ENSRNOG00000027278
ENSRNOG00000051619
ENSRNOG00000018179
Gene expression
VH
END
Female | Male

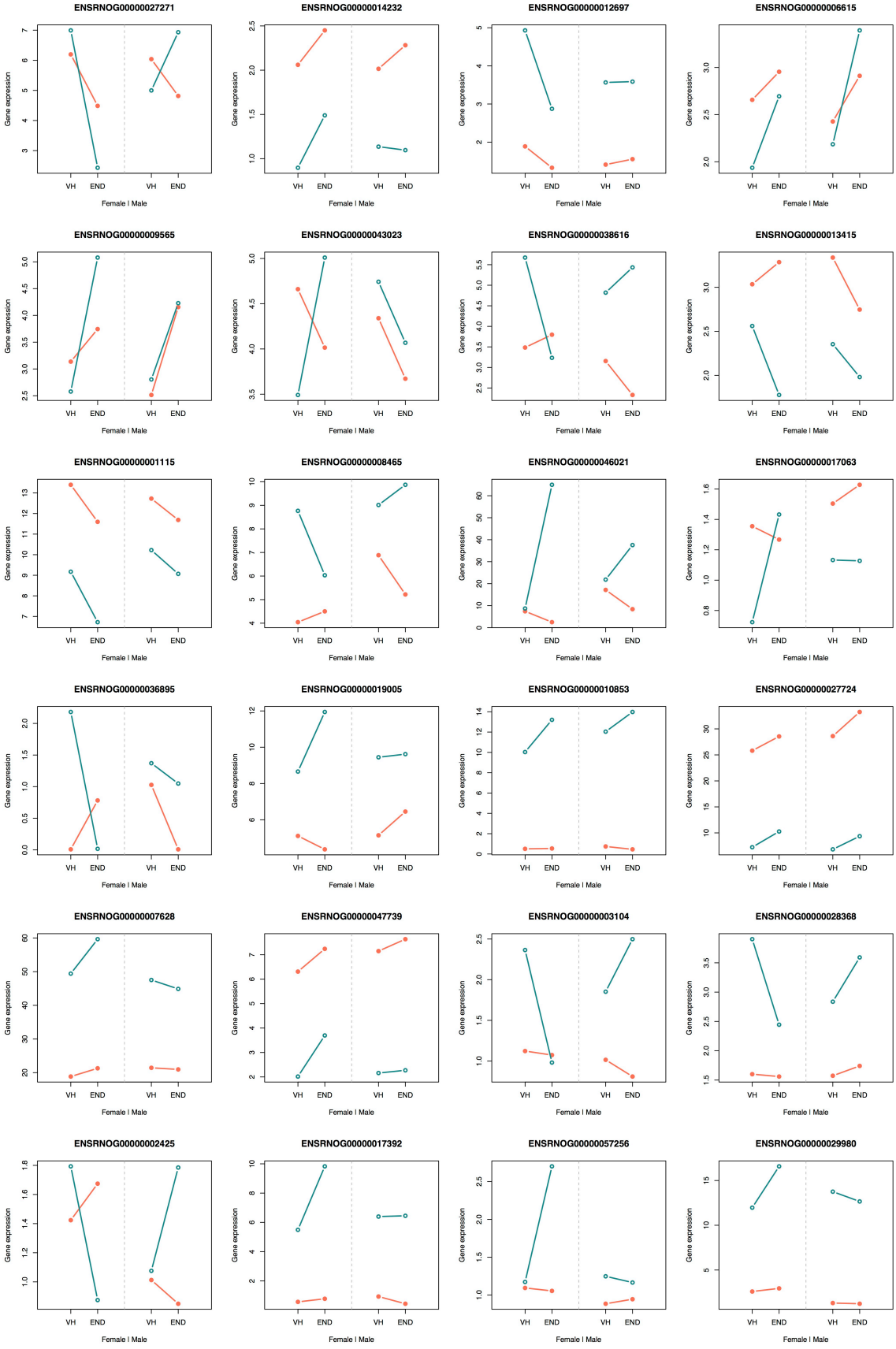
RNA-seq


ENSRNOG00000007407
ENSRNOG00000060130
ENSRNOG00000025768
ENSRNOG00000013340
ENSRNOG00000022372
ENSRNOG00000017279
ENSRNOG00000010352
ENSRNOG00000017693
ENSRNOG00000001047
ENSRNOG00000007291
ENSRNOG00000000098
ENSRNOG00000015858
ENSRNOG00000003714
ENSRNOG00000019221
ENSRNOG00000011071
ENSRNOG00000008415
ENSRNOG00000017434
ENSRNOG00000060728
ENSRNOG00000006970
ENSRNOG00000020421
ENSRNOG00000032492
ENSRNOG00000022764
ENSRNOG00000048861
ENSRNOG00000018467
Gene expression
VH
END
Female | Male

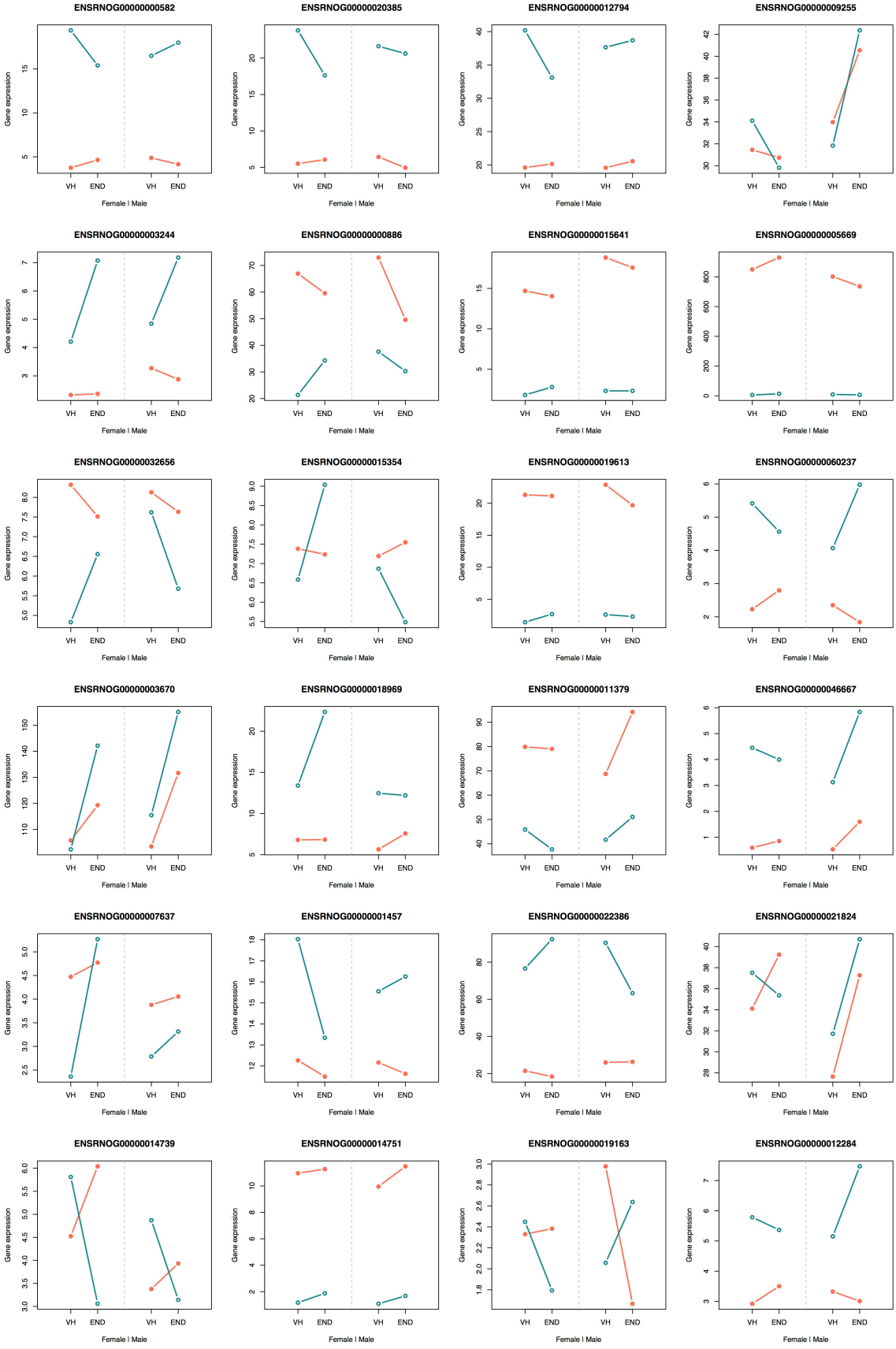
RNA-seq


ENSRNOG00000015695
ENSRNOG00000027229
ENSRNOG00000010994
ENSRNOG00000000956
ENSRNOG00000020322
ENSRNOG00000021787
ENSRNOG00000018851
ENSRNOG00000003443
ENSRNOG00000004145
ENSRNOG00000036677
ENSRNOG00000012521
ENSRNOG00000056457
ENSRNOG00000008960
ENSRNOG00000006806
ENSRNOG00000013171
ENSRNOG00000052564
ENSRNOG00000016112
ENSRNOG00000022932
ENSRNOG00000007003
ENSRNOG00000006723
ENSRNOG00000046452
ENSRNOG00000024798
ENSRNOG00000009505
ENSRNOG00000019693
Gene expression
VH END VH END
Female | Male

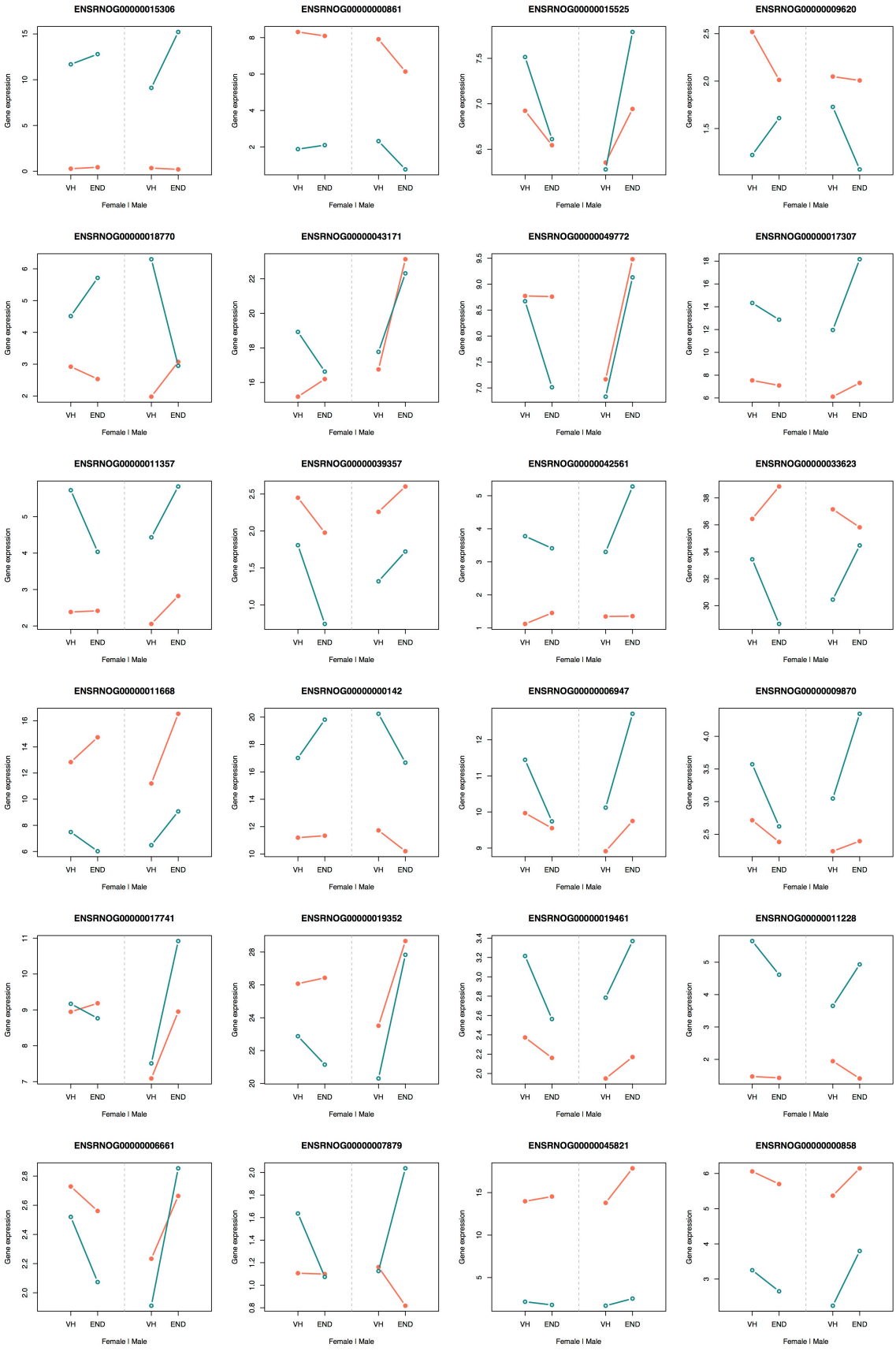
RNA-seq


ENSRNOG00000027271
ENSRNOG00000014232
ENSRNOG00000012697
ENSRNOG00000006615
ENSRNOG00000009565
ENSRNOG00000043023
ENSRNOG00000038616
ENSRNOG00000013415
ENSRNOG00000001115
ENSRNOG00000008465
ENSRNOG00000046021
ENSRNOG00000017063
ENSRNOG00000036895
ENSRNOG00000019005
ENSRNOG00000010853
ENSRNOG00000027724
ENSRNOG00000007628
ENSRNOG00000047739
ENSRNOG00000003104
ENSRNOG00000028368
ENSRNOG00000002425
ENSRNOG00000017392
ENSRNOG00000057256
ENSRNOG00000029980
Gene expression
VH
END
Female | Male

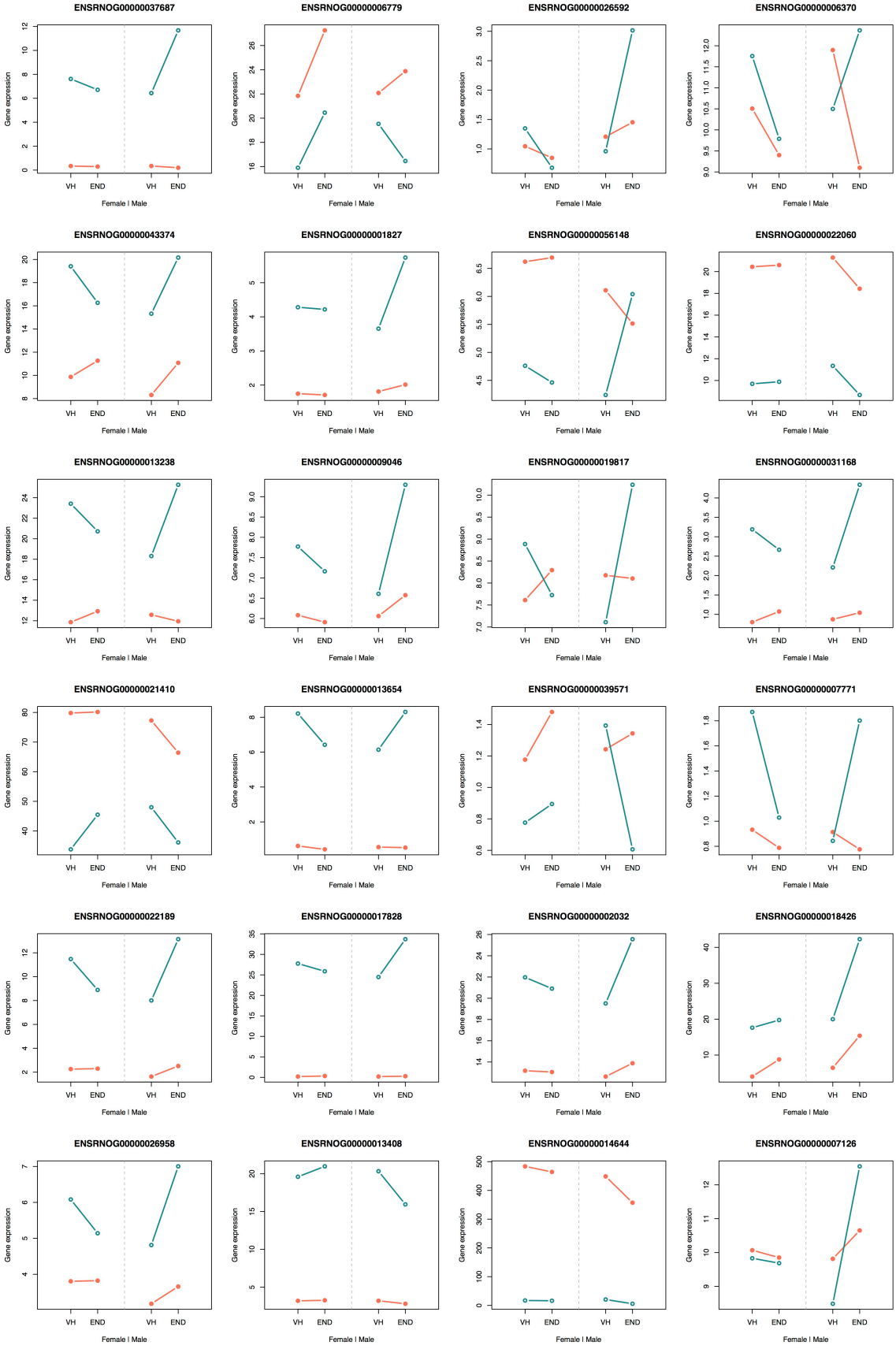
RNA-seq


ENSRNOG00000000582
ENSRNOG00000020385
ENSRNOG00000012794
ENSRNOG00000009255
ENSRNOG00000003244
ENSRNOG00000000886
ENSRNOG00000015641
ENSRNOG00000005669
ENSRNOG00000032656
ENSRNOG00000015354
ENSRNOG00000019613
ENSRNOG00000060237
ENSRNOG00000003670
ENSRNOG00000018969
ENSRNOG00000011379
ENSRNOG00000046667
ENSRNOG00000007637
ENSRNOG00000001457
ENSRNOG00000022386
ENSRNOG00000021824
ENSRNOG00000014739
ENSRNOG00000014751
ENSRNOG00000019163
ENSRNOG00000012284
Gene expression
VH
END
Female | Male

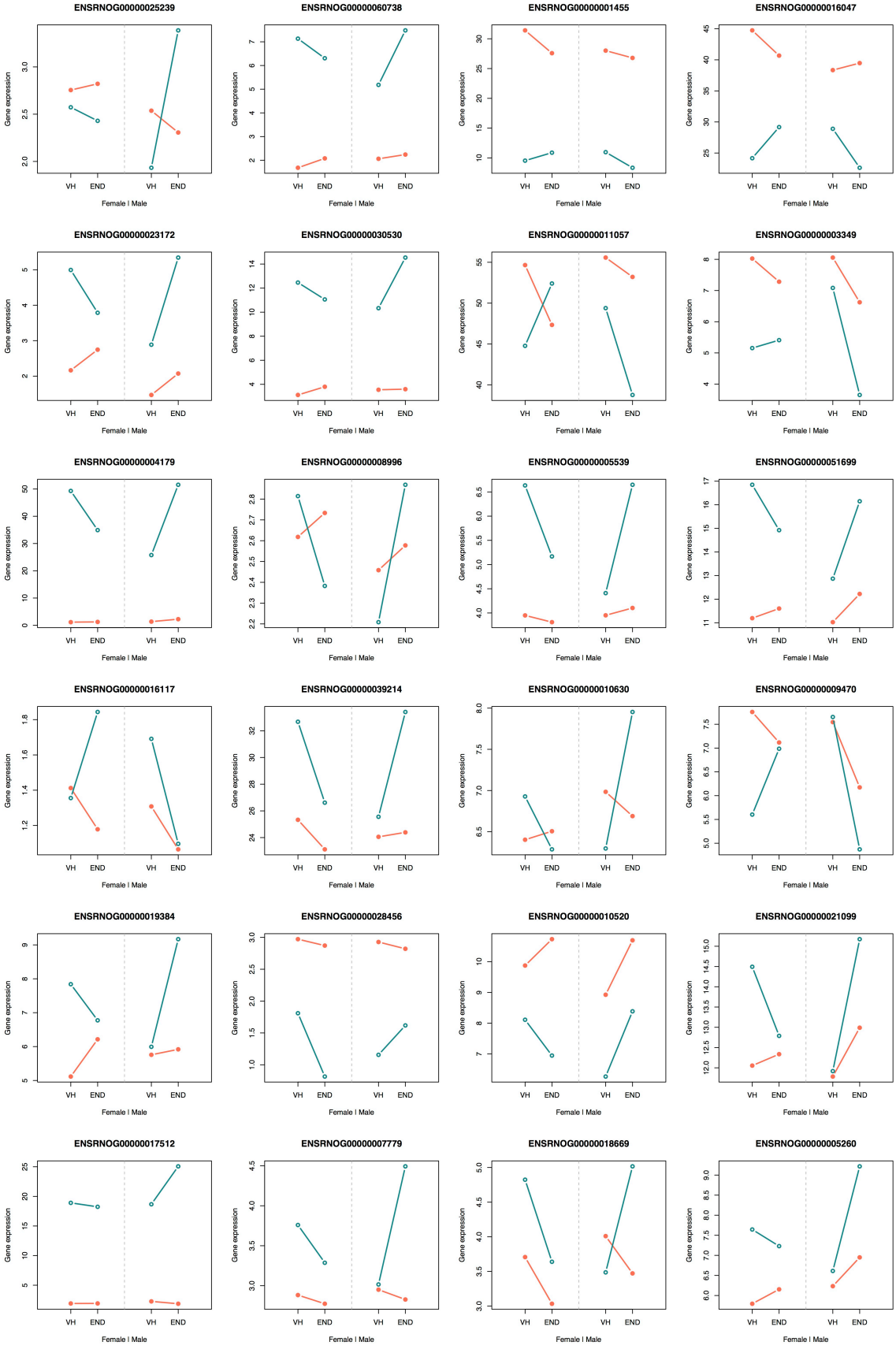
RNA-seq


ENSRNOG00000015306
ENSRNOG00000000861
ENSRNOG00000015525
ENSRNOG00000009620
ENSRNOG00000018770
ENSRNOG00000043171
ENSRNOG00000049772
ENSRNOG00000017307
ENSRNOG00000011357
ENSRNOG00000039357
ENSRNOG00000042561
ENSRNOG00000033623
ENSRNOG00000011668
ENSRNOG00000000142
ENSRNOG00000006947
ENSRNOG00000009870
ENSRNOG00000017741
ENSRNOG00000019352
ENSRNOG00000019461
ENSRNOG00000011228
ENSRNOG00000006661
ENSRNOG00000007879
ENSRNOG00000045821
ENSRNOG00000000858
Gene expression
VH END VH END
Female | Male

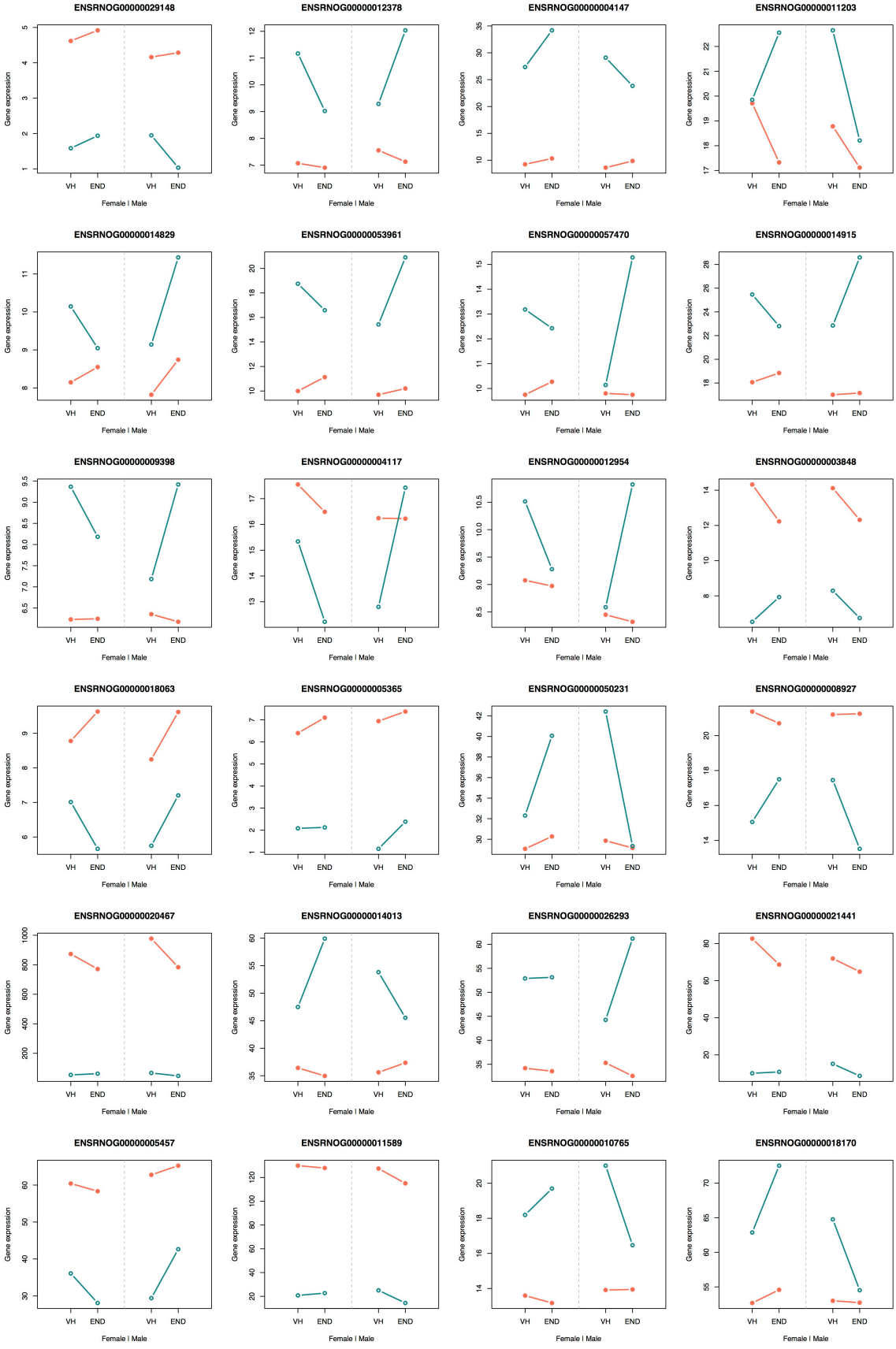
RNA-seq


ENSRNOG00000037687
ENSRNOG00000006779
ENSRNOG00000026592
ENSRNOG00000006370
ENSRNOG00000043374
ENSRNOG00000001827
ENSRNOG00000056148
ENSRNOG00000022060
ENSRNOG00000013238
ENSRNOG00000009046
ENSRNOG00000019817
ENSRNOG00000031168
ENSRNOG00000021410
ENSRNOG00000013654
ENSRNOG00000039571
ENSRNOG00000007771
ENSRNOG00000022189
ENSRNOG00000017828
ENSRNOG00000002032
ENSRNOG00000018426
ENSRNOG00000026958
ENSRNOG00000013408
ENSRNOG00000014644
ENSRNOG00000007126
Gene expression
VH END VH END
Female | Male

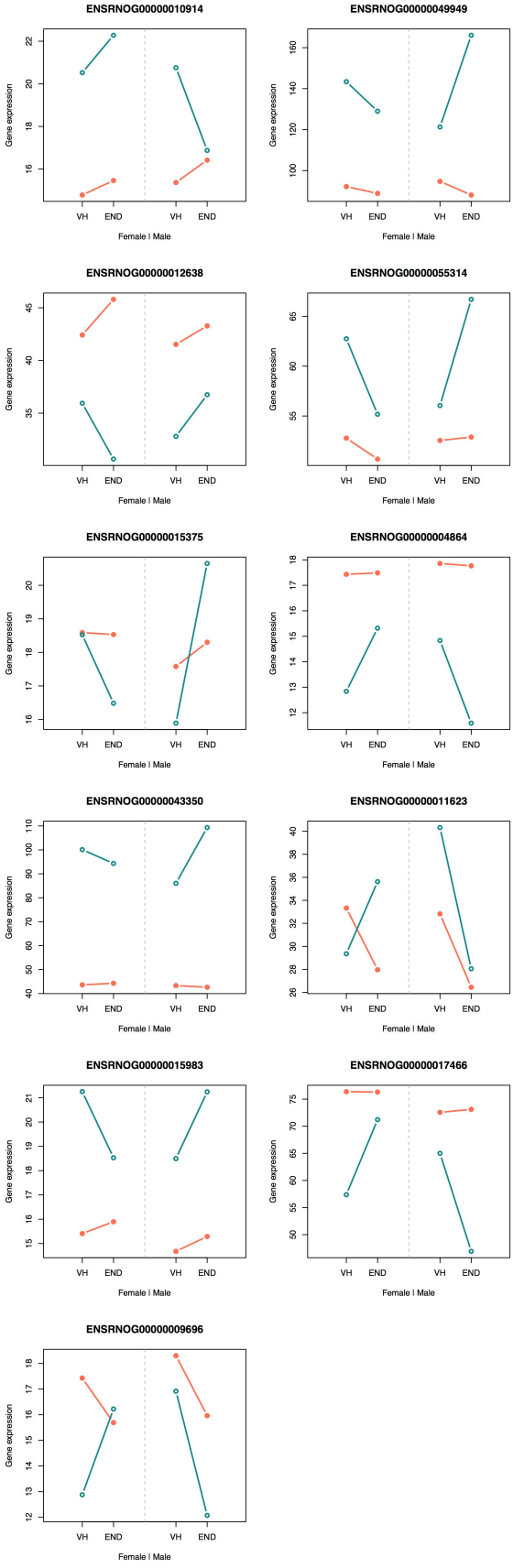
RNA-seq


ENSRNOG00000025239
ENSRNOG00000060738
ENSRNOG00000001455
ENSRNOG00000016047
ENSRNOG00000023172
ENSRNOG00000030530
ENSRNOG00000011057
ENSRNOG00000003349
ENSRNOG00000004179
ENSRNOG00000008996
ENSRNOG00000005539
ENSRNOG00000051699
ENSRNOG00000016117
ENSRNOG00000039214
ENSRNOG00000010630
ENSRNOG00000009470
ENSRNOG00000019384
ENSRNOG00000028456
ENSRNOG00000010520
ENSRNOG00000021099
ENSRNOG00000017512
ENSRNOG00000007779
ENSRNOG00000018669
ENSRNOG00000005260
Gene expression
VH
END
Female | Male

RNA-seq


ENSRNOG00000029148
ENSRNOG00000012378
ENSRNOG00000004147
ENSRNOG00000011203
ENSRNOG00000014829
ENSRNOG00000053961
ENSRNOG00000057470
ENSRNOG00000014915
ENSRNOG00000009398
ENSRNOG00000004117
ENSRNOG00000012954
ENSRNOG00000003848
ENSRNOG00000018063
ENSRNOG00000005365
ENSRNOG00000050231
ENSRNOG00000008927
ENSRNOG00000020467
ENSRNOG00000014013
ENSRNOG00000026293
ENSRNOG00000021441
ENSRNOG00000005457
ENSRNOG00000011589
ENSRNOG00000010765
ENSRNOG00000018170
Gene expression
VH END VH END
Female | Male

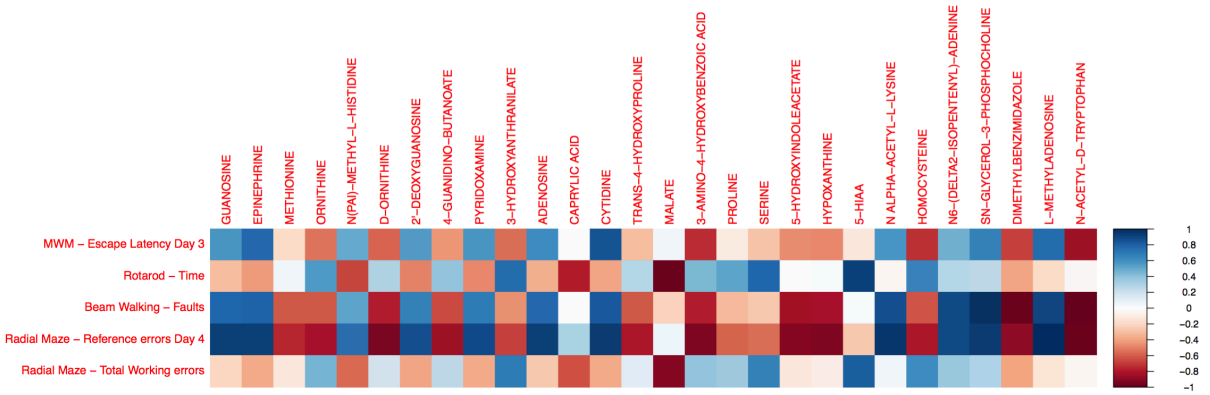
RNA-seq


ENSRNOG00000010914
ENSRNOG00000049949
ENSRNOG00000012638
ENSRNOG00000055314
ENSRNOG00000015375
ENSRNOG00000004864
ENSRNOG00000043350
ENSRNOG00000011623
ENSRNOG00000015983
ENSRNOG00000017466
ENSRNOG00000009696
Gene expression
VH
END
Female | Male

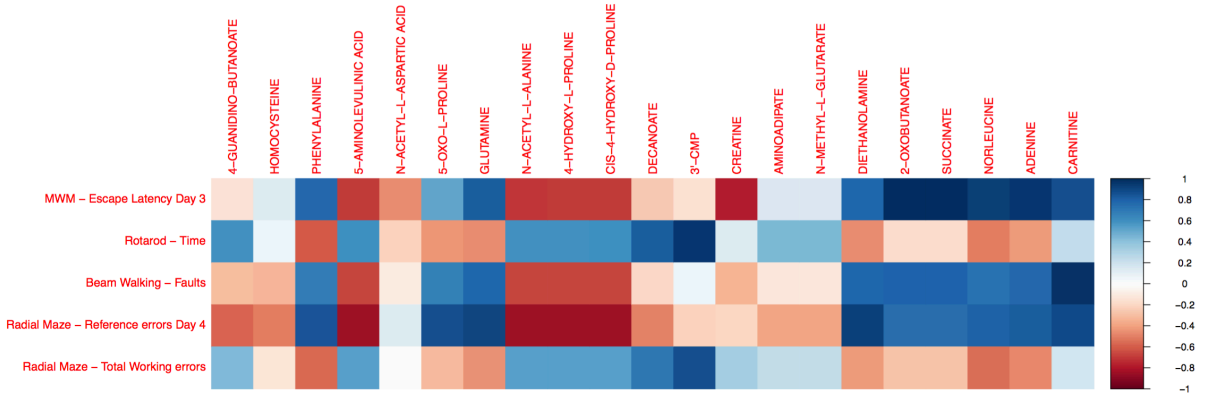
7.2.8. Attachment XIX: Correlation plots

Metabolomics

Cerebellum

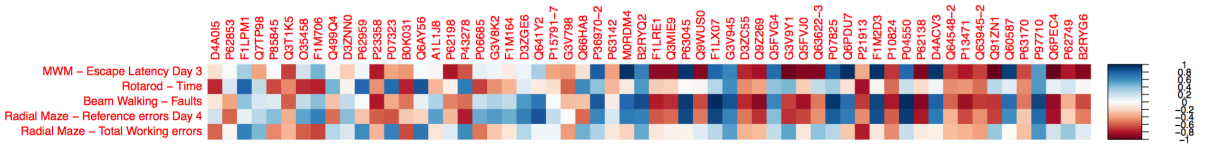


Hippocampus

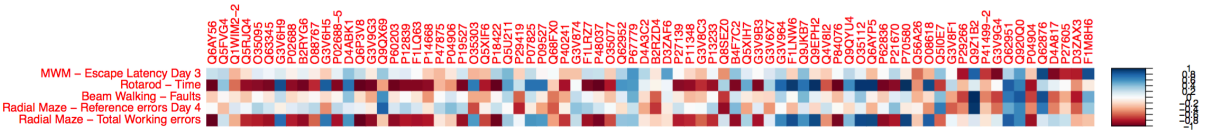


Proteomics

Cerebellum

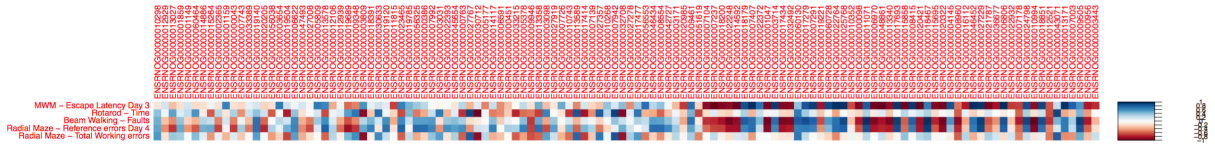


Hippocampus

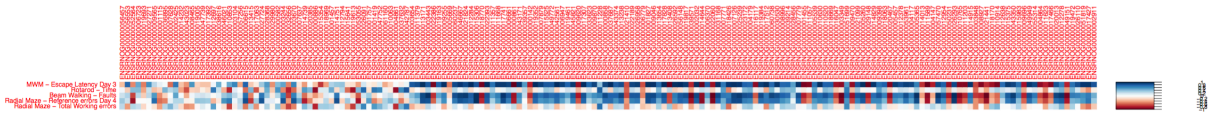


RNA-seq

Cerebellum

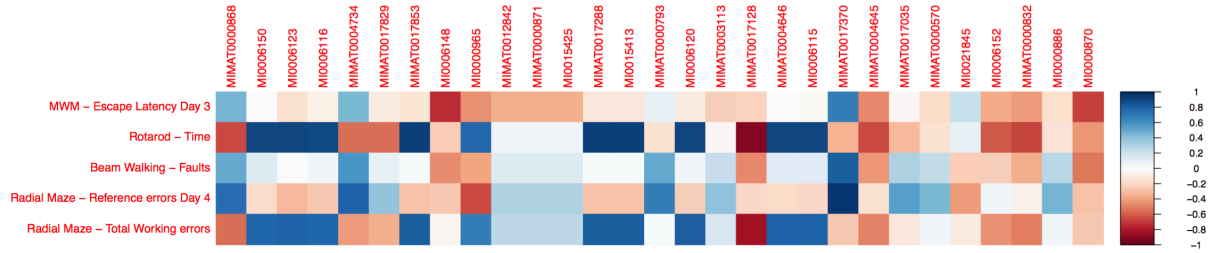


Hippocampus

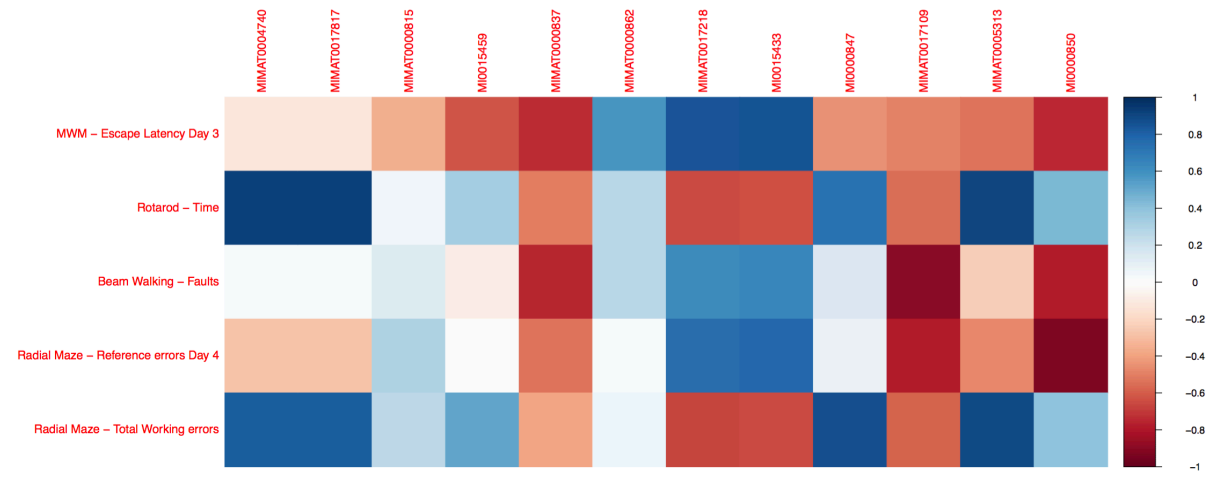


miRNA-seq

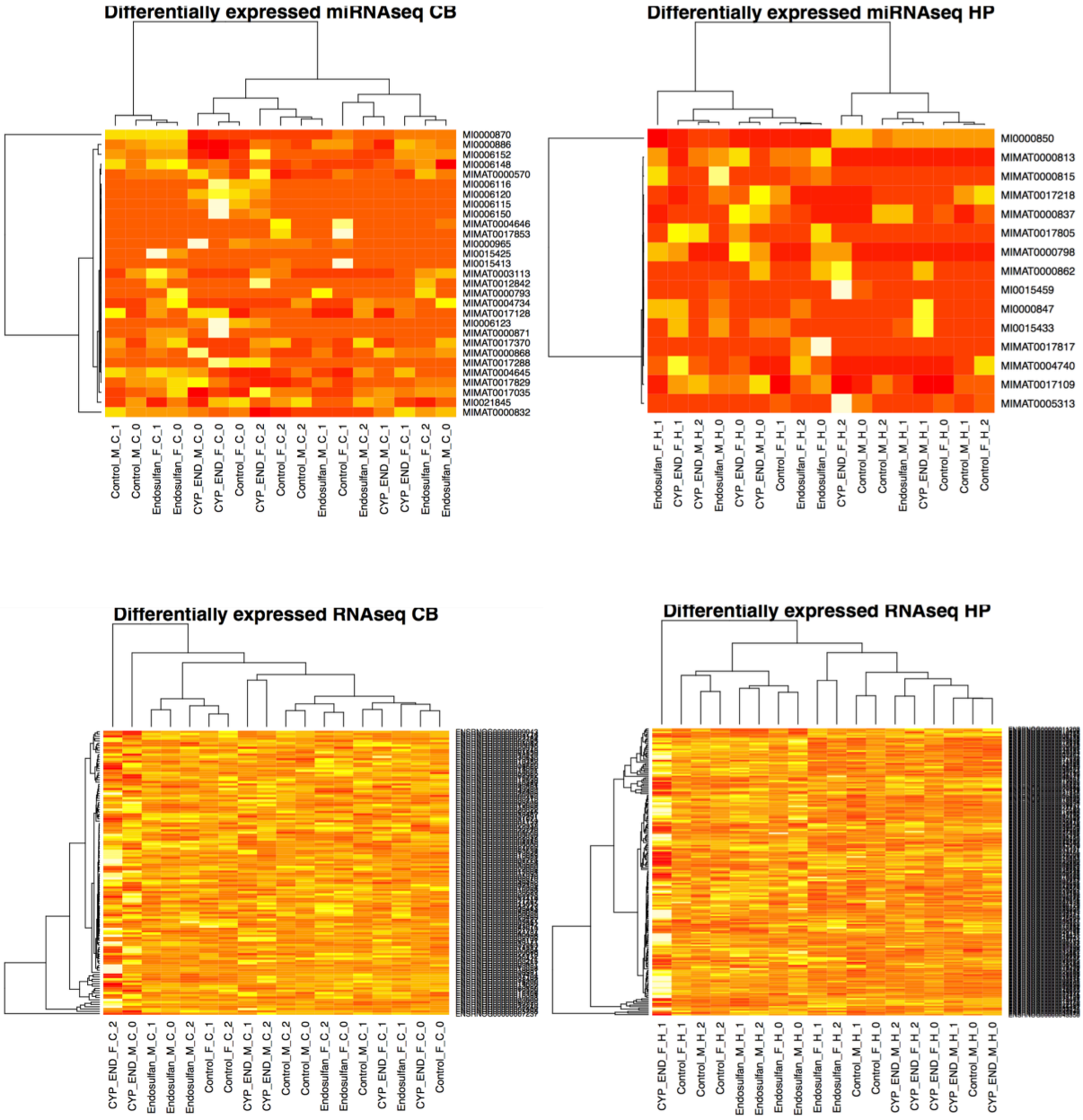
Cerebellum



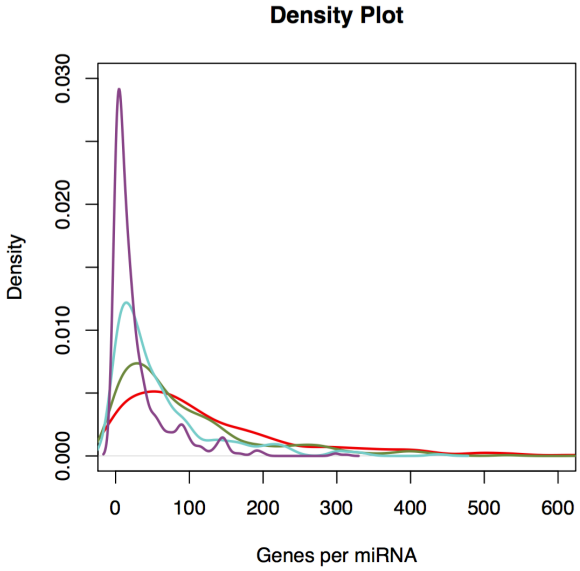
Hippocampus

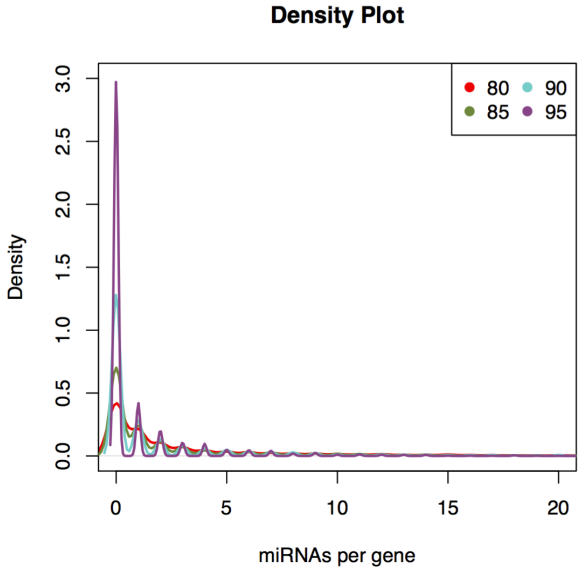


7.2.9. Attachment XX: Heatmaps DE transcriptomics

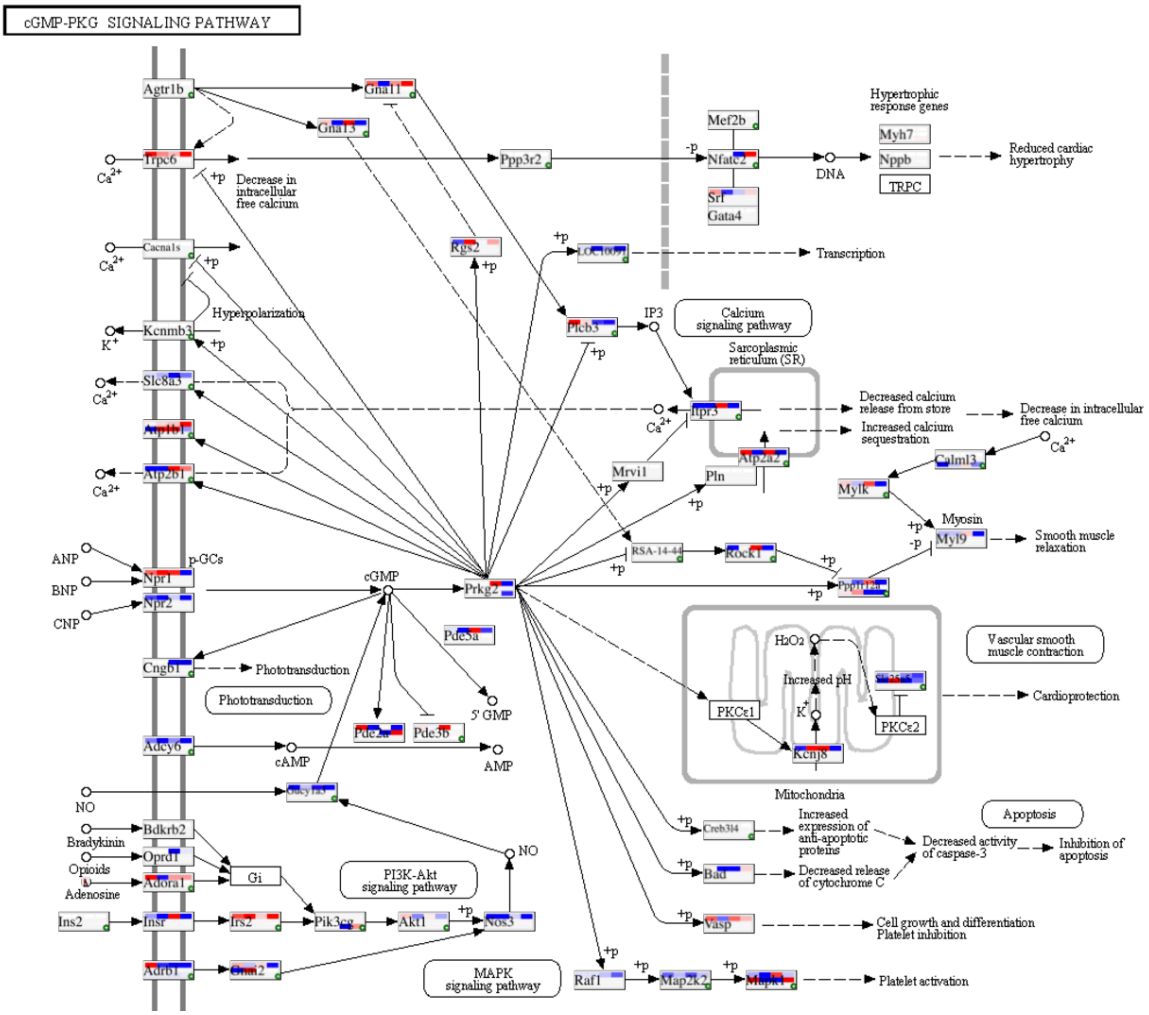


7.2.10. Attachment XXI: miRDB density plots





7.2.11. Attachment XXII: cGMP-PKC signaling pathway



7.2.12. Attachment XXIII: Parkinson's disease pathway