Analysis Techniques Applicable to Time-Limited Gene Expression Data

Dissertation for MAM9060

Author: Nathan Constantine-Cooke (nsc@aber.ac.uk)
Supervisor: Dr. Kim Kenobi (kik10@aber.ac.uk)

September 22, 2019
Version: 1.0 (Release)

This dissertation was submitted as partial fulfilment of a MSc degree in Statistics for Computational Biology (G499)

Declaration of originality

I confirm that:

- This submission is my own work, except where clearly indicated.
- I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
- I have read the regulations on Unacceptable Academic Practice from the University's Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
- In submitting this work I understand and agree to abide by the University's regulations governing these issues.

Consent to share this work

I hereby agree to this dissertation being made available to other students and academic staff of Aberystwyth University.

Acknowledgements

I am grateful to Dr Kim Kenobi whose supervision has been everything a student could ever ask for. I am indebted to Dr David Wilcockson for supplying a transcriptomics dataset and providing insight into the mind of a biologist. I am also grateful to Dr Hannah Dee and Mr Neil Taylor for providing a LaTeX template which has been adapted for this document.

I would like to thank my family for their support in everything I do. I am thankful for the incredible emotional support offered by my friends. In particular, I would like to thank Amy Major, Kayleigh Rippengale and Madeline Stapleton.

Abstract

Despite a growing interest in gene expression studies, the cost of transcriptomics experiments is often expensive. In order to reduce these costs, a researcher may choose to perform their experiment across a time period which is shorter in length than typical gene expression studies (less than 48 hours). As such, it is worthwhile to consider the statistical methods which are applicable to short time-course transcriptomics data- in particular cosinor regression models, cluster analysis and correlation-based techniques. These methods have been formally presented and applied to a transcriptomics dataset obtained from *Talitrus saltator* samples. A novel method for filtering a transcriptomics dataset using t-tests is also presented. The results obtained from applying these methods to the dataset are discussed. Some results provide significant evidence for genes being circadian or circatidal despite the dataset describing 21 hours of activity. The biological mechanisms which may explain the results are discussed: as is the significance of the results to specialists in the field of molecular biology. Opportunities for further research based on either the results of this project or the surrounding literature are presented. The nature of interdisciplinary work involving molecular biology is discussed alongside the future of similar research.

(196 words)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Project Overview

The Aims & Purpose of the Project

Gene activity, alternatively know as *gene expression*, profiling is often incredibly expensive with costs increasing significantly for experiments running for longer time periods[1]. It would therefore be fruitful to consider some of the statistical techniques which can be employed on data which describes gene activity over a short time period to find genes of interest to the scientific community. Many of the techniques usually applied to gene activity data are not applicable to data which only detail activity over a short period as these techniques involve either correlating a large segment of the time course with another large segment from the same time course (autocorrelation) or using Fourier analysis. This project uses data acquired from the marine amphipod *Talitrus saltator* which have been provided by O'Grady et al.[2]. For this dataset, time segments for autocorrelation can not be constructed which are large enough for accurate reliable results to be found. As a result, other techniques will be proposed and discussed. These techniques encompass cosinor models, cluster analysis and building networks based upon correlation.

Finding a *Gene of Interest*

There are many difficulties associated with finding genes of interest to the scientific community: not least due to the lack of a clear definition of what constitutes a *gene of interest*. There are perhaps a few qualities which should be expected from such a gene when analysing its activity. Firstly, this gene would be expected to show a change in activity over time. It would be of additional interest if the time profile of a gene aided in explaining the behaviour of an organism or elucidated a biological process. If a gene is shown to influence the activity of another gene then this should arouse the curiosity of molecular biologists. It would also be interesting if a gene demonstrated a repeating behaviour as this could explain regular biological processes in the subject species.

It is important to consider interaction effects between the genes as many genes influence other genes[3]. However, this potentially introduces significant computational concerns. It could result in every gene in a transcriptome or genome of over 100,000 genes having to be tested for significant interactions with every other gene in the transcriptome or genome. As such, a method of filtering a gene expression dataset which does not require high levels of computational power and is conservative in filtering genes, so as to not lose genes which may yield significant results, must be used.

If a gene has a repeating rhythm with a period of approximately 24 hours or 12.4 hours, then it is usually considered to belong to a special category of gene and is known as a *circadian* or *circatidal* respectively. Circadian and circatidal genes are of special interest to chronobiologists due to their connection to solar and tidal rhythms. As a circadian rhythm can be mathematically expressed as any continuous non-negative function which repeats every 24 hours, there are theoretically an infinite number of functions which describe circadian rhythms. In figure 1.1, it can be seen that each plot demonstrates a very different profile. However, all four genes have been confirmed by biologists as genes which are hugely important to the circadian rhythms in animals. These genes are known as core clock genes and will be discussed further in this chapter.

There is a substantial lack of published research investigating circatidal rhythms. To date, there is no circatidal model organism. However, it has been shown the circadian and circatidal core clocks are independent in *Eurydice pulchra*[4]. Therefore, the circadian and circatidal clocks should be considered as distinct mechanisms.

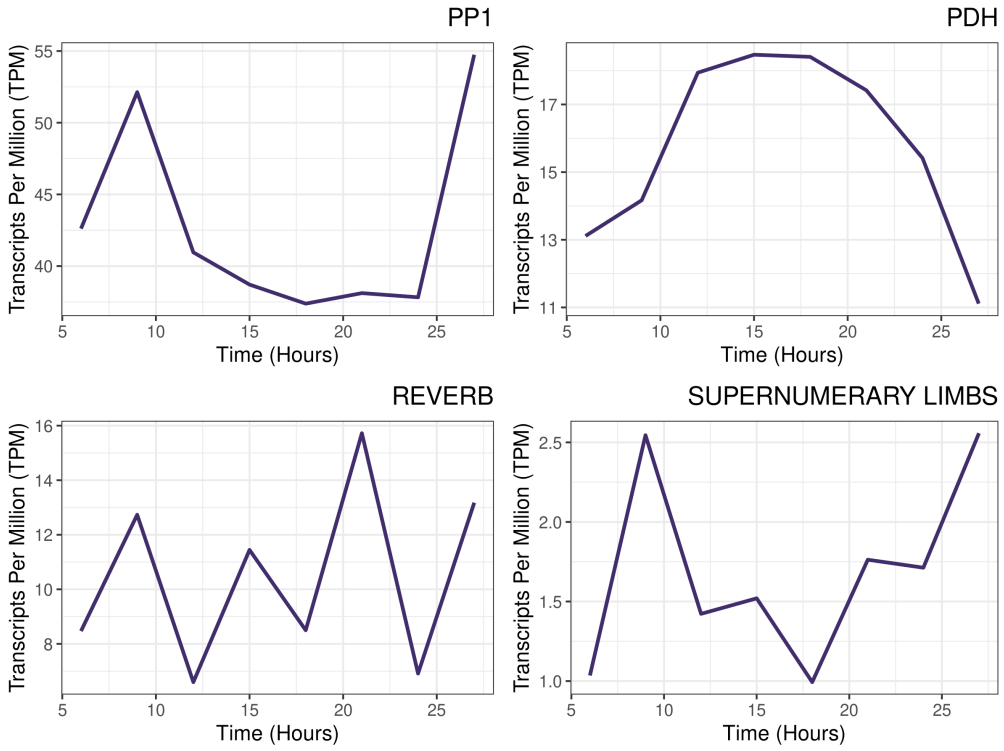


Figure 1.1:
Plots of genes identified as circadian via BLAST (see *Bioinformatics Techniques*)

Produced using R[5] and the `ggplot2` package[6]

Bioinformatics Techniques

It would be beneficial to discuss how genes of interest are typically detected by biologists. In the bioscience fields, such a gene is usually identified by using the Basic Local Alignment Search Tool (BLAST) or by using a derivation of this algorithm. BLAST has been described as "the single most important piece of software in the field of bioinformatics" due to its speed and reliability[7]. BLAST involves testing a gene for similarities with known genes stored in a database. The bases of DNA (DeoxyriboNucleic Acid) or RNA (RiboNucleic Acid), when arranged in a sequence, define a gene. These sequences can be searched against a database using the BLAST algorithm to find genes with closely related sequences. The bases of DNA are adenine, thymine, guanine and cytosine. RNA shares these bases with the exception of thymine being replaced by uracil. Usually, if a gene is found to be similar to a gene of interest then it is determined that the former is also of interest. An important limitation of BLAST is it relies upon the genes in the desired search results to have already been identified as having the characteristics of interest to the searcher. In the context of circadian and circatidal genes, these genes have usually been found to be rhythmic with the period of interest by using autocorrelation techniques.

The Dataset

It should also be noted gene expression measurements are very noisy which necessitates replicates when measuring samples. The transcriptome used for this study involves 156,768 genes with gene expression measured at 8 time points with 4 replicates. The activity measurements begin at hour 6 of the experiment and end at hour 27. Thus, the data span 21 hours and is therefore shorter

than a full circadian rhythm. It should be noted that one of the replicates for hour 18 is zero for all genes the measurements for this replicate have accordingly been removed from the dataset. After all of the genes which show no expression are removed, as is customary, the dataset describes 91,311 genes.

The `CircadianTools` R package

As part of the research undertaken for this project, an R package, `CircadianTools` has been developed. To date, this package includes over 60 functions and 4,400 lines of code. There are three key benefits for presenting the R code written for this project as a package:

1. Reproducibility. Reliance upon variables in the global environment of an R session is minimised in this approach. It should be less likely that code provided for reproducibility purposes will rely on a variable in the global environment which has not been defined in the code. A user should also be able to reproduce any results detailed in this document by entering only a few lines of code into the R console.
2. Ease of code distribution. When `CircadianTools` is installed, any R packages which are dependencies for `CircadianTools` on will be downloaded from CRAN (the Comprehensive R Archive Network) and then installed (with the exception of dependencies exclusively available on the Bioconductor repository).
3. Robust documentation. In addition to all of the code being commented, additional documentation is available for each function which provides descriptions of the functions themselves as well as, arguments and returned values. Examples for every function in `CircadianTools` are also available.

`CircadianTools` uses the multithreaded nature of modern CPUs to increase the speed of calculations by parallelism through the use of the `foreach`[8] and `doParallel`[9] packages. The current implementation creates worker processes via forking. Whilst forking reduces RAM usage when compared to alternative methods, this approach also limits multithreaded processing to only Linux and Mac OS systems.

`CircadianTools` can be found at `https://github.com/nathansam/CircadianTools` where all of the code for the functions used by `CircadianTools` can be found. More detailed instructions for installing `CircadianTools`, including installing from local files, can be found in Appendix A.

Benchmark System

All of the computations carried out for this research were done on a system with an AMD Ryzen 1600X 6 core processor with a 4GHz base clock speed and 24GB of 3000MHz DDR4 RAM running Manjaro Linux. Any processing times quoted in this document are for this system unless otherwise specified. Turbo core was disabled to provide consistent results and thermal throttling was not observed. As a result, CPU clock speed was not adjusted in response to CPU temperatures. R version 3.6.1 was used for all computations.

Code Appendices

Code appendices for all of the results chapters can be found at the end of this document. The appendices show how the functions in `CircadianTools` were used to produce the results described in each chapter. The appendices can be used to easily reproduce the results.

RNA and Transcriptomics

RNA

Ribonucleic acid is an essential building block for all life. It is generally accepted that RNA existed before proteins and was the sole form of genetic material before the contemporary dominant method of using DNA[10]. As previously mentioned, RNA shares three bases with DNA: adenine, guanine and cytosine. The final base of RNA is uracil which replaces the thymine found in DNA. RNA is single-stranded in contrast with the double-stranded structure of DNA and is involved in the coding, decoding, regulation and expression of genes.

The Central Dogma of Molecular Biology

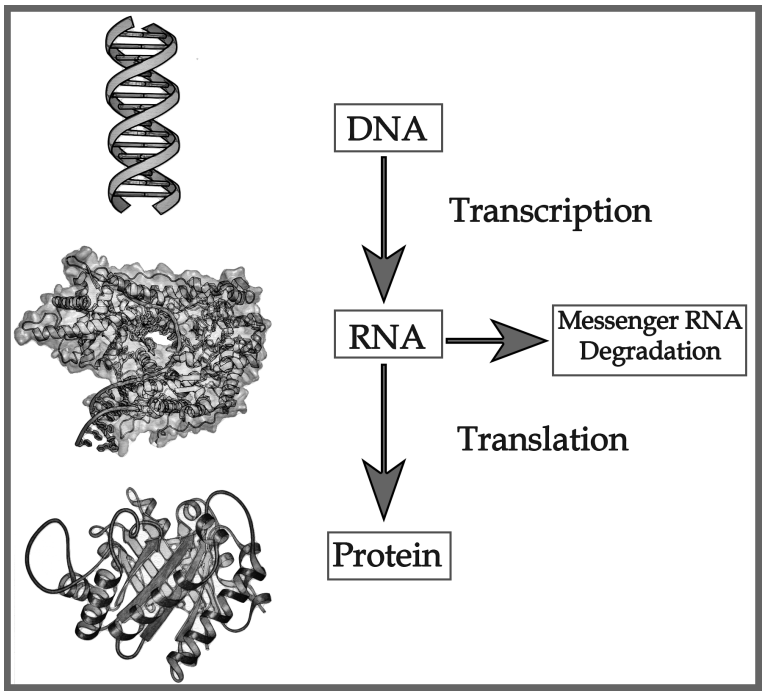


Figure 1.2:
Visualisation of the central dogma of Molecular Biology

Attribution: T7 RNA polymerase ©Litvinanna / CC BY-SA 4.0. Modified &
TriosePhosphateIsomerase Ribbon pastel photo ©Jane Richardson / CC BY-A 3.0 UNPORTED. Modified

The central dogma of molecular biology explains how genetic information flows from DNA to Proteins via RNA[11]. DNA creates messenger RNA (mRNA) through the process of transcription which involves copying DNA bases into RNA bases: converting thymine into uracil in the process. Information is carried by mRNA to the ribosomes which then create proteins through the process of translation. The quantification and analysis of mRNA and RNA is called transcriptomics. Transcriptomics often focuses on exploring the expression profile of genes and is typically measured in a $\log_2$ transformation of the transcript count over time.

RNA-Seq

Advantages and Disadvantages of RNA-Seq

The currently preferred method of gene expression profiling is called RNA-Seq[12] and was used to obtain the transcriptomics data of *Talitrus saltator* used in this study. This next-generation sequencing high-throughput technique has replaced the previous method, microarrays, due to a few significant advantages:

- RNA-Seq can be used for de novo transcriptome assembly[13]. This means a transcriptome, the set of all RNA molecules in a cell, can be assembled without the genomic sequence for the organism previously being known. This is significant as it allows the transcriptomes of non-model species, such as *Talitrus saltator*, to be assembled: massively increasing the number of organisms scientists can investigate the transcriptome of.
- It has been shown RNA-Seq provides more accurate readings for the number of transcripts than microarrays[14]. RNA-Seq has better dynamic range than microarray-based alternatives and is considerably more accurate when assessing a low abundance of transcripts.
- RNA-Seq also requires a smaller sample size than microarrays which has rendered single-cell transcriptomics a reality. Single-cell transcriptomics allows questions related to the stochastic nature of gene expression to be answered and has particular importance in neuroscience[15].

RNA-Seq reduces the difficulty of carrying out a transcriptomics study but challenges still remain. Financial expenditure remains a serious concern[1]: as is consistency across individual readings. Whilst the number of readings can be increased to improve reliability, this increases cost. Of course, cost can be reduced by decreasing the number of replicates at the expense of reliability. An alternative to this problem is to carry out the experiment across a shorter time period: as is the case in the work by O'Grady et al.[2].

How is Gene Expression Data Produced When Using RNA-Seq?

The process of acquiring activity profiles of genes from biological sequences via RNA-Seq is somewhat complicated. Firstly, the RNAs in the sample are fragmented and are reverse transcribed into complimentary DNAs (cDNAs)[16]. This reverse transcription takes place by using reverse transcriptase: an enzyme. The cDNA is amplified and next generation sequencing is then performed. This generates millions of reads which are mapped to a reference genome or transcriptome. Reads are mapped to the most likely gene for each read. However, there are biological processes which often complicate the mapping process. Single nucleotide polymorphisms, which change a single nucleotide in an organism, may result in a read being assigned the the wrong gene. Reads can also contain small insertions or deletions which further complicates mapping reads. Situations where reads have multiple genes which they can be mapped to also frequently occur. Multiple algorithms have been developed which attempt to remedy these problems. After all of the reads have been mapped, the quantity of reads mapped to each gene are used to quantify the expression level for the gene. This information is often referred to as counts. It should be noted the count of reads aligned to each gene is typically given relative to the total number of reads : such as transcripts per million (TPM).

Tatlitrus saltator

An Introduction to *Talitrus saltator*

The subject of the experiment which generated the data used in this study is the amphipod crustacean, *Tatlitrus saltator*: commonly known as a sand hopper. This common name is due to its proclivity to hop, without control of direction, when threatened. They can be found in abundance across the beaches of western Europe and its diet typically consists of partially decayed seaweed[17].

Figure 1.3:
Photograph of *T. Saltator*

Attribution: ©Hans Hillewaert / CC BY-SA 4.0.

Rhythmic Mechanisms

T. saltator is of interest in this study due to the internal clock mechanisms found in the organism which is demonstrated by its rhythmic behaviour. An example of this behaviour is found in their burrowing activity. During the day, *T. saltator* burrows into sand. Usually, the burrow collapses in the process which means light cannot reach the crustacean[18]. However, *T. saltator* will still leave the burrow at night, despite the lack of a method to detect the light level outside the burrow, and will journey down the beach as the tide retreats to feed on the seaweed left behind.

Another documented example of rhythmic mechanisms can be found in the form of compass mechanisms. *T. saltator* is able to use the sun or moon to navigate: demonstrating circadian and circalunar mechanisms respectively. The circalunar mechanism has been proven to originate in the antennae by an experiment which surgically removed the antennae of subjects[19]. Amputated subjects no longer travelled towards the moon at night and instead journeyed towards the brightest light source which heavily implies the subjects no longer knew where the moon should be positioned in relation to the time of the day.

As a result of these mechanisms, *T. saltator* has been the subject of many research projects.

Despite the amount of research carried out on *T. Saltator* however, there is still a reported phenomenon in which groups of *T. Saltator* synchronise clock mechanisms. This behaviour has been neither proven nor disproven[18].

T. saltator behaviour seems to be controlled by light-dark cycles more so than most other isopods. Breeding seasons are typically dictated by temperature in isopods whereas the *T.saltator* breeding season begins once daily sunlight exceeds 14 hours[20, 17]. Mating occurs during the nightly beach migration and broods of around 14 members are typical. Juveniles cannot burrow for protection from both predators and drying out and instead live in seaweed for until they can burrow.

Relevant Mechanisms in Molecular Biology

The Importance of Understanding the Context of Data

It is essential for a statistician to have an understanding of the nature of the data which they are working with. For those working with biological data, such as transcriptomics, this understanding should be extended to include the biological mechanisms which underlie the data. As such, understanding the results generated for this project requires knowledge of the mechanisms which drive and regulate gene expression. Furthermore, awareness of the systems which control circadian behaviour is beneficial as circadian genes, as previously noted, make excellent candidates for genes of interest to the scientific community.

Gene Expression Regulation

Regulation by Transcription

The process of transcription, previously described when discussing the central dogma of molecular biology, is where most gene-expression is regulated.[21]. Expression is regulated by either activating or repressing the transcription of particular genes. Transcription is usually controlled by gene promoters, gene enhancers or transcription factors.

A gene promoter is a region of DNA which initiates the transcription of a specific genes whilst a gene enhancer increases the rate of transcription. The latter are involved in the assembly of the pre-initiation complex which is required for the transcription of protein-coding genes[22]. Both gene promoters and gene enhancers interact with transcription factors (TF). Transcription factors are proteins which bind to specific DNA sequences which are called transcription factor binding sites[23]. A TF can bind to multiple genes and either activate or repress these genes; a TF is able to activate a gene by binding to a promoter site and attracting RNA polymerase to the binding site. The increase in RNA polymerase increases the effectiveness of the promoter. A TF can repress a gene by competing with another TF at the same binding site which leads to less efficient binding of RNA polymerase. Transcription factors are often found by biologists by a DNA mobility shift assay which attempts to determine the potential for a gene to be transcribed[24].

Post-Transcriptional Gene Regulation

Whilst most gene regulation is performed at the transcription level, protein phosphorylation can also regulate genes[25]. During protein phosphorylation, a member of of the phosphoryl group (a chemical ion containing phosphorous and oxygen) attaches to a protein molecule. This process activates, deactivates, or changes the function of the protein. Phosphorylation is driven by a kinase which is an enzyme which acts as a catalyst for the phosphorylation process.

The Circadian Clock

Properties of Circadian Rhythms

Circadian rhythms were discovered at least as early as 1729 by Jean-Jacques d'Ortous de Mairan who observed a plant's leaves, which would open daily in order to process light, would still open

even in a dark room[26]. There are now three established properties required for a rhythm to be circadian:

1. All circadian rhythms should be internal rhythms which synchronise with at least one external time-cue. This synchronisation usually called entrainment and the external time-cue is typically called a zeitgeber. The most commonly discussed zeitgeber is the light-dark cycle. During an experiment testing circadian rhythms, it is therefore important to account for any potential zeitgebers.
2. The second property of a circadian rhythm is, when zeitgebers are absent (also known as a free running), the period of the rhythm should be very close to 24 hours.
3. The final property expected of a circadian rhythm is the period of the rhythm should largely be unaffected by temperature[27]. Such a behaviour is atypical for biological processes and the mechanism which compensates for temperature is not known.

Despite there being specific criteria needed to be met for a rhythm to be circadian, it has been found that nearly all daily rhythms are circadian[28]. This is useful when performing statistical analysis as this means verifying if a gene with a 24 hour rhythm has all the properties of a circadian rhythm is not as important as it would be without this finding.

Core Clock Proteins

Transcription factors which drive rhythmic expression in genes are known as core clock proteins. Core clock proteins influence a great many more genes which express circadian behaviour as a result. These gene are known as clock controlled genes. Some core clock genes are kinases and are able to self regulate through feedback loops.

The first core clock protein was found by Konopka and Benzer in 1971[29]. The pair noticed certain behaviour, which would ordinarily follow a 24 hour rhythm in wild type *Drosophila melanogaster* (a fruit fly), was radically different in mutant *D. melanogaster*. Knopka and Benzer were able to identify the mutated gene which was named PERIOD. However, a gene-based approach was not commonly accepted in the chronobiology community when the research was published and their work was not extended until many years later. However, their findings would result in *D. melanogaster* becoming a model organism for circadian studies in later years.

In 1984, two groups, working independently, showed the circadian behaviour in mutant *D. melanogaster* could be restored by modifying the PERIOD gene[30, 31]. Three members from these groups, Hall, Rosbash and Young would later receive the 2017 Nobel Prize in Physiology and Medicine for their discoveries of molecular mechanisms which control circadian rhythms: highlighting the importance of understanding the circadian clock in physiology and medicine[32].

In 1990, Hardin, Hall and Rosbash described the feedback loop system in PERIOD[33]. They argued the activity of the PERIOD protein resulted in the rhythmic cycling of its own RNA. They showed the gene encodes protein which accumulates during the night and degrades during the day.

The second core clock protein, TIMELESS, was found in 1994[34]. Mutations of the timeless gene were found to produce arrhythmic behaviour in *D. melanogaster*. A mechanism which involved both the PERIOD and TIMELESS genes was suggested as it was observed TIMELESS altered the circadian oscillations of PERIOD RNA. However, this mechanism was not described in-depth.

Four years later, the DOUBLE-TIME core clock protein was discovered[35]. DOUBLE-TIME was found to regulate PERIOD protein accumulation which also affected TIMELESS.

The finding that mutations of TIMELESS led to the circadian clock being abolished led to attempts to create models of the interaction between TIMELESS and PERIOD where it was assumed PERIOD proteins were rapidly phosphorylated by DOUBLE-TIME and then degraded[36].

CRY and DCRY were discovered in 1998 and 1999 respectively[37, 38]. Both of these core clock proteins were found to be involved in light entrainment and their discovery elucidated how entrainment to zeitgebers operated at the molecular level for the first time.

Chapter 2

Cosinor Models

Introduction to Cosinor Models

Preamble

A sensible approach to finding genes with a repeating rhythm is to fit periodic linear models to gene expression data and test if the models provide a good fit for any of the genes. One of the advantages of fitting such models is the period of the linear function can be set to a period of interest: such as the circadian period of 24 hours or the circatidal period of 12.4 hours. Sine and cosine functions are periodic and frequently describe natural behaviour which led to early developments in the fields of chronopharmacy[39]. They are therefore a natural suggestion for the functions to use as a basis for periodic linear models in this context. We can fit models with such a basis by using single-component cosinor models[40].

A key strength of cosinor models is they allow genes which are likely to be circadian to be found without needing data across a 48 hour or even a 24 hour period. However, an obvious disadvantage of using cosinor models is only genes with a sinusoidal activity profile will be found as circadian. This is a substantial issue as many circadian genes do not have such an activity profile. Furthermore a cosinor model also does not account for possible gene interactions.

Formal Definitions

The regression model for a single-component cosinor model is given by

$$Y_t = M + A\cos\left(\frac{2\pi t}{\tau} + \phi\right) + e_t, \tag{1}$$

where Y_t is a prediction at time t, M is the rhythm adjusted mean, A is the amplitude, τ is the period, ϕ is the acrophase (the time between $t = 0$ and the first peak of the model), and e_t is the error term. When τ is known then the equation can be written as

$$Y_t = M + A\left(cos(\phi)cos\left(\frac{2\pi t}{\tau}\right) - sin(\phi)sin\left(\frac{2\pi t}{\tau}\right)\right) + e_t, \tag{2}$$

Cosinor models are regressed using the least squares method, thus the parameters of the model are chosen to minimise the residual sum of squares (RSS).

$$RSS = \sum_i^N \left(Y_i - \left(\hat{M} + \hat{A}\cos\left(\frac{2\pi t}{\hat{\tau}} + \hat{\phi}\right)\right)\right)^2, \tag{3}$$

Where N is the total number of observations. This calculation requires an assumption of equal standard deviations between residuals in order for it to be valid.

The fit of a cosinor model can be assessed through the use of an F-test where the F statistic can be calculated using the RSS and MSS (Mean Sum of Squares).

$$F = \frac{\frac{MSS}{2}}{\frac{RSS}{N-3}}, \tag{4}$$

Where $MSS = \sum_i \left(\hat{Y}_i - \bar{Y}\right)^2$. The degrees of freedom for this statistic are 2 and N-3, thus the null hypothesis stating there is no rhythm in the model (the amplitude is 0) is rejected at the α significance level if $F > F_{1-\alpha}(2, N-3)$.

The Application of Single-Component Cosinor Models to the *Talitrus saltator* Dataset

Cosinor models were fitted to each gene in the dataset and p-values from F-tests were calculated. Models of both 24 and 12.4 hour periods were generated in order to test for circadian and circatidal qualities. Bonferroni p-value corrections were applied due to the large quantity of genes being tested (91,311). The data was also transformed so the first activity reading was for hour 0 instead of hour 6.

24H Period Results

Cosinor models were shown to not be computationally intensive. Fitting cosinor models to 91,311 genes and carrying out F-tests on these models required only 4.3 minutes when using the `CosinorAnalysis` function in the `CircadianTools` package. 15 genes were found to be significant at the 5% level when cosinor models with a period of 24 hours were fitted. These genes were used as search queries with BLAST. All of the most significant results were genes predicted to belong to *Hyalella azteca*: an amphipod crustacean commonly found in North America. These genes, the adjusted p-values and BLAST results can be found in table 2.1. Four of these significant genes can be seen in figure 2.1. It can be seen these genes have a circadian time profile. This attribute can be found in all of the significant genes.

Sample ID	Adjusted P-value	Protein found via BLAST
comp99801_c1_seq1	2.246×10^{-6}	uncharacterized protein LOC108681637
comp80445_c1_seq1	3.783×10^{-5}	N/A
comp89211_c0_seq2	1.470×10^{-4}	N/A
comp96806_c0_seq1	4.191×10^{-4}	la-related protein 6-like
comp98714_c0_seq2	6.070×10^{-4}	N/A
comp100026_c0_seq2	$6.175 \times 10{-}4$	myb-like protein A
comp102333_c0_seq8	0.006	N/A
comp102333_c0_seq2	0.003	uncharacterized protein LOC108665706
comp23718_c0_seq1	0.006	myb-like protein A
comp102333_c0_seq21	0.007	uncharacterized protein LOC108665706
comp101772_c1_seq2	0.010	uncharacterized protein LOC108679659
comp71855_c0_seq1	0.011	N/A
comp98599_c2_seq1	0.013	N/A
comp606_c0_seq1	0.017	N/A
comp102333_c0_seq9	0.042	uncharacterized protein LOC108665706

Table 2.1:
The genes which were found to be significant at the 5% level when fitted with cosinor models with 24 hour periods. All Proteins found via BLAST are predicted sequences for *Hyalella azteca*. N/A indicates no significant sequences were found.

12.4H Period Results

No gene fitted with a cosinor model was found to be significant from F-test results after the Bonferroni correction was applied. Alternatives to the Bonferroni correction were considered however none of the adjustments supported by the `p.adjust` function in base R returned any p-values below even 0.95. The 25 highest ranking genes were plotted and assessed for circatidal characteristics by

eye. Of these genes, eight were deemed likely to be circatidal. Four high-ranking genes can be seen in figure 2.2. Two of the genes in the figure were deemed to likely be circatidal whilst the remaining two were not. Fitting cosinor models with this period and then investigating the highest ranked genes is still a method which can be used to consider a number of genes which is feasible for human judgement. However, a more robust statistical approach should be greatly desired. It is perplexing statistically significant evidence has been produced which argues some specific genes have 24 hour period rhythmic expression using data across 21 hours, yet no statistically significant evidence for genes with a 12.4 hour rhythm was found. Before conducting the analysis, it was assumed evidence for the latter would be more likely to be found due to having over one full oscillation in the data. It should be noted, for both 12.4H and 24H period cosinor models, the most significant models were for genes which show low levels of gene activity. Table 2.2 shows the eight genes deemed to be circatidal ,their unadjusted p-values and most significant BLAST results if applicable. Similar to the 24 hour period results, all significant genes found via BLAST were predicted for *Hyalella azteca*. The p-values have been reported without applying a Bonferroni adjustment as once this is applied, all p-values are rounded to one within R.

Sample ID	Unadjusted P-value	Protein found via BLAST
comp97780_c0_seq11	$1 \times 10^{-}4$	N/A
comp80084_c0_seq1	$4 \times 10^{-}4$	N/A
comp37769_c0_seq1	$4 \times 10^{-}4$	N/A
comp5046_c0_seq2	$8 \times 10^{-}4$	N/A
comp93816_c1_seq3	0.002	A-agglutinin anchorage subunit-like
comp97124_c1_seq3	0.003	uncharacterized protein LOC108670690
comp45880_c0_seq1	0.004	N/A
comp93114_c0_seq1	0.005	flocculation protein FLO11-like

Table 2.2:
All of the genes which were found to be significant at the 5% level when fitted with cosinor models with periods of 12.4 hours. All Proteins found via BLAST are predicted sequences for *Hyalella azteca*. N/A indicates no significant sequences were found.

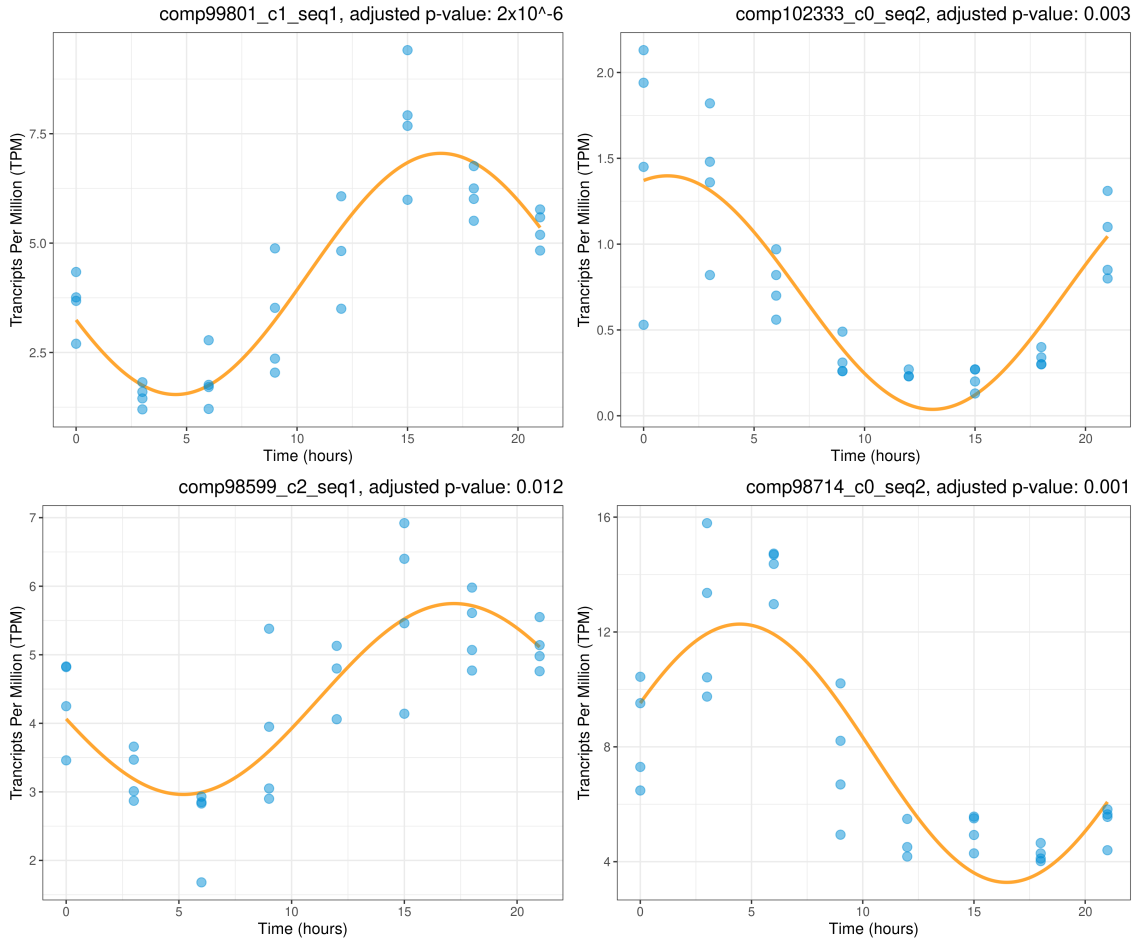


Figure 2.1:
Four genes found to be significant at the 5% significance level via F-tests on cosinor models with 24H periods.

Produced using the `cosinor`[41] and `cosinor2`[42] packages.

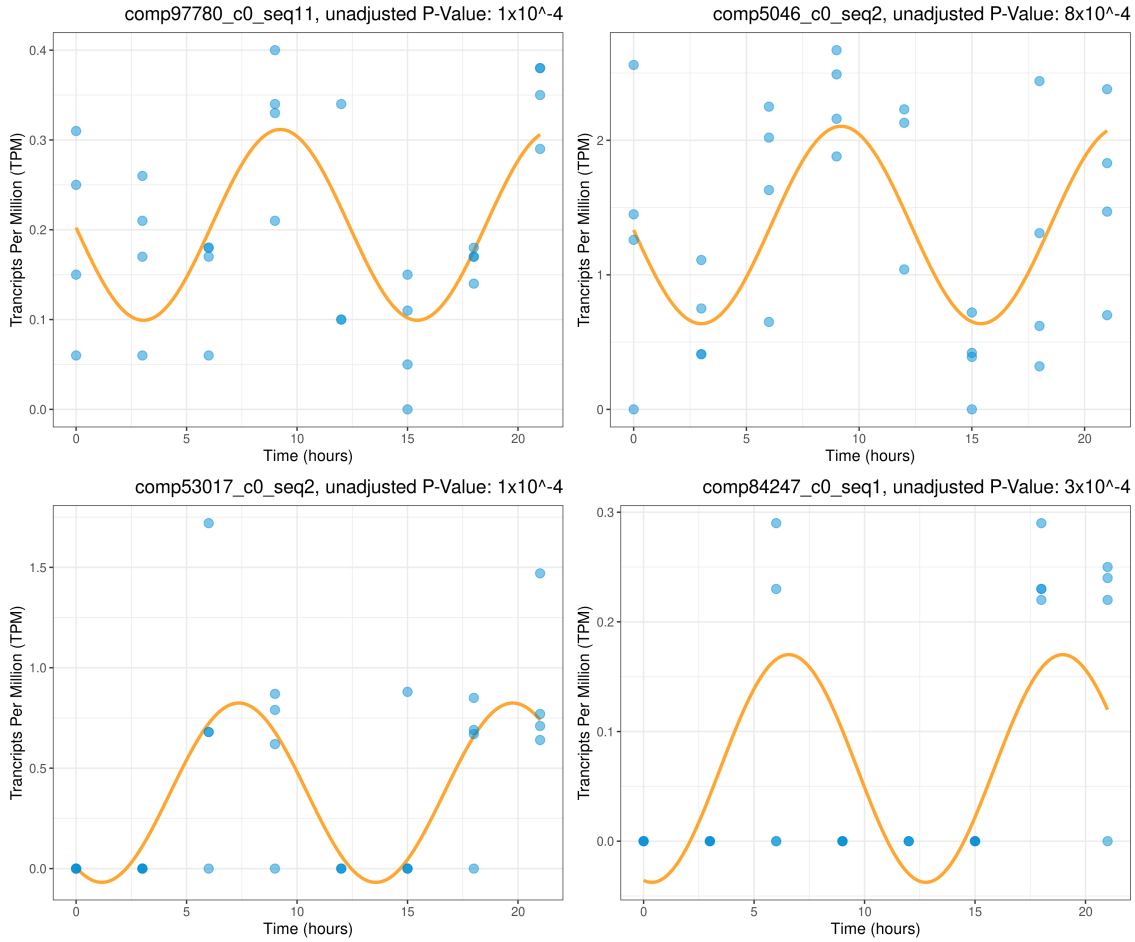


Figure 2.2:
Four genes ranked highly by p-value when fitted with cosinor models with 12.4H periods. The top plots were deemed to possibly be circatidal whilst the remaining plots were not.

Cosinor Models with Additional Terms

Motivation

Plotting the residuals for the significant 24 hour period cosinor models showed sinusoidal profiles in some cases which implies there are genes which also have a sinusoidal component with a different period. In particular, these residuals seem to have a period of approximately 12 hours. This implication led to a hypothesis being formed stating introducing additional sine and cosine terms with a different period results in a better fit for some of the genes with significant 24 hour period cosinor models. This hypothesis was tested via ANOVA. Formal definitions for ANOVA can be found in the next chapter. Of the 15 genes found significant, three genes were selected which appear to show sinusoidal residuals, comp29718_c0_seq1, comp71855_c0_seq1, and comp98714_c0_seq2 and three more were chosen which were deemed to not have residuals with this characteristic, comp98599_c2_seq1, comp101772_c1_seq2, and comp102333_c0_seq9. Plots of the residuals for these genes can be seen in figure 2.3.

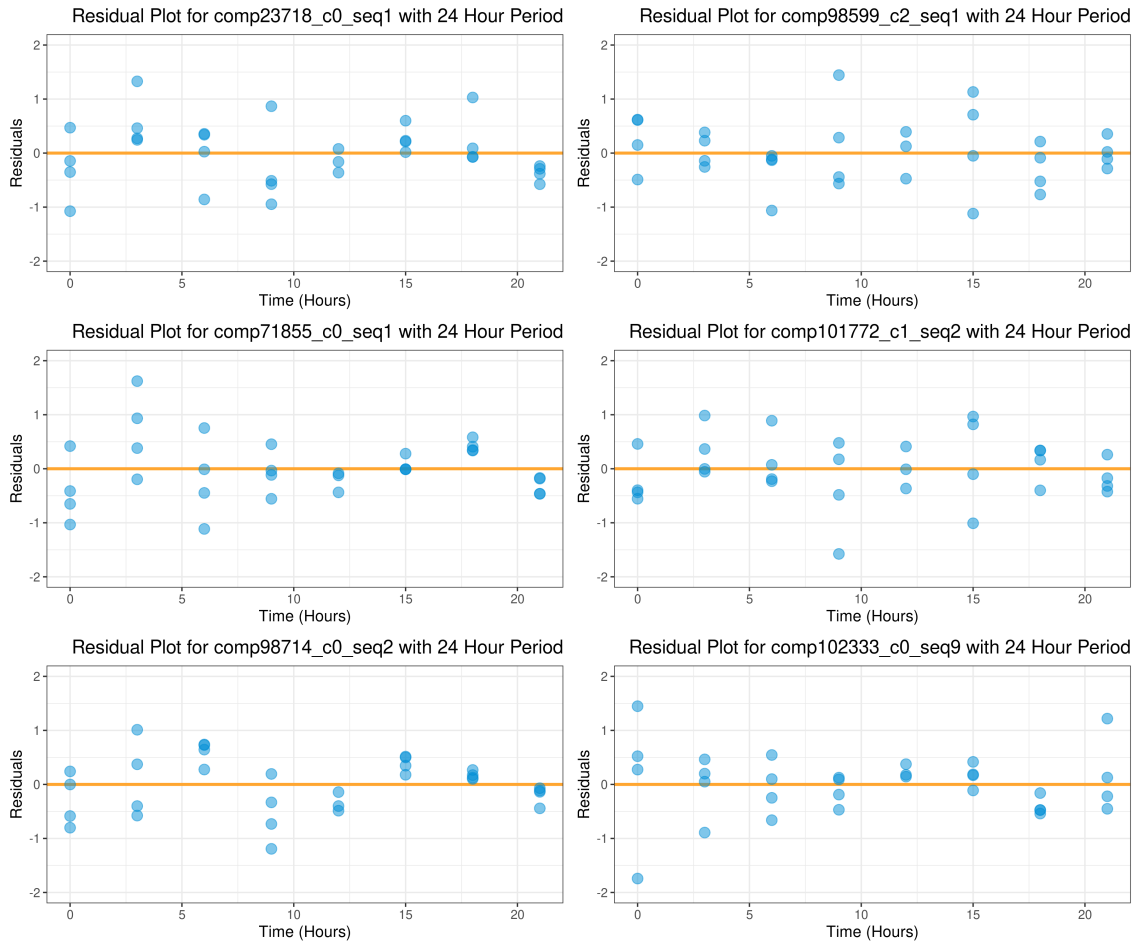


Figure 2.3:
Residuals of circadian cosinor models of six selected genes. The plots in the left column were judged to still have sinusoidal characteristics whilst the plots in the right column were not.

Extending a Cosinor Model with Additional Terms

From Equation (2), a derivation of a single-component cosinor model can be obtained by setting $B = A\cos(\phi)$ and $C = -A\sin(\phi)$ (noting ϕ is a constant). This derivation produces the following equation:

$$Y_t = M + B\cos\left(\frac{2\pi t}{\tau}\right) + C\sin\left(\frac{2\pi t}{\tau}\right) + e_t, \tag{5}$$

This model can easily be extended to include terms with a second period:

$$Y_t = M + B_1\cos\left(\frac{2\pi t}{\tau_1}\right) + C_1\sin\left(\frac{2\pi t}{\tau_1}\right) + B_2\cos\left(\frac{2\pi t}{\tau_2}\right) + C_2\sin\left(\frac{2\pi t}{\tau_2}\right) + e_t, \tag{6}$$

Application of Extended Cosinor Models to the Selected Genes

Simple cosinor models, with 24 hour periods, and extended cosinor models, with additional 12.4 hour period terms, were fitted to the six selected genes. The term *simple cosinor model* will be used instead of the *single-component cosinor* so as to not arise confusion due to the model having both sine and cosine components. ANOVA was used to test if the inclusion of the additional terms results in a significant improvement in fit.

Comp23718_c0_seq1

For the simple cosinor model, both the coefficient for the cosine and the coefficient for the sine term were found to be significantly different from zero at the 5% level. (p-values of 1×10^{-7} and 0.001 respectively).

Both of the 24 hour period terms were found to still be significant at the 5% level when the extended cosinor model was considered (p-value of 1×10^{-8} for the cosine term and 2×10^{-4} for the sine term). Whilst the cosine term with a 12.4 hour period was found to not be significant (p-value of 0.75), the sine term with this period was found to be significant (p-value of 0.003).

Carrying out ANOVA on the two linear models results in a p-value of 0.011 which indicates including the additional circatidal terms results in a significantly better fit.

Comp71855_c0_seq1

For the simple model, the coefficients for both the cosine term and the sine term were found to be significantly different from zero (p-values of 4×10^{-8} and 0.043 respectively).

The extended cosinor model for this sample produces comparable ANOVA results to the ANOVA results for the extended cosinor model for comp23718_c0_seq1. Both cosine and sine terms with 24 hour periods were found to be significant at the 5% level (p-values of 1×10^{-8} and 0.029 respectively). The cosine term with a circatidal period was not significant (p-value of 0.379) whereas the sine term with a circatidal period was found to be significant (p-value = 0.018)

Carrying out ANOVA between the simple and extended models produces a p-value of 0.041 which denotes the extended model provides a significantly better fit.

Comp98714_c0_seq2

Carrying out ANOVA tests on the simple cosinor model for this sample once again showed the sine and cosine terms in the simple cosinor model to both be significant at the 5% level (p-values of 0.002 and 6×10^{-9} respectively).

For the extended cosinor model, all four terms were found to be significant at the 5% level. The p-values for the circadian cosine and sine terms are 1×10^{-10} and 0.002 respectively. The p-values are 0.039 and 0.001 for the circatidal cosine and sine terms.

ANOVA between the two cosinor models provides strong evidence for the extended cosinor model providing a significantly better fit as a p-value of 0.001 is obtained from the test.

Samples Without Sinusoidal Residuals

For the three genes which did not appear to show sinusoidal residuals, none of the circatidal terms in the extended models were found to be significant at the 5% level. Additionally, the ANOVA tests between simple and extended cosinor models did not find the extended models to provide a significantly better fit at the 5% significance level (p- values of 0.321, 0.167 and 0.457 for comp98599_c2_seq1, comp101772_c1_seq2 and comp102333_c0_seq9 respectively)

Conclusion of results

A summary of the results of the ANOVA tests between the simple and extended cosinor models can be found in table 2.3

Out of the three samples which show sinusoidal residuals, the extended cosinor models of the samples have been shown to provide significantly better fit over simple cosinor models. This suggests the three genes are being influenced by both circadian and circatidal mechanisms.

The approach of using extended cosinor models has enabled circatidal genes to be found in a statistically robust approach instead of simply judging genes to be significant by visual inspection: as was the case for single-component circatidal cosinor models. In this particular case, only genes which were already found to have significant circadian cosinor models were used. It would possibly be fruitful to test all genes in a transcriptomics dataset for significant extended cosinor models with circadian and circatidal periods. Cases where there is a significant circadian term and a significant circatidal term may be found.

Sample ID	P-value	Sinusoidal Residuals
comp23718_c0_seq1	0.01	✓
comp71855_c0_seq1	0.04	✓
comp98714_c0_seq2	0.001	✓
comp98599_c2_seq1	0.321	✗
comp101772_c1_seq2	0.167	✗
comp102333_c0_seq9	0.457	✗

Table 2.3:
Results of ANOVA tests between the simple and extended cosinor models for six selected samples.

Chapter 3

Filtering

ANOVA filtering

Preamble

In order for many statistical techniques to be feasible on modern computer hardware, a transcriptomics dataset must first be reduced in size by filtering out samples which show very little activity or only demonstrate noise.

The most common technique to filter a transcriptomics dataset is to use ANOVA (ANalysis Of VAriance). Each gene is is tested to decide if its activity at any time point is significantly different from the mean activity of the gene. If this activity is not found to be significantly different from the mean for at least one time point then the gene is filtered out of the dataset. This means that the order of the time points is not a factor when using ANOVA for filtering.

Formal Definitions

The ANOVA model can be written as:

$$Y_{ij} = \mu + \tau_i + e_{ij}, \tag{1}$$

where μ is the overall mean, τ_i is the group effect for group i and e_{ij} is the error term. The hypothesis proposed is

$$\begin{aligned} H_0 &: \quad \mu_1 = \mu_2 = \quad \ldots \quad = \mu_k \\ H_1 &: \qquad\quad \mu_i \neq \quad \mu_j, \end{aligned} \tag{2}$$

for some positive integers $i, j \in [1, k]$ where μ_i is the mean for group i. ANOVA requires three key assumptions:

1. The samples are independent.
2. The data are normally distributed.
3. Each group has the same variance.

Application of ANOVA Filtering to the *T. saltator* Dataset

Each gene of the *T. saltator* dataset was first centred so the overall mean for each gene is zero. Initially the dataset was filtered by ANOVA using a p-value cut-off of 5%. However this results in reducing the dataset to 11186 genes which is still far too large for may modern computational techniques. It is of interest to know how many genes identified as core clock genes by BLAST are in the reduced dataset. Five of the 18 core clock genes identified by BLAST were included in this filtered dataset. These genes have sample IDs comp97405_c0_seq1, comp99101_c0_seq1, comp102279_c0_seq7, comp102609_c0_seq1, comp939723_c0_seq1 which correspond to PP1, CASEIN KINASE 2β, PERIOD, CRY2 and CYCLE respectively. As there were too many genes in the filtered dataset, a significance level of 2.5% was used which reduced the dataset to 7124 genes which is a more feasible amount of genes for advanced computational techniques. Only three of the genes identified as circadian by BLAST can be found in the reduced dataset: PP1, CASEIN KINASE 2β, and PERIOD.

ANOVA filtering was found to be computational efficient: taking 84 seconds to filter the dataset.

Filtering via T-Tests

Preamble

An alternative method to filtering via ANOVA, based on two-sample t-tests, will now be outlined. This method is not found in the literature. Two-sample t-tests are used to decide if a change between group means is significant. In the proposed method of filtering, two-tailed t-tests are used to find if the change in mean between consecutive time points is significant. Each group is the replicate measurements for a single time point in this method. The sign of the t-statistic calculated indicates if the change in mean is positive or negative. The proposed method involves including only genes with at least a certain number of significant changes and almost as many positive significant changes as negative significant changes in the filtered dataset. The reasoning for the stipulation of there being a minimum number of significant changes is to ensure that the gene activity changes significantly and the gene is therefore possibly of interest. The stipulation of having close to as many significant positive changes as negative is due to the expectation that a rhythmic gene would often be expected to have approximately as many significant positive changes as significant negative changes in order to have the same activity level after one full period. This is possibly not the case for all rhythmic genes however. For instance, if the profile of a true circadian gene has lots of small changes in activity level which are all decreasing and then one or two huge increases in the opposite direction, then this gene would likely be filtered out via the t-tests method. However, this may describe relatively few genes of interest.

Formal Definitions

The null, H_0, and alternative, H_1, hypothesises for a two sample two-tailed t-test are

$$\begin{aligned} H_0 &: \mu_A = \mu_B \\ H_1 &: \mu_A \neq \mu_B, \end{aligned} \tag{3}$$

where μ_i is the mean for group i . The test statistic for a t-test is calculated by

$$T = \frac{\bar{Y_A} - \bar{Y_B}}{\sqrt{\frac{{S_A}^2}{N_A} + \frac{{S_B}^2}{N_B}}}, \tag{4}$$

where $\bar{Y_i}$ is the sample mean for group i, ${S_i}^2$ is the sample standard deviation of group i and N_i is the number of elements in group i. If $|T| > t_{1-\frac{\alpha}{2}, N_1+N_2-2}$ then T is found to be significant at the α significance level. Carrying out t-tests requires three assumptions:

1. Observations must be independent
2. The distribution of observations should be approximately normal
3. The standard deviations of the groups are equal

Application of T-Test Filtering to the *T. saltator* Dataset

Filtering the dataset by only allowing genes which showed at least two significant changes and a maximum difference of one between the number of positive and negative significant changes reduced the dataset to 6,294 genes. The significance level used was 5%. The results were interesting as many genes in the reduced dataset could possibly be circadian due to the start and end points

of the time series being approximately the same. Of course, without a 24 hour time course, it is impossible to confirm if these genes indeed have the same activity levels after approximately 24 hours. A selection of the plots in the filtered dataset can be seen in figure 3.1. Only two of the 18 known core clock genes were included in the dataset reduced by using t-tests: CRY2 and CYCLE. Comparing the genes in the dataset reduced by ANOVA with genes in this reduced dataset resulted in the discovery of the two datasets sharing 3,097 genes: just under half of the 6,294 genes in the t-test filtered dataset. Filtering by t-tests was found to be slightly quicker than filtering by ANOVA: taking 80 seconds to filter the dataset.

Parameter Experimentation

As filtering by t-tests involves three parameters which can be adjusted, the effect of varying the parameters was investigated. The three parameters are: the significance level required for a change between time points to be found significant (p), the minimum significant changes needed for a gene to not be filtered out (c) and the maximum difference allowed between the significant positive and negative changes for a gene to not be filtered out (d). The parameters were experimented with by fixing two of the parameters and then varying the third.

Firstly, the value of p was varied whilst c was fixed to two and d was fixed to one. Table 3.1 shows the results of this experimentation. It can be seen reducing the value of p considerably reduces the quantity of genes in the reduced dataset: as is to be expected. This table provides evidence for a significance level of 5% to be suitable for reducing the dataset to a reasonable size.

p	Number of genes
0.05	6294
0.025	2643
0.01	805

Table 3.1:
Results of experimenting with the significance level when using the t-test filtering method.

The minimum number of significant changes was varied after fixing $p = 0.05$ and fixing $d = 1$. Table 3.2 details the results found when varying the parameter from one to five in increments of one. Setting the parameter to one resulted in 22,614 genes to be included in the filtered dataset: too many genes for some computational techniques. Setting the parameter to two resulted in a dataset with 6,294 genes being produced. Setting the parameter to three produced a dataset with 1175 genes which is fewer genes than is typically desired. As such, it is sensible to argue the optimal value for this parameter is two.

c	Number of genes
1	22614
2	6294
3	1175
4	259
5	55

Table 3.2:
Results of experimenting with the minimum number of significant changes when using the t-test filtering method.

The final parameter, the maximum difference between the number of significant positive changes and significant changes was varied. The results can be found in table 3.3. Setting $d = 0$, so that the number of significant positive changes must be equal to the number of significant negative changes, resulted in 4724 genes to be included in the reduced database. However it was deemed

the expectation for a gene of interest to have exactly as many significant positive changes as negative changes was too strict. Changing the parameter from one to two resulted in 1,003 genes to be included in the reduced dataset. However, increasing the parameter from two to three resulted in only 15 more genes being added to the reduced dataset. Increasing d further did not change the number of genes in the filtered dataset. This implies there are no genes where the difference between the number of significant positive and negative changes is four or more.

d	Number of genes
0	4724
1	6294
2	7297
3	7312
4	7312
5	7312

Table 3.3:
Results of experimenting with the maximum difference between significant positive and negative changes when using the t-test filtering method.

In conclusion, an argument has been made for the optimal parameters for t-test filtering with the *T.saltator* dataset being $p = 0.05$, $c = 2$, and $d = 1$. Attention should be drawn to the specificity of these findings to this particular dataset. The results are likely to be closely related to the number of time-points in the dataset. If the t-test filtering method was applied to a transcriptomics dataset with more than the 8 time points found in the *T. saltator* dataset, the second and third parameters considered would possibly need to be different than the values decided for this dataset in order to ensure genes of interest are included in a reasonably small filtered dataset.

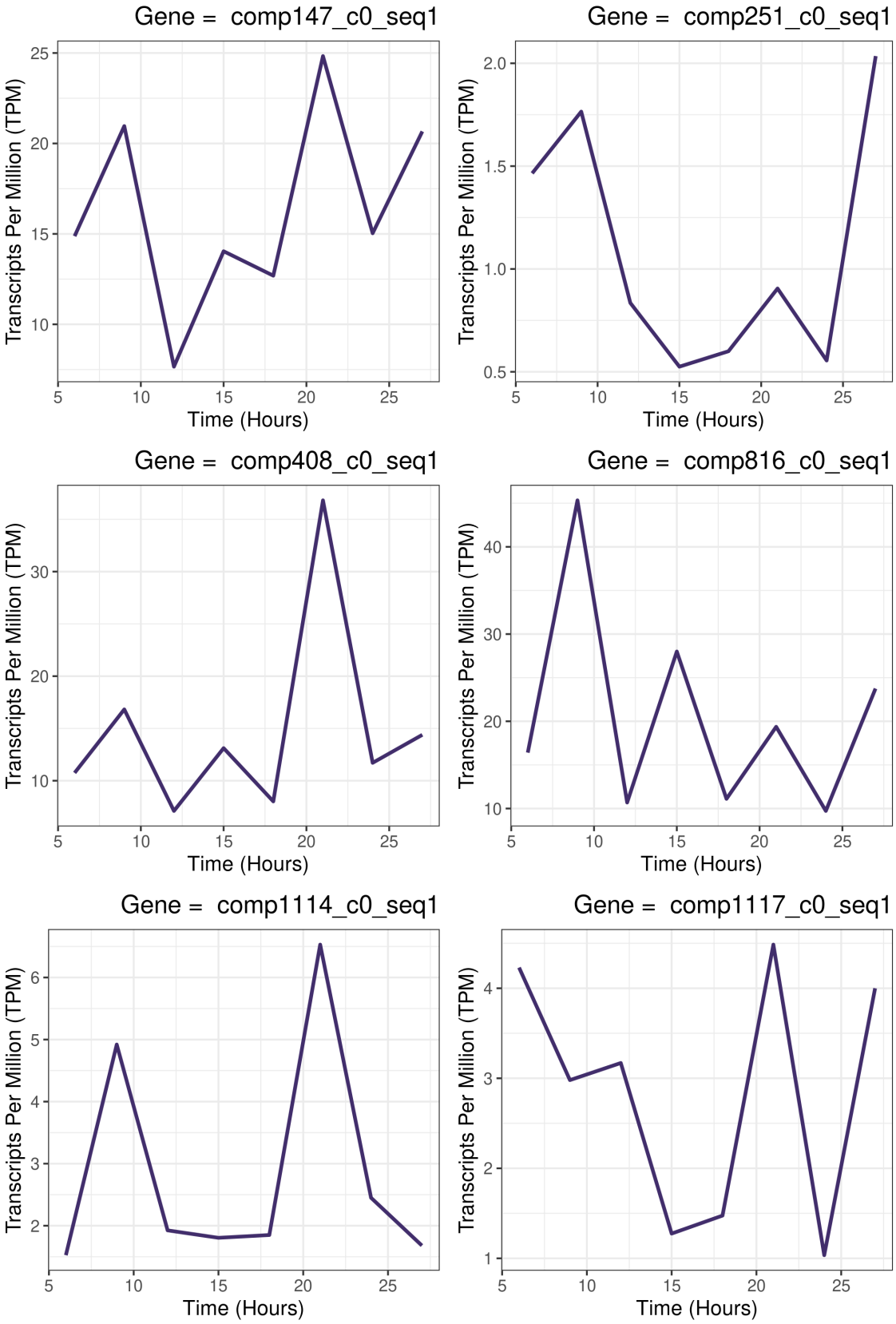


Figure 3.1:
A selection of the genes included in the reduced dataset generated by filtering via t-tests

Chapter 4

Cluster Analysis

Introduction to Clustering & Distance Measures

The Purpose of Clustering

Clustering involves classifying data into partitions which are usually referred to as clusters. The aim of clustering is to populate these clusters in such a way that all of the observations in a cluster are similar to one another whilst observations belonging to different clusters are dissimilar to each another. This technique allows a dataset to be reduced to only a few elements, which is advantageous for data visualisation, whilst still having data which are indicative of the dataset as a whole. Clustering is an unsupervised machine learning technique, thus clusters are populated based upon the dissimilarities between observations instead of being based upon labels accompanying the data which denote the true classification for each element as is the case in supervised machine learning. An unsupervised technique is required with this study as labels which denote a classification for the samples in the *T. saltator* dataset are not available. In machine learning, an observation is often referred to as an *object*. This is the nomenclature which will be used.

A way to quantify how dissimilar an object is to another object is to use a distance measure. A distance measure is constructed in such a way that the distance between objects which are similar is low whilst dissimilar elements have a high distance. The terms distance metric and distance measure will be used interchangeably.

Euclidean Distance

The most commonly used, and intuitive, distance measure is Euclidean distance: the distance between two points in Euclidean or Cartesian space[43]. Euclidean distance between two points lies in the interval $[0, \infty)$. If two objects P and Q both have n dimensions then they can be represented as $P = (p_1, p_2, \ldots, p_n)$ and $Q = (q_1, q_2, \ldots, q_n)$ respectively. The euclidean distance between P and Q is given by

$$d_{\text{Euc}}(P, Q) = \sqrt{\sum_{j=1}^{n} (p_j - q_j)^2} \tag{1}$$

Absolute Pearson's Correlation

The absolute Pearson's correlation distance measure has also been used in this study as a contrast to Euclidean distance. The two measures are distinctly different as the Euclidean distance belongs to the Minkowski family of distances whilst the Absolute Person's Correlation is correlation-based. A correlation-based distance is appropriate given that correlation is another measure of similarity. When using absolute Pearson's correlation between two objects which are perfectly correlated, either positively or negatively, then the distance between the objects is zero. This distance measure lies in the interval of [0,1]. Absolute Pearson's correlation is given by

$$d_{\text{cor}}(P, Q) = 1 - |r(P, Q)|, \tag{2}$$

where $r(P, Q)$ denotes the Pearson correlation coefficient, also known as the sample correlation

coefficient [44], which is defined as

$$r(P,Q) = \frac{\sum_{j=1}^{n} p_j q_j - n\bar{p}\bar{q}}{\sqrt{\left(\sum_{j=1}^{n} p_j^2 - n\bar{p}^2\right)\left(\sum_{j=1}^{n} q_j^2 - n\bar{q}^2\right)}} \tag{3}$$

with the same definitions of n, P and Q as for the Euclidean distance measure.

Clustering Methods

Agglomerative Hierarchical Clustering

One of the oldest clustering methods is agglomerative hierarchical clustering with incomplete iterative formulas for the process being proposed at least as early as 1966[45]. It is typical now for the term *hierarchical clustering* to be used to describe only agglomerative hierarchical clustering due to the prevalence of these methods when compared to alternatives which also produce a hierarchy.

The algorithm starts by classifying N objects into N clusters which each contain one object. The algorithm then identifies which two clusters are the closest to each other using a distance measure and merges these clusters into one cluster. The algorithm then identifies which two clusters are closest together after the merger and the process is repeated until the clusters have been merged into one cluster. The step-by-step results of the algorithm can be visualised with a diagram called a dendogram which graphically presents the mergers in a tree-like plot. The tree can be cut to ensure there are the desired number of clusters.

One of the advantages of hierarchical clustering is the intuitive method of visualisation offered by a dendogram. Figure 4.1 is a dendogram which shows how the genes in a dataset are grouped together to make a cluster. The height at which two branches are joined together denotes the dissimilarity. Mergers at the top of the plot denote a high degree of dissimilarity between the branches.

A tunable parameter when using hierarchical clustering is linkage criteria. The parameter describes where the distance between clusters is measured from and to. Single-linkage denotes the distance is measure between elements which are most similar to each other. Complete-linkage is measure between the elements which are most dissimilar[46]. Average-linkage is measured between the centres of each cluster. The algorithm for complete-linkage has previously been found to tend to produce a balanced hierarchy[47]. Based upon this finding, the decision to use complete-linkage was made.

Once a hierarchy is generated, a list of the classifications for each element can quickly be obtained by 'cutting' the tree. This is very advantageous as it allows cluster validation metrics, which are used to to decide upon the optimal number of clusters, to be quickly calculated.

DIANA Clustering

DIANA (DIvisive ANAlysis) is a divisive clustering technique first proposed by Kaufman and Rousseeuw in 1990[49]. Whilst still hierarchical in nature, a divisive clustering technique proceeds in the opposite direction to an agglomerative method; DIANA begins with a single cluster of N objects and then repeatedly splits the clusters until there are N clusters which each contain one object.

The decision of how to split the clusters in a divisive approach is not simply the inverse of the method used to merge clusters in agglomerative hierarchical clustering. According to the originators of the algorithm, this is due, in part, to the number of objects which must be considered in the first step of each approach. In the first step of an agglomerative approach, all possible mergers are considered which requires $\frac{N(N-1)}{2}$ options to be considered. However, considering all possible divisions of a cluster with N objects for a divisive approach would involve considering $2^{N-1}-1$ possible divisions. An exponential function results in a huge increase of possibilities to consider as N increases when compared with a quadratic function. As such, DIANA does not consider all possible divisions and instead looks for which object is most dissimilar to all other objects. For DIANA, the most dissimilar object is defined as being the object with the highest average distance from every other object. The most dissimilar object is removed from the cluster and forms a new

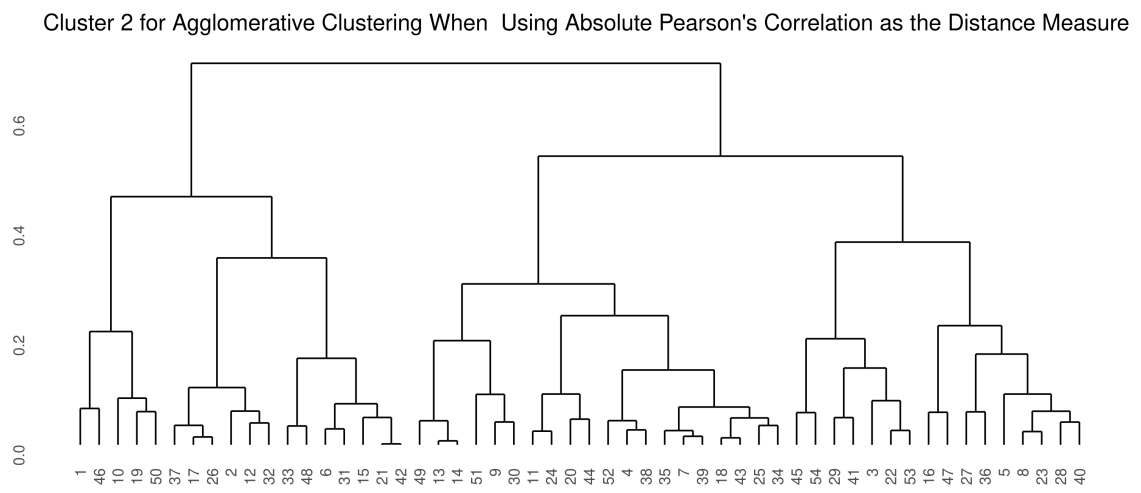


Figure 4.1:
Dendogram of a cluster generated using agglomerative hierarchical clustering.

Produced using the `ggdendro` package[48]

cluster. For each object O_i in the original cluster, the dissimilarities between O_i and all other objects in the original cluster are averaged. The dissimilarities between O_i and the objects in the new cluster are also averaged. The two averages are then compared. If the average dissimilarity between O_i and the new cluster is less than the average dissimilarity between O_i and the original cluster, then O_i joins the new cluster. The steps then repeat by considering the largest cluster at each step until every cluster contains only one object.

According to Kaufman and Rousseeuw, DIANA is still around two times slower than standard agglomerative hierarchical clustering. It should be noted that, as DIANA is not simply hierarchical clustering in the inverse direction, the results found by using the two methods should differ in general. DIANA has many of the same advantages as agglomerative hierarchical clustering. A dendogram can be used as a visual representation of the clustering process and once the algorithm has finished, clustering partitions with different numbers of clusters can be quickly found.

PAM

Another method of clustering proposed by Kaufmann and Rosseeuw is PAM (Partitioning Around Medoids)[50]. PAM begins by randomly choosing k representative objects, referred to as medoids. Each object in the dataset is assigned to the cluster which contains the medoid the object is closest to. Similar to DIANA, the average dissimilarity is then considered. The next step then involves considering a different choice of medoids and classifying the data into clusters again. If the average dissimilarity decreases then this set of cluster labels becomes the preferred labelling. The steps then repeat until a local minimum is found and the final set of cluster labels is found which results in the lowest average dissimilarity.

An advantage of PAM is that the optimal medoids chosen by the method can be reported. PAM is also more robust against noise and outliers than the similar algorithm, k-means, as PAM minimises a sum of similarities instead of a sum of squared euclidean distances and uses cluster representatives instead of cluster mean values[51]. A disadvantage of PAM is when varying k, the cluster labels have to be generated by running the whole algorithm again which is unlike hierarchical clustering and DIANA where the trees generated by these methods can simply be 'cut' in a different place to achieve the number of clusters required. Running the algorithm multiple times can lead

to huge increases in processing time. The multiple cores found in modern processors can be used to somewhat mitigate the time it takes to run PAM multiple times by running instances of PAM with different numbers of clusters in parallel. However, it is still incredibly time-consuming to run PAM multiple times.

Internal Cluster Validation Metrics

In order to compare the results generated by different clustering methods (or by different parameters), metrics which denote how well the clusters proposed fulfil the purpose of clustering are used. The purpose, as previously discussed, is to classify objects into clusters where all objects belonging to a cluster are similar, whilst objects belonging to different clusters are dissimilar. Three cluster validation metrics, which attempt to address how well this aim has been met, will now be presented. All formal definitions have been adapted from those found in a vignette for the `clValid` R package[52].

Dunn Index

Preamble

Dunn index is the minimum distance between clusters in a dataset divided by the maximum cluster diameter (the distance between the most dissimilar objects in a cluster) in the dataset. The Dunn index is associated with being sensitive to outliers and also having a high computational cost as the number of clusters increase. Dunn index lies in the interval $(0, \infty)$ with higher values indicating better clustering.

Formal Definitions

The Dunn Index, DI, for k clusters is defined as

$$DI_k = \frac{\min_{1 \leq i < j \leq k} \delta(C_i, C_j)}{\max_{1 \leq i \leq k} (\Delta_i)}, \tag{4}$$

where C denotes a cluster, k denotes the number of clusters, i and j denotes cluster indexes, $\delta(C_i, C_j)$ denotes the distance between the i^{th} and j^{th} clusters and Δ_i denotes the diameter of the i^{th} cluster.

Connectivity

Preamble

Connectivity is used to describe to what extent objects are placed in the same cluster as their k-nearest neighbours. Connectivity lies in the interval $[0, \infty]$ with the optimal value being zero.

Formal Definition

If N is the total number of objects in a dataset, $\mathcal{C} = \{C_1, \ldots, C_k\}$ is the clustering partition of k clusters, $nn_{i(j)}$ is the j^{th} nearest neighbour of object i and letting $x_{i,nn_{i(j)}} = 0$ if i, j belong to the same cluster and $\frac{1}{j}$ otherwise then connectivity is given by

$$\text{conn}(\mathcal{C}) = \sum_{i=1}^{N} \sum_{j=1}^{l} x_{i,nn_{i(j)}}, \tag{5}$$

where l is the number of nearest neighbours considered for each object.

Silhouette Width

Preamble

Silhouette width estimates the average distance between clusters. Silhouette width is the mean of each object's silhouette value which measures the degree of confidence in a clustering assignment. Silhouette width is in the range $[-1, 1]$ with one being the optimal value.

Formal Definitions

The silhouette of object i is defined as

$$S(i) = \frac{b_i - a_i}{\max(b_i, a_i)}, \tag{6}$$

where a_i is the average distance between i and all other elements in the same cluster and b_i is average distance between i and the objects in the nearest cluster. The silhouette width for a set of clusters containing N objects is defined as

$$SW(\mathcal{C}) = \frac{\sum_{i=1}^{N} S(i)}{N} \tag{7}$$

The Application of Cluster Analysis to the *T. saltator* Dataset

Preamble

The three discussed clustering methods were applied to two datasets created by filtering the original transcriptomics dataset for *T. saltator* using different filtering methods. The first dataset was created by using ANOVA filtering with a significance level of 2.5%. The second dataset was created by filtering using the t-test method where the parameters were set such that only samples with as least two significant changes and a maximum difference of one between the number of significant positive and negative changes were included in the filtered dataset. A significance level of 2.5% was used. In addition to the genes being centred before filtering, the genes were scaled after filtering. The clustering methods were run on both datasets using both Euclidean distance and absolute Pearson's correlation as distance measures. The number of clusters, k, was varied between two and five by intervals of one and then considered for every increment of five until $k = 200$. The Dunn index, silhouette width and connectivity were calculated at each of these values.

Comparison of Distance Measures

To visualise the difference between using Euclidean distance and using absolute Pearson's correlation, the quantiles of the distances between clusters have been plotted as histograms. Figure 4.2 shows the histograms generated when using agglomerative clustering with 95 clusters on the ANOVA filtered dataset. It should be noted that figure 4.2 is representative of the plots for other methods and thus insights gathered from these plots also apply for the other methods. The distances between each cluster were calculated using the centres of each cluster (therefore using average linkage).

The separation between quantiles is considerably more distinct for the absolute correlation plot than for the Euclidean distance plot. Whilst there is still some crossover for the 25% and and 50% quantiles in the former, there is much more pronounced crossover between all quantiles for the latter. Having distinct quantiles indicates the clustering is consistent in performance for each individual cluster.

Of particular interest is the 100% quantile: the furthest distance to another another cluster for each cluster. The correlation plot shows very little spread for this quantile: over half of the observations belonging to the quantile have been placed in the final bin. This close spread shows most of the clusters have a correlation of approximately zero with at least one other cluster in the cluster partition. However, the Euclidean distance plots shows a much wider spread for the 100% quantile: suggesting the maximum distance to another cluster varies considerably for different clusters.

The 50% quantile also provides evidence for absolute Pearson's correlation being the better distance measure for clustering in this context. As clustering aims to segment data into clusters where each cluster is dissimilar to the other clusters, it is beneficial for the 50% quantiles to be far away from zero. It can clearly be seen the 50% quantiles are all beyond halfway across the x-axis in the absolute correlation plot. This is not the case for many of the clusters in the euclidean distance plot.

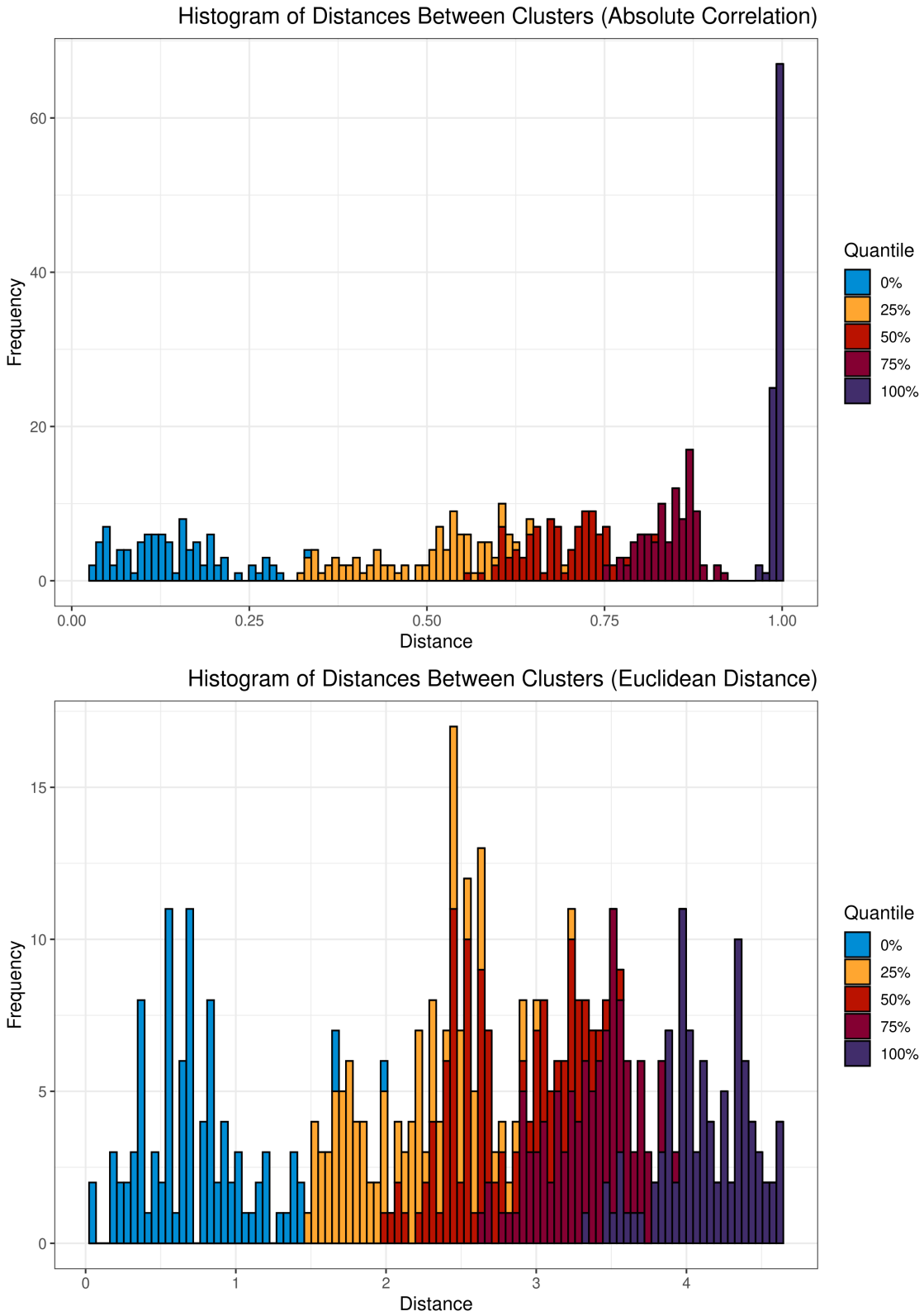


Figure 4.2:
Histogram plots of quantile distances between the centres of clusters using Euclidean distance and Absolute Pearson's correlation. The clusters were generated using agglomerative hierarchical clustering.

Comparison of Running Times

The time required to produce a cluster partition for $k = 95$ for each of the three clustering algorithms was found. The times reported do not include the time required to calculate the dissimilarities between genes which were found in advance . It should be noted that, as agglomerative hierarchical clustering and DIANA both require a full hierarchy to be created when clustering, changing the value of k does not change the running time of the algorithm. Running times on the ANOVA filtered dataset can be found in table 4.1. PAM was found to the slowest method by a considerable amount: taking over 75 minutes to finish. This long running time became particularly problematic when generating cluster validation plots as it resulted in each figure taking over 48 hours to produce despite using server grade hardware with 24 cores. The DIANA algorithm took 614.886 seconds and agglomerative hierarchical clustering took only 9.837 seconds. These results show DIANA to be 62.5 times slower than agglomerative hierarchical clustering which disputes Kaufman and Rousseeuw's claim of DIANA being only two times slower.

Method	Time (s)
Agglomerative	9.387
DIANA	614.886
PAM	4525.637

Table 4.1:
Running times for clustering algorithms for $k = 95$ on 7124 genes.

ANOVA Filtered with Euclidean Distance Cluster Validation

The three plots of the validation metrics against k for the ANOVA filtered dataset when using a Euclidean distance measure can be seen in figure 4.3.

Dunn Index

The plot of Dunn index against k shows a deep decrease in Dunn index for both PAM and agglomerative hierarchical clustering for $2 \leq k \leq 5$ whilst the Dunn index for DIANA increases in this interval. Peculiarly, PAM has another sharp drop in Dunn index for $k = 55$. This is odd given the Dunn index for PAM for $k = 50$ and $k = 60$ are quite similar and the Dunn index at $k = 55$ does not follow the overall trend. The plot suggests agglomerative hierarchical clustering with $k = 75$ produces the best cluster partition. However, if a clustering partition with 105 clusters or more is desired then this plot provides evidence for DIANA being the better performing algorithm

Connectivity

The plot of connectivity suggests agglomerative hierarchical clustering is the best clustering algorithm being considered whilst PAM is the worst for $k < 180$ and DIANA is the worst for $k > 180$. Despite DIANA having consistently higher connectivity than agglomerative hierarchical clustering, the overall trend for both methods is very similar. This makes sense given the two methods are closely related.

Silhouette Plot

The silhouette plot suggests PAM is the best method of clustering whilst agglomerative hierarchical clustering is the worst: in direct contradiction with the findings from the connectivity plot. There is a decreasing trend for all three methods.

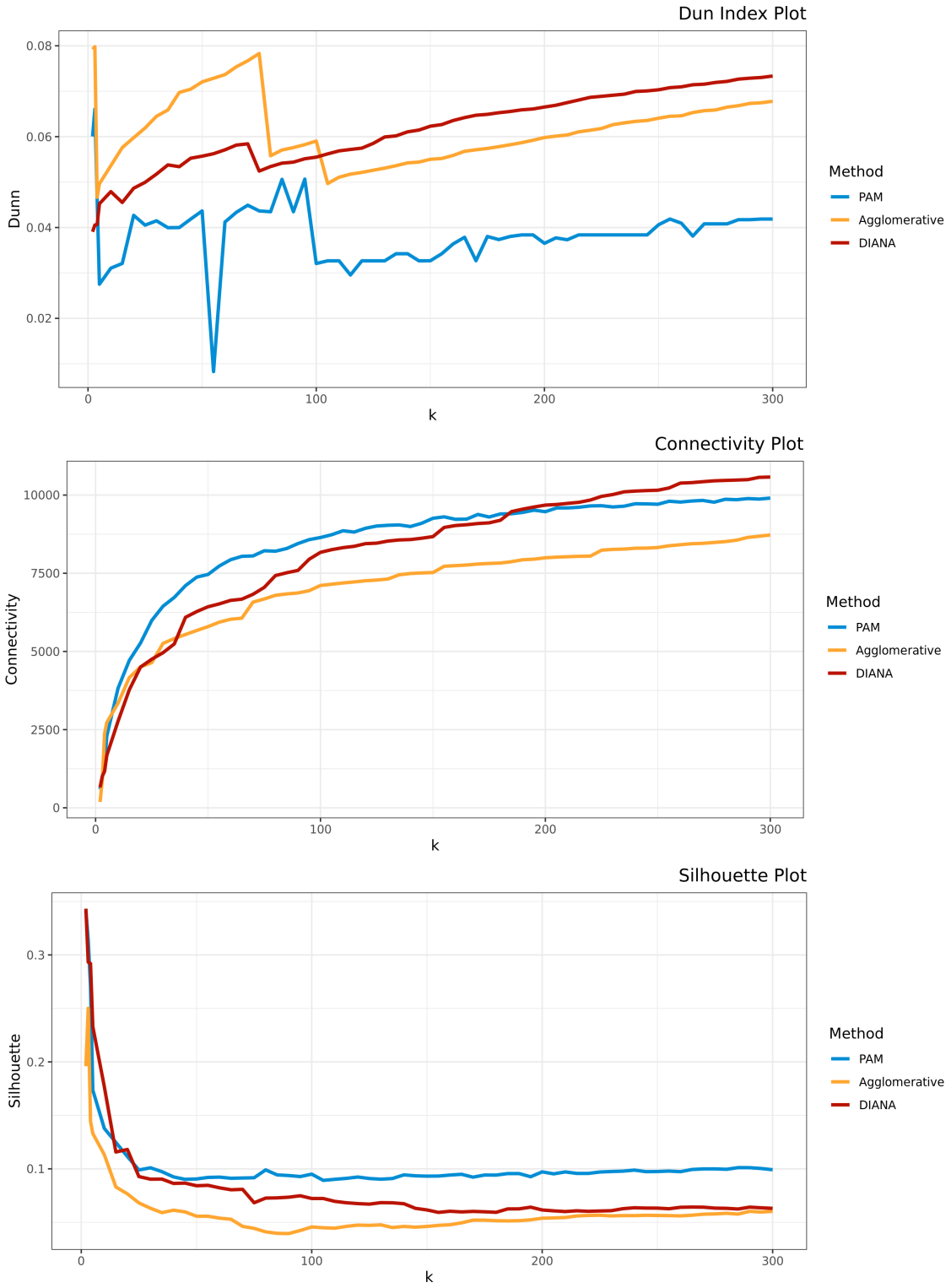


Figure 4.3:
Plots of validation metrics against k for the clustering methods when using the ANOVA dataset with Euclidean Distance

Produced using the `clValid`[52] package.

ANOVA Filtered with Absolute Pearson's Correlation Cluster Validation

The plots of the validation metrics for the clustering methods when applied to the ANOVA dataset and using absolute Pearson's correlation as a distance metric can be seen in figure 4.4

Dunn Index

The Dunn index plot shows a very steep drop in Dunn index for agglomerative hierarchical clustering for very low values of k. Despite this, the best Dunn index for this dataset is returned by agglomerative hierarchical clustering: regardless of the value of k. In the plot, the Dunn index for PAM seems to be random and does not suggest an optimal value of k for DIANA is not suggested either as, for the most part, the plot shows a horizontal line. It may be possible there is similar behaviour to the Dunn index for DIANA as there is for PAM but on a scale which cannot be seen in the plot.

Connectivity

The connectivity plot suggests agglomerative hierarchical clustering performance is poor when using a low number of cluster partitions but is the best method for $k \geq 25$. Conversely, DIANA appears to be the best method for low values of k but the worst method for higher values of k.

Silhouette Width

The Silhouette width plot suggests PAM is the best clustering method. This is surprising due to the frequent inferences made thus far which argue PAM is the worst clustering method. Also interesting is, for $k > 10$, there appears to be a rough line of symmetry between the silhouette width values for DIANA and hierarchical clustering. This behaviour is possibly due to the similarities between the two methods and the symmetry of the absolute value function, That is to say, if $f(x) = |x|$ then $f(x) = f(-x)$.

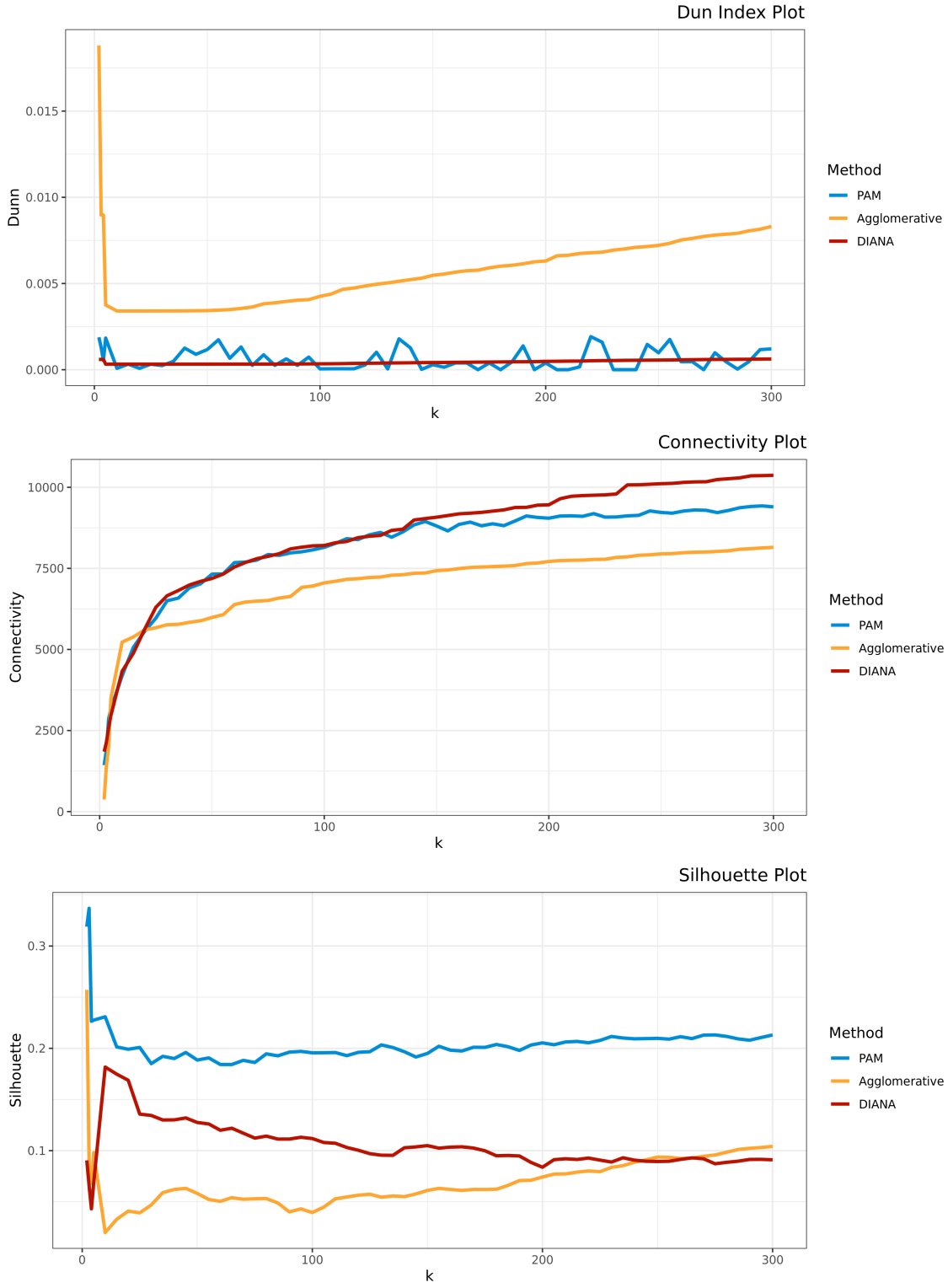


Figure 4.4:
Plots of validation metrics against k for the clustering methods when using the ANOVA filtered dataset with absolute Pearson's correlation

T-Test Filtered with Euclidean Distance Cluster Validation

Figure 4.5 shows the plots which will hereby be discussed.

Dunn index

There are a few discrepancies between the plot of Dunn index for the t-test filtered dataset using Euclidean distance and its ANOVA filtered counterpart. In the former, there is not a sharp decrease in Dunn index for agglomerative hierarchical clustering for very low values of k. This is dissimilar to the behaviour observed in the ANOVA filtered plot which did show a sharp decrease. Furthermore, the t-test filtered plot provides evidence for agglomerative hierarchical clustering being the best method for large values of k ($k > 170$) and DIANA being the better method of clustering for smaller values of k ($k < 170$) when using Euclidean distance. This contradicts the results from the ANOVA filtered dataset which suggested agglomerative hierarchical clustering is the best method for small values of k ($k < 105$) and DIANA is better for large values of k ($k > 105$). Also of interest is the profile for agglomerative hierarchical clustering, which shows a consistent increase in Dunn index as k increases, and the profile for PAM which is erratic until $k = 95$ and then shows a large decrease in Dunn index for $k = 220$

Connectivity

On the other hand, the connectivity plot for the t-test filtered dataset using Euclidean distance is comparable to its ANOVA filtered equivalent. The connectivity for agglomerative hierarchical clustering and DIANA is similar for low values of k ($k < 50$), but agglomerative hierarchical clustering clearly returns the best connectivity for larger values of k

Silhouette Width

The plot of silhouette width produces a similar result to its ANOVA filtered counterpart: providing evidence for PAM being the best clustering algorithm out of the three compared.

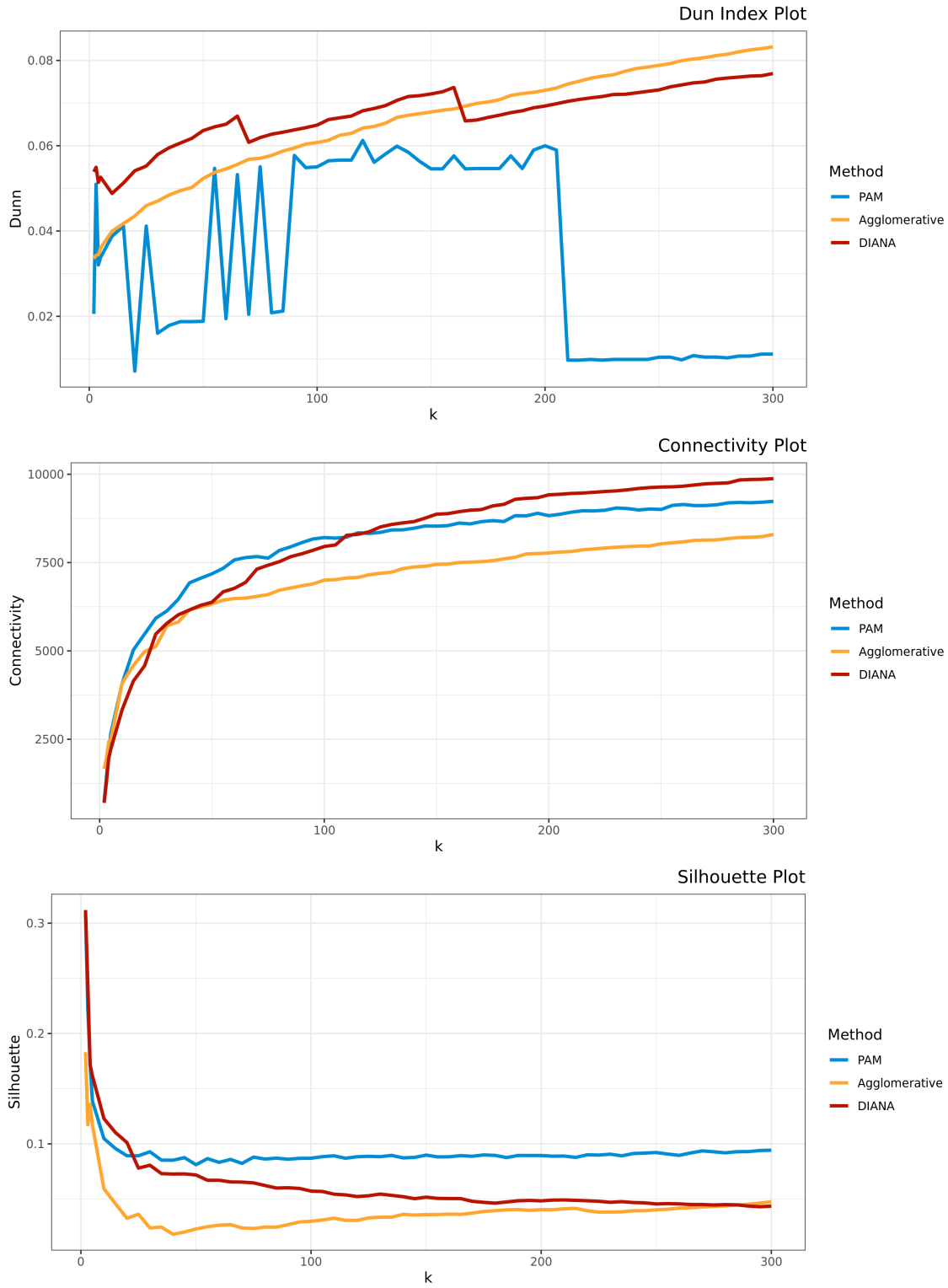


Figure 4.5:
Plots of validation metrics against k for the clustering methods when using the t-test filtered dataset with Euclidean distance

T-Test Filtered with Absolute Pearson's Correlation Cluster Validation

Figure 4.6 shows the validation metric plots which will hereby be discussed.

Dunn index

The plot of Dunn index for the t-test filtered dataset with the absolute Pearson's correlation distance measure suggests agglomerative hierarchical clustering is the best method when using absolute Pearson's correlation. This is the same result as for the ANOVA filtered counterpart to this plot. DIANA is presented as the worst method which is also similar to the ANOVA filtered counterpart plot which showed DIANA to consistently perform worse. However, PAM was shows to perform better than in the ANOVA plot.

Connectivity

The conclusions drawn after plotting connectivity for the t-test dataset are the same as for the ANOVA equivalent. Both plots rank agglomerative hierarchical clustering as the best method, PAM as the second best and DIANA as the worst.

Silhouette Width

The plot of silhouette width suggests PAM is the best method: similar to the silhouette plot of the ANOVA dataset when also using absolute Pearson's correlation. This plot also presents the same reflective relationship between agglomerative hierarchical clustering and DIANA as previously described for the ANOVA counterpart to this plot. This provides further evidence that this apparent relationship is indeed a relationship and not just a coincidence in the previous plot.

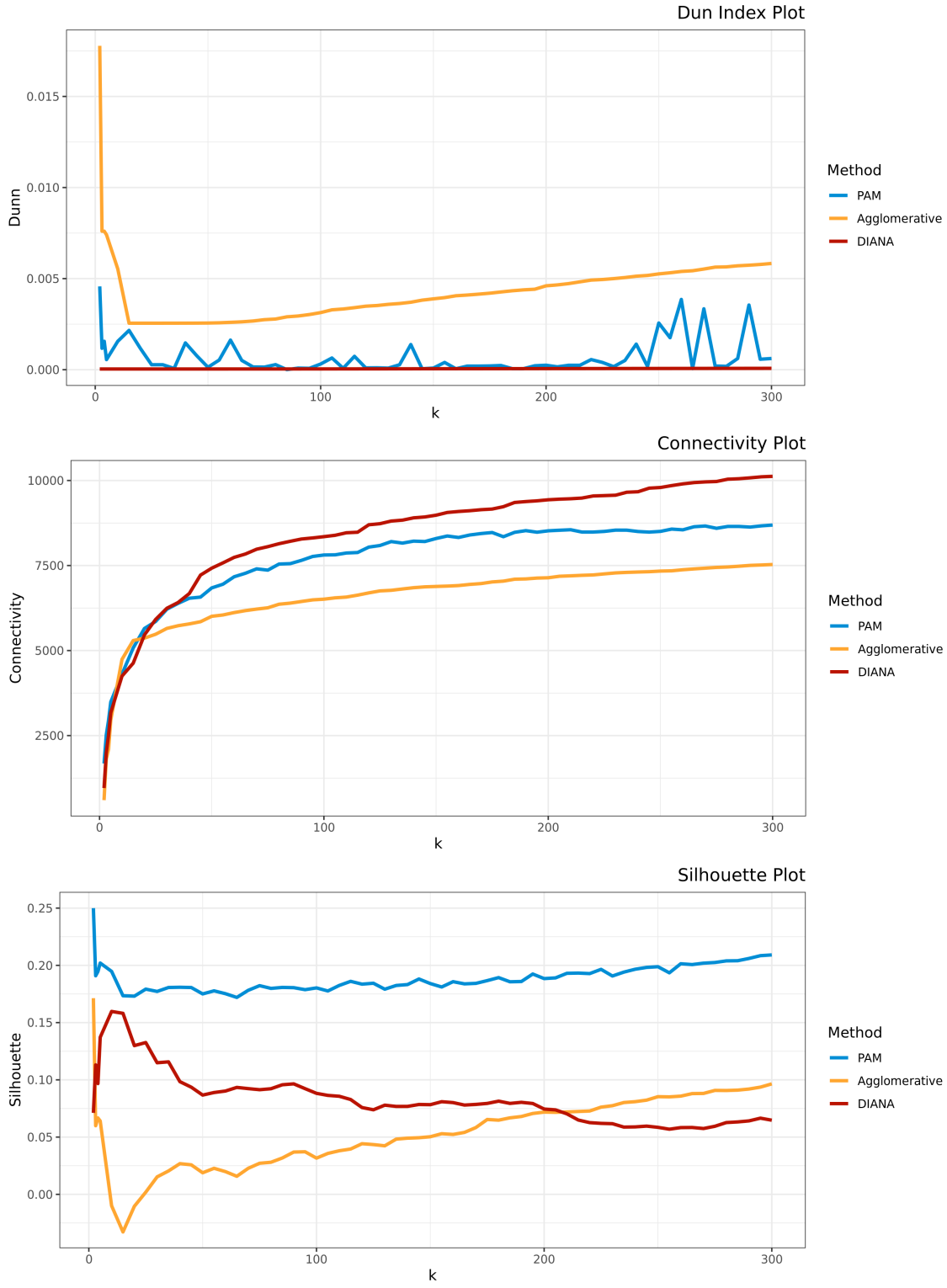


Figure 4.6:
Plots of validation metrics against k for the clustering methods when using the t-test filtered dataset with absolute Pearson's Correlation

Summary of Cluster Validation Results

Plotting Dunn index against k does not provide strong evidence for whether DIANA or agglomerative hierarchical clustering is the better clustering algorithm when using Euclidean distance. Agglomerative hierarchical clustering was shown to produce the best Dunn index results when using absolute Pearson's correlation.

Despite the contradictory results for Dunn index when using Euclidean distance which suggested different optimal methods when considering high and low values of k, agglomerative hierarchical clustering was shown to produce the best connectivity results across both datasets and both distance measures.

There is an interesting relationship between DIANA and agglomerative hierarchical clustering. When plotting silhouette width against k, the silhouette width for the two methods seem to converge if Euclidean distance is used. However, the silhouette width between the two methods seems to show a mirrored relationship when using absolute Pearson's correlation instead.

The decision was made to use $k = 95$ across the three methods when generating clusters for any further analysis which required the data to first be clustered. k was set to 95 as performance for this value is acceptable and it did not take an excessive amount of time to generate the cluster partition when using PAM. Whilst specific methods may perform better with different values of k, the same value of k was used across all three clustering methods so the results generated could be more directly compared.

Chapter 5

Correlation Analysis

Preamble

In addition to being used in a distance metric, Pearson's correlation can be used to provide additional insights into the relationships between genes by constructing networks based upon correlation. These networks allow an intuitive visualisation of the relationships between genes to be constructed. As previously discussed, Pearson's correlation can be used to find the similarity between genes. However, by lagging a gene against another gene and then calculating the correlation between the two genes, genes can be found which may potentially be involved in changing the expression levels of other genes. In order for the visualisation to be readable, only the most strongly correlated relationships should be shown in the visualisation. Cytoscape, a prominent program used for visualising biological networks, was used to produce figures for this document[53].

The visualisations are easy to understand. Nodes represent a gene whilst edges show a relationship between two genes. The edges have arrows which denote which gene may be influencing which. For example, if A and B are genes then $A \longrightarrow B$ denotes the gene expression of A may influence the gene expression of B. If an edge is red the edge represents negative correlation which implies A may be repressing B. Blue edges denote correlation which instead implies A is activating B.

Application of Correlation Analysis to the *T. saltator* Dataset

Correlating Genes Directly

Correlating all genes in the ANOVA filtered dataset and visualising the most significant correlations was considered. The activity profile for each gene was found by calculating the median activity at each time point. Pearson's correlation, r, was calculated between the set of medians across all time points for every gene. If $r > 0.95$ or $r < -0.95$ then the correlation was included in the visualisation of the network as an edge. Each node represents the expression profile of a gene. If a gene did not have any significant correlations with another gene then the gene was not included in the figure as a node. However, as can be seen in figure 5.1, the visualisations generated by this method are far too excessively detailed to make any meaningful inferences beyond many genes in the dataset being highly correlated. This is the case even if more stringent requirements are set for correlation to be represented as an edge such as $r > 0.99$ and $r < -0.99$.

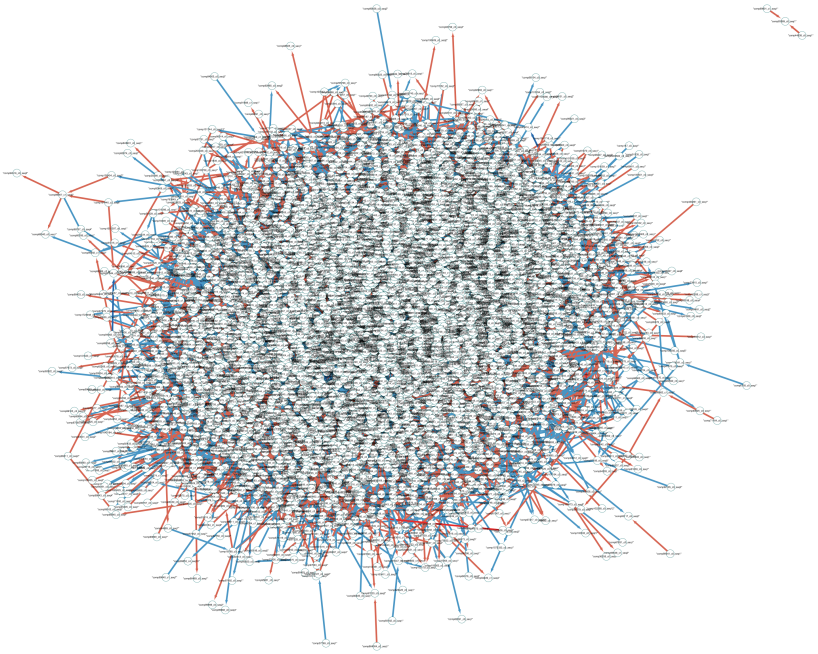


Figure 5.1:
Visualisation of a Gene Correlation Network. Genes are from the t-test filtered dataset.

Produced using Cytoscape[53]

Correlating Clusters

Bearing the previous results in mind, it would be appropriate to employ clustering as a method to create more readable correlation networks. The activity profile for each cluster was produced by finding the profile for each gene, as before, and then finding the mean activity across the entire cluster for each time point if using Euclidean distance or median activity if using absolute Pearson's correlation. Networks were then visualised by only including correlations where $r > 0.95$ or $r < -0.95$ and using cluster activity profiles as nodes.

A large number of networks were considered. PAM, DIANA and agglomerative hierarchical clustering, with 95 clusters, were applied to both ANOVA and t-test filtered datasets, Both Euclidean distance and absolute Pearson's correlation were used as distance metrics, which further increased the number of networks considered, and lags of either one or two time steps were considered. Interactive versions of all of these networks can be found at `users.aber.ac.uk/nsc/cytoscape.html`. Visualisations have been selected for inclusion in this document based upon interesting characteristics. The nodes which contribute to these characteristics are yellow in the visualisations.

The time profiles for nodes of interest have also been plotted. However, the plotting of these profiles varies depending on whether Euclidean distance or absolute Pearson's correlation was used when clustering. In both cases, the profile for each individual gene was found by finding the median expression level at each time point. For Euclidean distance, the mean at each time point across the individual gene profiles were plotted alongside error bars which show ± 2 standard errors for the gene expression measurements. For absolute Pearson's correlation, genes are sorted into positive or negative correlation categories. The mean profiles for the two categories are then plotted.

Cluster Correlation Network 1

Figure 5.2 is the Cytoscape visualisation produced for clusters produced by Agglomerative hierarchical clustering on the ANOVA filtered dataset when using absolute Pearson's correlation as a distance metric. The correlation between the clusters have been calculated with a lag of one time step (three hours). It would appear to be the case that clusters are mostly positively correlated as the figure shows only one relationship where there is negative correlation. However, due to the use of absolute Pearson's correlation for the distance metric, whether the correlation is positive or negative has little meaning. This makes it impossible to ascertain if a cluster is possibly activating or repressing another cluster.

The node for cluster 73 is interesting as six more clusters seem to be correlated, when lagged, with this cluster. However, plotting time profiles for cluster 73 as well as two of the six aforementioned clusters, cluster 4 and cluster 8, shows whilst cluster 73 correlates well with the two other clusters once lagged, these two other clusters are remarkably similar. Fig 5.3 presents these plots.

Cluster Correlation Network 2

The next network considered can be seen in figure 5.4. This visualisation shows correlation between clusters generated by DIANA clustering applied to the ANOVA filtered dataset with absolute Pearson's correlation used as the distance metric. The correlation between clusters has been lagged two time points (six hours). This figure demonstrates how complex networks which show similarities between genes can be developed as over 42 clusters are in a single connected network.

Of interest is the node for cluster 66 which shows the cluster is correlated with a lagged version of itself. The overall time profile for this cluster can be seen in figure 5.5. This strong autocorrelation provides evidence for the genes in the cluster being circadian and circatidal due the overall cluster profile seemingly having a period of 12 hours.

Cluster Correlation Network 3

Figure 5.6 shows another Cytoscape visualisation. This figure shows the network constructed when the correlations between clusters generated via PAM using Euclidean distance are found when the clusters have been lagged by two time steps. The nodes which represent cluster 60 and cluster 90 were deemed to be of interest due to both clusters showing high degrees of correlation with many other clusters: including each other.

Figure 5.7 shows the time profiles for cluster 60 and cluster 90. The figure shows activity which would be expected if the genes in cluster 90 are upregulating the genes in cluster 60. The network visualisation suggests the genes in clusters 91, 2, 77, 26 and 76 may upregulate the genes in cluster 90 which upregulates the genes in cluster 60 which in turn upregulates many more genes.

Cluster Correlation Network 4

The final Cytoscape visualisation which was selected for inclusion is figure 5.8. This is a visualisation of the correlations between clusters lagged by three hours. The clusters were generated by agglomerative hierarchical clustering applied to the t-test filtered dataset. Euclidean distance was used for the distance metric. As can be plainly seen, there are no large networks in this visualisation; the largest connected network is only five nodes. However, similar to figure 5.4, there is a node which is shown to be highly correlated with a lagged version of itself: cluster 88. Figure 5.9 shows the time profile for this cluster. As one would expect given the lag is one time point, the profile for this cluster shows oscillatory behaviour which is similar to the core clock gene REVERB. As this profile has a period of hours which is a divisor of both the circatidal and circadian periods, this is evidence which argues in favour of the genes in the cluster being both circadian and circatidal.

Conclusion of Findings

Using correlation as the basis for building networks between clusters can clearly provide useful results as long as the requirements needed for correlation to be included in the network is strict enough. The threshold, T, should be large enough that an edge between two nodes n_1, n_2 is only included in the network if $r(n_1, n_2) > T$ where T ensures a reasonable number of edges in the network. Of course, what constitutes a reasonable number of edges is subject to debate and should depend on the number of genes being considered. It should be noted when using a practical transcriptomics dataset, which has not been heavily filtered, correlating genes directly is unlikely to produce a useful visualisation. At the very least, a value of T would be required which would be too strict and thus many important relationships would be lost. As such, first clustering the genes and then finding overall profiles for each cluster, as demonstrated here, seems to be a better approach.

An obvious limitation of using clusters is it is not possible to tell which genes belong to which cluster from the visualisation. A possible way to solve this problem would be to correlate clusters, choose edges between clusters which are of interest, and then create correlation networks between the genes in the clusters. Correlation between genes in the same cluster would not be shown unless the edge of interest denotes autocorrelation. This approach should still often produce readable visualisations. The nodes could be coloured to denote which cluster each gene belongs to.

Interpreting a correlation network between clusters classified using absolute Pearson's correlation requires more nuance than when Euclidean distance has been used for generating a cluster partition. Due to the properties of the absolute value function, whether a cluster is negatively or positively correlated with another cluster makes no difference. Visualising a correlation network of individual genes belonging to clusters which have been deemed to be of interest as described in the previous paragraph, may be beneficial.

Finally the popular mantra amongst statisticians, "correlation does not imply causation", applies. Therefore the findings gathered from correlation networks may show genes which are influencing the activity of other genes, it is also possible this is not the case. Biological experimentation would be needed to prove or disprove such a relationship. However, correlation networks can at least be used to find possible relationships between genes and then used as the basis for an experiment conducted by biologists.

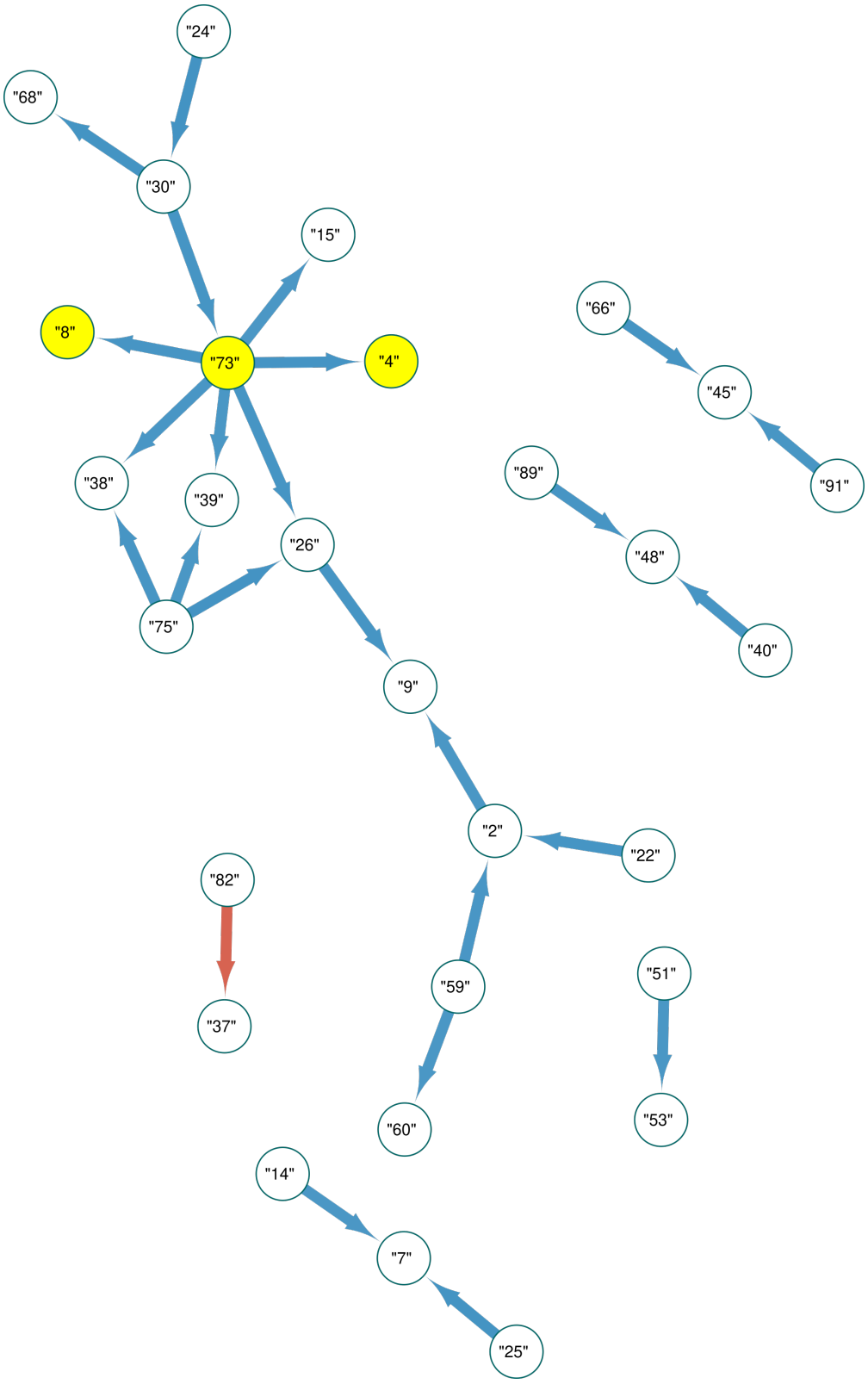


Figure 5.2:
Visualisation of Cluster Correlation Network 1. Produced by agglomerative hierarchical clustering with absolute correlation applied to the ANOVA filtered dataset. A lag of three hours was applied when correlating.

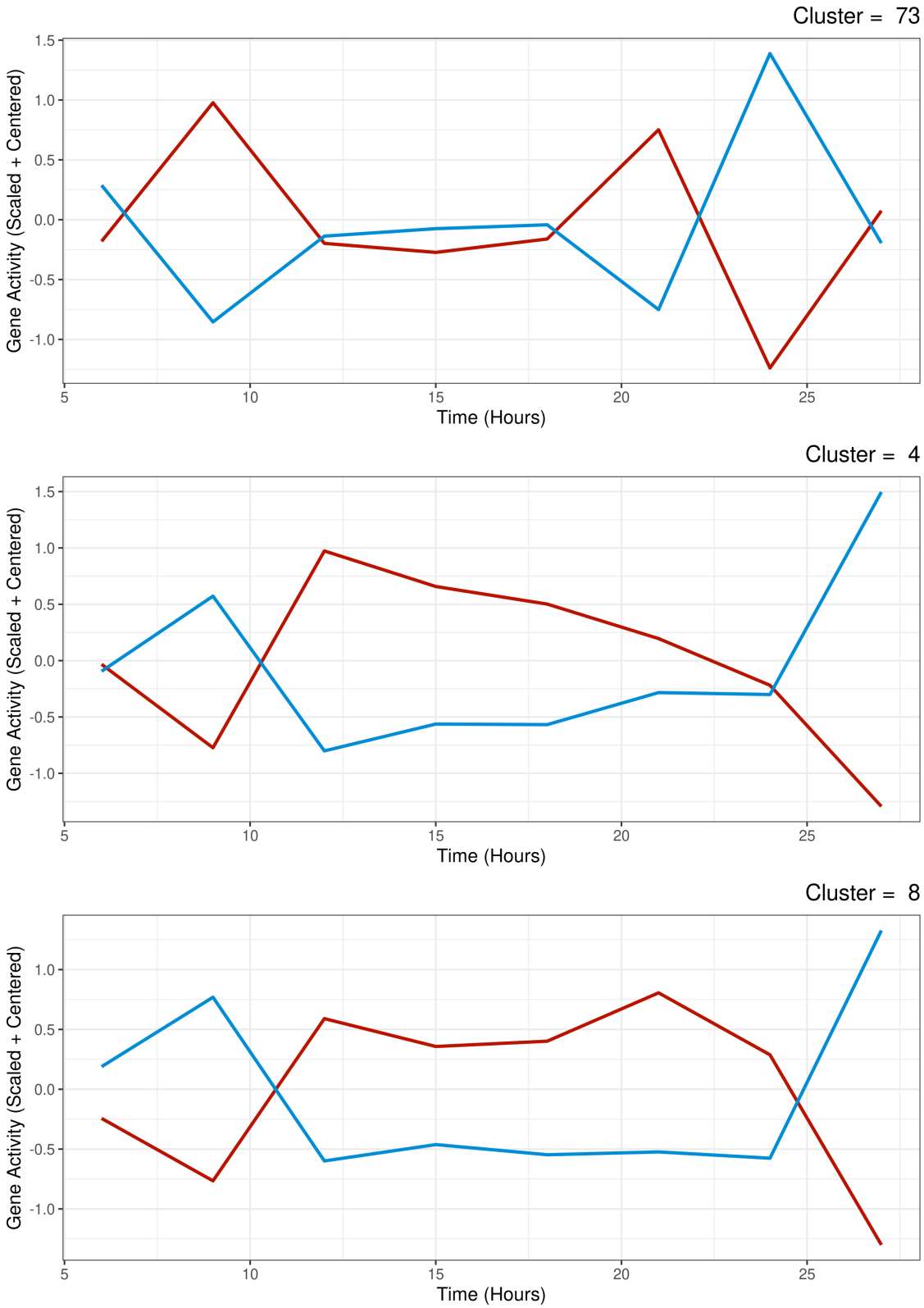


Figure 5.3:
Profiles of three clusters selected from Cluster Correlation Network 1. Cluster 73 was shown to correlate highly with both cluster 4 and cluster 8 when the latter two clusters were lagged by three hours.

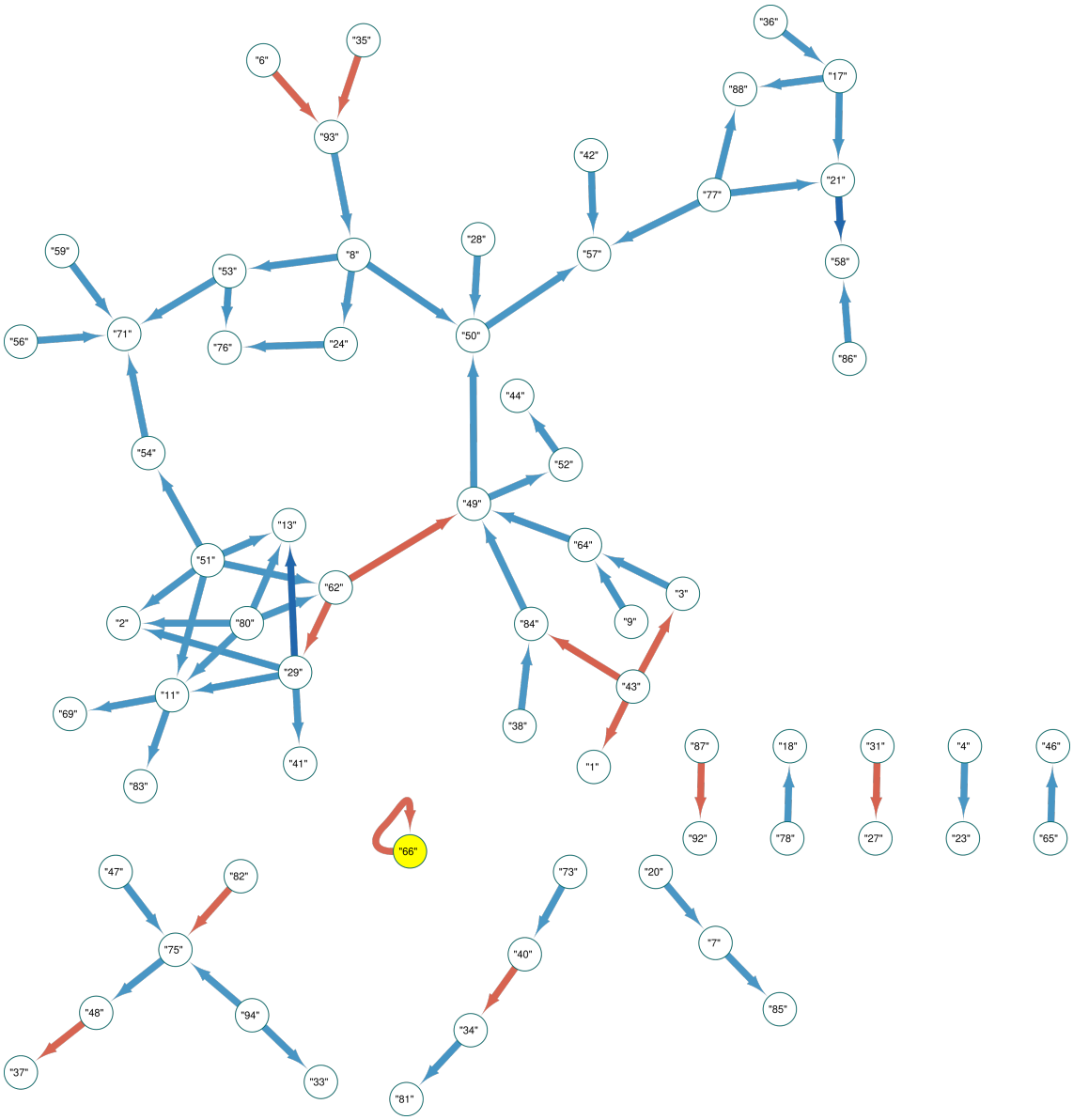


Figure 5.4:
Visualisation of Cluster Correlation Network 2. Produced by DIANA clustering with absolute correlation applied to the ANOVA filtered dataset. A lag of six hours was applied when correlating.

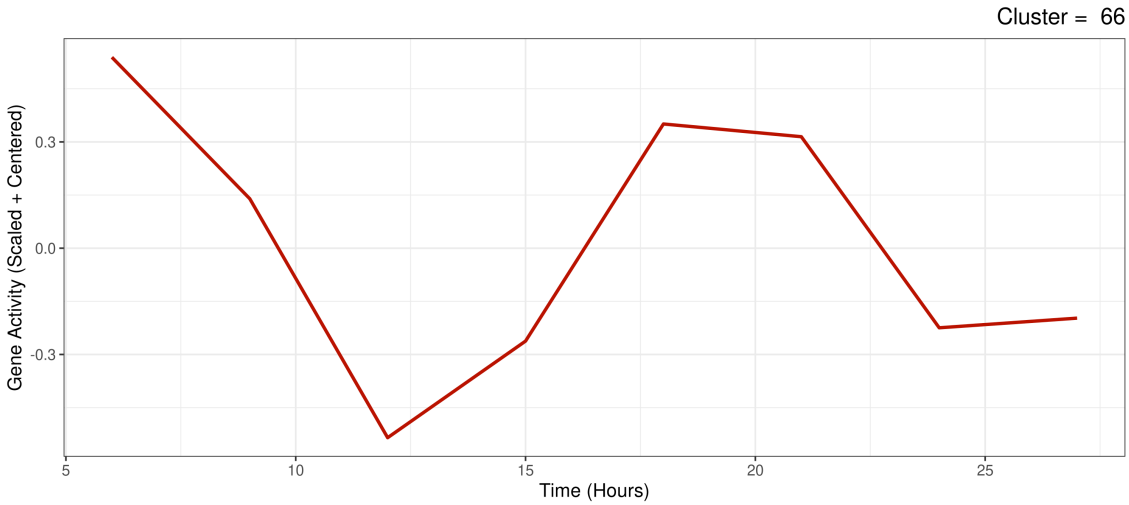


Figure 5.5:
Profile of cluster 66 from Cluster Correlation Network 2 which was shown to be strongly correlated with a version of itself lagged by six hours.

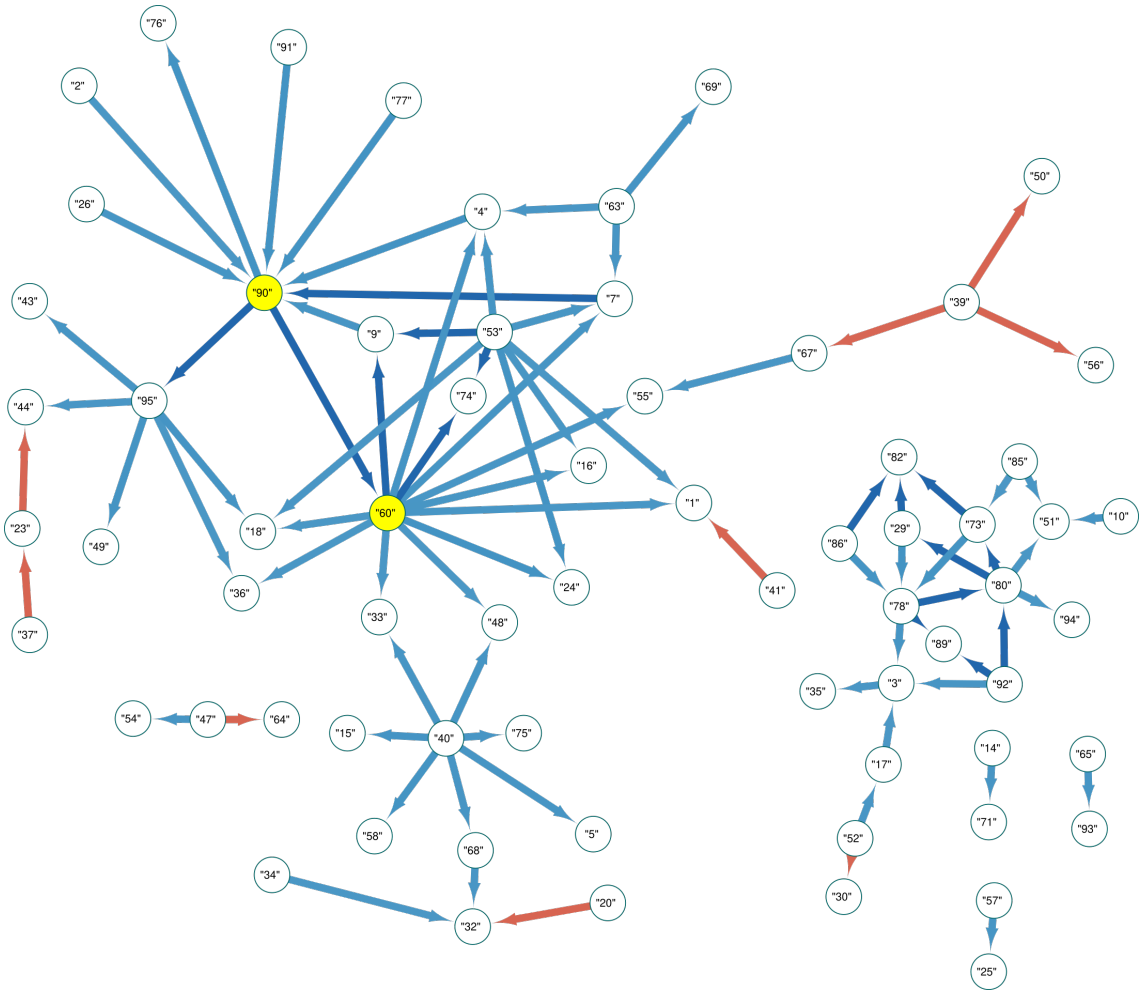


Figure 5.6:
Visualisation of Cluster Correlation Network 3. Produced by PAM clustering with Euclidean distance applied to the ANOVA filtered dataset. A lag of six hours was applied when correlating.

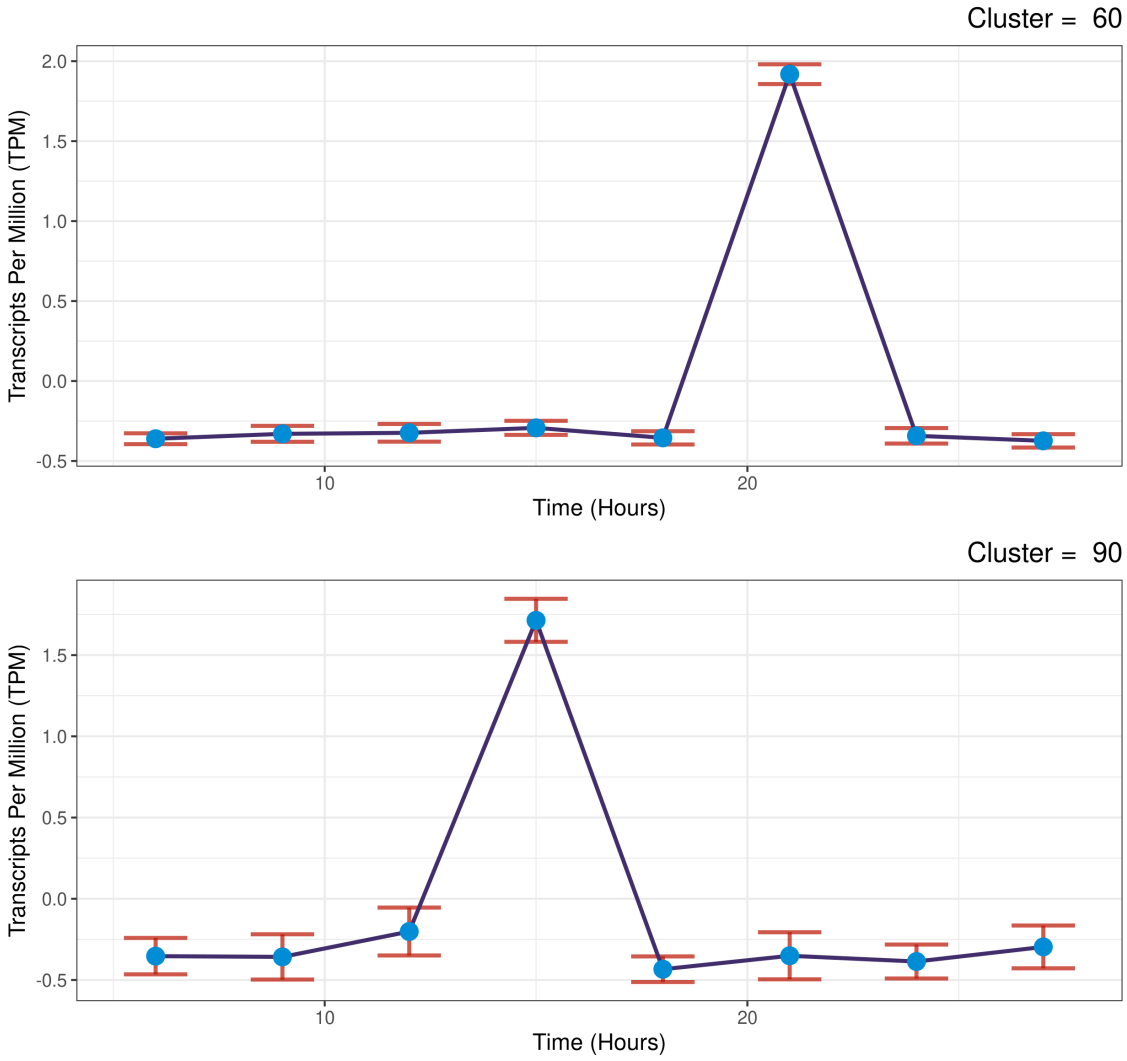


Figure 5.7:
Profiles of three clusters selected from Cluster Correlation Network 3. Cluster 60 was shown to correlate highly with cluster 90 when lagged by six hours.

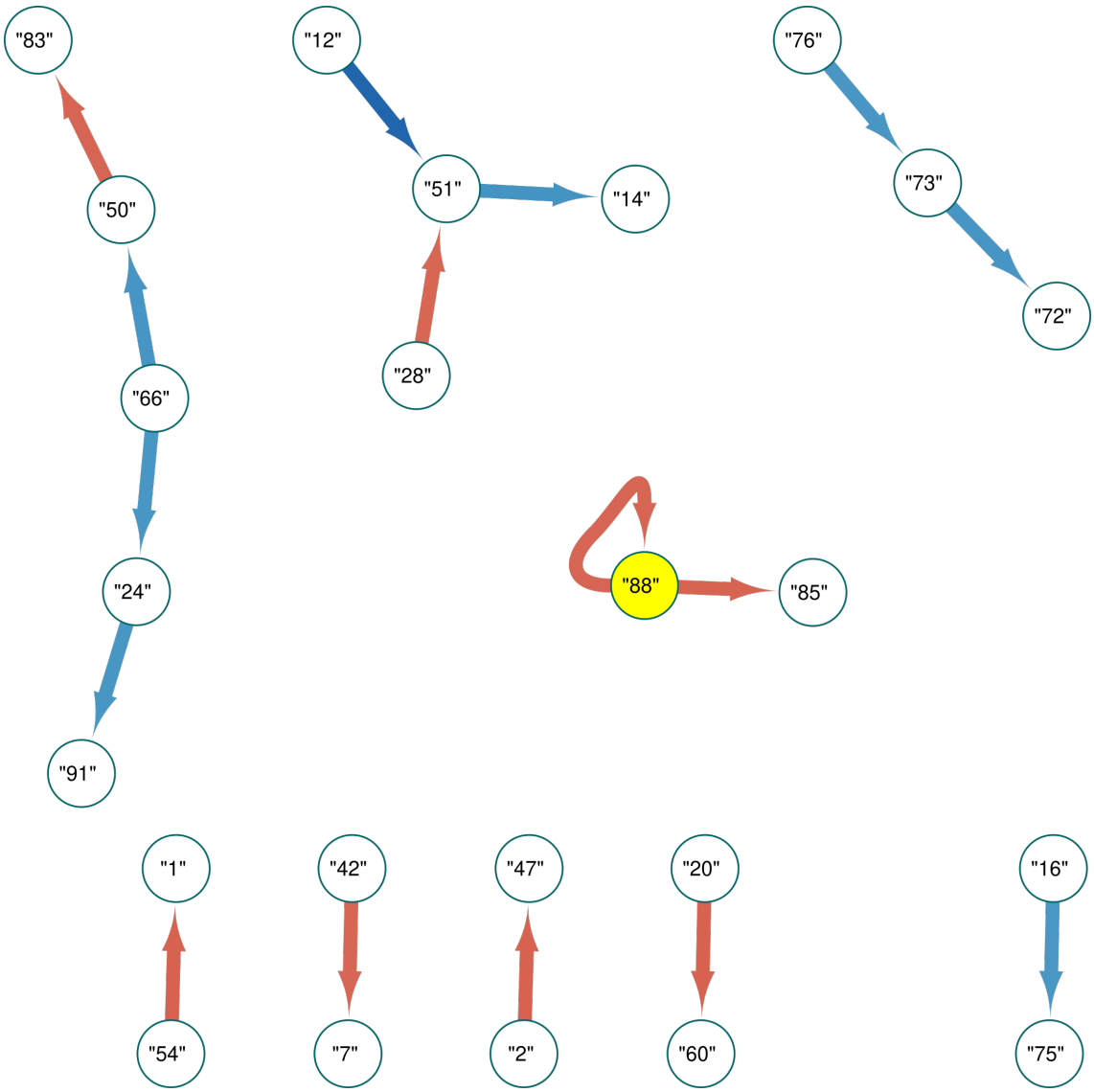


Figure 5.8:
Visualisation of Cluster Correlation Network 4. Produced by agglomerative hierarchical clustering with Euclidean distance applied to the t-test filtered dataset. A lag of three hours was applied when correlating.

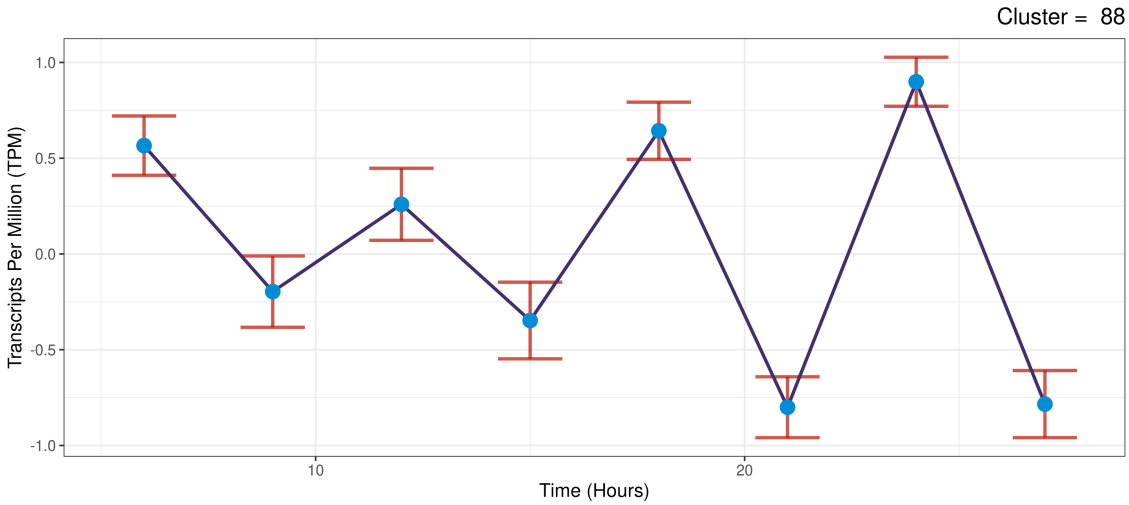


Figure 5.9:
Profile of cluster 88 from Cluster Correlation Network 4 which was shown to be strongly correlated with a version of itself lagged by three hours.

Chapter 6

Discussion

Cosinor Models

The technique of using cosinor models to detect rhythmicity was found to convincingly detect rhythmic genes with circadian periods. However, none of the core clock genes were found to be significant with the employed methodology: testing for significance by F-test. As such, it can be argued that this method is too conservative. Wu et al. have previously described a cosinor-based method which finds three core clock genes of *Carassius auratus* to be significant[54]. Their method found genes to be significantly rhythmic if they were found to be significant by ANOVA at the 5% significance level and if $\frac{SE(A)}{A} < 0.3$ where A is the amplitude of a cosinor model fitted to the gene and $SE\,(A)$ is the standard error of the amplitude. However, not all of the genes found to be significant by this method have convincingly rhythmic expression.

As previously mentioned, there is a notable lack of research on circatidal rhythms. No examples of using cosinor models could be found in published research. It is feasible this is the first time cosinor models have successfully been used as a method to detect circatidal rhythms. If this is the case then the use of extended cosinor models to find circatidal genes should be considered a remarkable development.

The core concept of cosinor models, fitting data to periodic models which frequently describe biological behaviour, has been developed further by the JTK_CYCLE and RAIN algorithms which fit models based on known circadian waveforms to gene expression data and assess the fit of these models[55, 56]. Sinusoidal waveforms are considered by both methods. JTK_CYCLE is limited to symmetric waveforms whilst RAIN can detect non-symmetric waveforms such as sawtooth waves. Whilst cosinor models limit rhythmicity detection to sinusoidal waveforms, this research has shown notable results can still be obtained. A cosinor model's status as a linear model is an advantage. The linearity property can be easily exploited by adding additional terms to the model, as is the case in the extended cosinor model, to further improve the model's fit: allowing rhythmic terms with different periods to be used.

In the case of extended cosinor models, the ability to have separate circadian and circatidal terms allows us to consider the circadian and circatidal clocks as separate mechanisms. This is useful given Wu et al. have provided evidence for these two mechanisms being independent[4].

Filtering

Filtering by t-tests was shown to be slightly quicker than ANOVA filtering but was also shown to produce a dataset with one less core clock gene when using the default parameters. As a new method, its necessity should be questioned. However, there are justifications for why filtering by t-tests should be used instead of contemporary methods.

1. Filtering by t-tests has explicitly been formulated with detecting rhythmic genes in mind. Ordinarily genes are filtered by using more general methods such as by ANOVA or by removing genes which have the lowest mean expression[57]. It may be beneficial, when investigating rhythmicity, to use a technique tailor-made for finding rhythmic data.
2. The additional parameters used in filtering by t-tests allows for more sophisticated tuning than ANOVA. The three parameters can be adjusted until a reasonable balance between the number of genes in the reduced dataset and the number of known genes of interest (such as core clock genes) is found.

Assumptions of gene expression data being normally distributed at the population level, as required when using a t-test, are commonly made. However, increasing evidence is being presented for this assumption being invalid[58, 59]. It would therefore be fruitful to produce a similar method which uses a non parametric alternative, such as the Mann-Whitney U test, instead.

Cluster Analysis

Clustering gene expression data is frequently practised and many examples of its use can be found in published research. Clustering is particularly useful in the context of gene expression data due to the structure of genes. In many cases, genes in the same cluster may share the same function, or are co-expressed, thus controlled by the same transcriptional regulatory mechanisms[60].

Previous work has suggested partitional clustering, such as PAM and k-means, is more accurate than hierarchical methods due to the former's robustness against outliers[61, 62]. However, in this analysis, strong evidence for DIANA and agglomerative hierarchical clustering being superior to PAM in general is provided via cluster validation results. A possible explanation for this result is the data have four replicates for each time point instead of the more common three replicates. In the case of the *T. saltator* dataset, the choice of having gene expression measurements across just 21 hours has resulted in an additional time point at each time point to be more financially feasible which has decreased the influence of outliers: a significant concern in noisy data such as gene expression.

In this work, evidence has been presented for absolute Pearson's correlation being a more ideal distance measure than Euclidean distance in the form of histograms showing the quantiles of distances between the centres of clusters. This evidence agrees with most existing literature which argues correlation based distance measures have better performance[63]. The general consensus seems to be in favour of of using Pearson's correlation over Euclidean distance. However, jackknife correlation is often used when single outliers are suspected. When calculating a jackknife correlation, each elements is temporarily removed one-by-one a correlation is found and then the element is returned. The lowest correlation found is the returned value. if there is a sudden decrease in correlation when an elements is removed then the removed value is likely to be an outlier. There is not a particularly strong argument for using jackknife correlation in the *T. saltator* dataset due to the aforementioned additional replicate measurements.

Whilst clustering allows us to explore gene expression data whilst not becoming 'lost' due to the quantity of the genes, each clustering algorithm introduces its own bias on the data[64]. However, the introduction of some bias seems to be an acceptable trade-off for the benefit of being able to identify which data are similar and see the data as a whole in a format which can be more easily understood by a human.

Correlation Analysis

It is of great importance to recognise correlation may signify up-regulation, may signify genes are being controlled by the same regulatory mechanism, or the existence of correlation may simply be due to randomness[60]. Therefore, any potential biological mechanisms between clusters which have been discussed should not be assumed to certainly exist. Practical biological experimentation is necessary to verify such mechanisms. It can clearly be seen how the visualisation of correlation networks, as presented in this work, could be used as the motivation for experiments between genes belonging to different clusters.

The method of constructing networks via genes or clusters used in this work is perhaps overly simplistic for data as complex as gene expression data. A more common approach, frequently used in the literature, is WeiGhted Correlation Network Analysis (WGCNA)[65]. The aim of WGCNA is to build co-expression networks where each node represents a gene's expression profile: similar to the correlation networks used in this work. When using WGCNA, it is also possible to find modules: clusters of highly interconnected genes. As previously, it is still possible strong correlation is found due to the noise inherent in gene expression data. WGCNA also allows fuzzy module membership where genes close to a shared boundary between modules are identifies as such. This is different to PAM, DIANA and agglomerative hierarchical clustering which only allows binary assignment (I.E in a cluster or not).

Being able to find modules via WGCNA is a powerful tool; modules have been found which relate to coronary artery disease and hyperlipidemia[66, 67]. The use of modules also enables readable visualisations to be produced.

Chapter 7

Reflection on the Project and Thoughts on Interdisciplinary Approaches

Potential for Future Research

Potential Based on Previous Work

During the course of this project, multiple opportunities for future research have been identified.

It has previously been reported *T. saltator* groups may influence the circadian rhythms of one another[18]. Evidence that social interactions can be a zeitgeber in the mammal species *Microtus oeconomus* has previously been produced[68]. However, this evidence did not explore the mechanisms on a molecular level. *T. saltator* seems to be an excellent candidate for investigating if social interactions can be a zeitgeber in a non-mammalian species and, if social interaction is indeed a zeitgeber for *T. saltator*, then the process could likely be investigated on the molecular level : given a trascriptome of *T. saltator* has already been assembled by O'Grady et al.[2]. An experiment could be established where *T. saltator* subjects are either isolated or in groups and the differences between gene expression could be analysed: possibly by using differential gene expression techniques.

The differences between gene expression in the antennae and the brain have previously been investigated by Ugolini et al.[19]. This investigation yielded the finding of a sun compass, which is likely located in the brain, and a lunar compass which is located in the antennae. However, analysis of gene expression data was largely limited to discovering if the core circadian genes could be found in samples taken from both brain and antennae tissue. Interesting results could likely be found by carrying out differential gene expression analysis. The key aim of differential genes expression (DGE) is to carry out statistical tests which decide if a gene's expression varies between different conditions[69]. As such, DGE could be used to find the difference in gene expression between antennae cells and brain cells across the entire *T. saltator* transcriptome. In DGE usually either a negative binomial or a log normal distribution is assumed and either a t-test, Wald test or a likelihood-ratio test is used to test for differences in genes.

Potential Based on This Project

If expanding on the results of this project, the method of correlating lagged clusters by their overall profile, identifying inter-cluster correlations of interest and then investigating the correlations between the genes in these clusters may produce promising results.

Alternatives to the distances metrics used could also be investigated. For instance, a distance metric based on Spearman's rank correlation, instead of Pearson's correlation, could be used. Jackknife distance has been found to effectively handle outliers[70], and could also be investigated further. Both of these distance metrics could be compared with the Euclidean distance and absolute Pearson's correlation distance metrics used in this project.

As previously noted, PAM clustering is incredibly slow when using large values of k as agglomerative hierarchical clustering was over 460 times faster than PAM for $k = 95$. If this research was to be used as the basis for additional research, a faster alternative to PAM, such as an alternative algorithm produced by Schubert and Rousseeuw should be considered[71]. The swap phase of this alternative method claims to be 200 times faster than PAM for $k = 100$. The result is noteworthy as the swap phase is the most time-consuming phase of PAM clustering. It was also claimed this alternative method returns results of comparable quality to that of PAM. However, this is a much newer algorithm and may have unexpected, and unreported eccentricities, which the user may be unaware of when using.

Attempts to find circatidal genes were mostly unsuccessful in this research. In particular a statistically robust approach to find circatidal genes, without finding the gene to be circadian first, was not found. However, the promising results for circadian period cosinor models suggests there is still potential for a cosinor-based approach to find circatidal genes. Creating a new model, based

on cosinor models, may yield promising results. As previously mentioned, the potential of extended cosinor models could be explored further.

The PAM algorithm allows the medoid of each cluster, the representative object for each cluster which is used to add additional objects to the cluster, to be specified by the user instead of randomly chosen until a local maximum of average dissimilarity has been found. However, PAM does not allow some medoids to be specified and then for additional medoids to be randomly chosen until the usual criteria is met. PAM could be modified to create a new algorithm where k is specified and m medoids are given by the user where $m < k$. $k - m$ medoids could then be randomly chosen until medoids are found such that a local maximum of dissimilarity is found. This algorithm could be employed with the specified medoids being the profiles of core clock genes. If k is suitably high then any genes which are assigned to the clusters generated by specified medoids may be clock controlled genes driven by the core clock gene whose profile acted as the medoid.

Thoughts on Systems Biology

The advent of high-throughput sequencing has produced huge quantities of data belonging to the '*-omics'* fields (genomics, proteomics, metabonomics, and transcriptomics) which explain aspects of biological systems. Ultimately, this means the invention of new technologies has produced new data which can answer pressing biological questions but often require new computational techniques to analyse fully. The analysis often leads to new insights being made by biologists. These insights then lead to new biological questions being asked which often require new technological advances to answer. As such, the progress in these fields is cyclical in terms of what is required to increase human-kind's understanding of biological processes. A great deal of cooperation is required by those who specialise in biomedical, statistical, computer science, engineering, bioinformatics, and many other disciplines to facilitate these developments.

Biological networks are complex. For instance, whilst information is transferred from DNA to RNA to proteins, as described by the central dogma of molecular biology, gene regulation can be regulated by protein phosphorylation, as previously discussed. This means an understanding of what is occurring at the protein level is required, in addition to the RNA level, to better understand gene expression. Interactions between these different levels is critical when understanding these biological networks. As a result, a field of study has emerged, systems biology, which focuses on taking a holistic approach to complex biological networks by considering networks as a whole[72]. Prior to systems biology, reductionist approaches, where only a small portion of the biological network instead of the greater whole, was the de facto approach. Whilst reductionist approaches have resulted in significant discoveries, it has been argued they are not effective in explaining why properties of a system emerge, thus limiting our understanding of these networks[73].

A hallmark of systems biology is the application of mathematical models to data which explain many elements of a biological network[74]. This necessitates multidisciplinary cooperation. However, it also means future progress is only limited by either a lack of technological advancement, a lack of new statistical techniques, or a lack of unanswered biological questions.

It seems unlikely technological advancements will not continue at a rapid pace. Advances in technology has massively reduced the price of sequencing an entire genome. Estimated as costing $100,000,000 USD in 2001, it now costs approximately $1,000 to sequence an entire genome. This has enabled large scale studies, such as PREdiCCt[75], where all of the participants in the study have their genome fully sequenced. Portable DNA sequencing kits are also available: allowing researcher to expedite their field work and carry out experiments which would otherwise be much more difficult or even impossible[76]. Advances in the power of computer hardware are also certain which will enable mathematical models which require additional computational power. This increase in power will also enable larger datasets to be used than those used presently.

Methods of statistically analysing biological data are also developing. In particular, Bayesian hierarchical models have rapidly grown in popularity in recent years and are being used to predict the progression of diseases and to estimate the time of a disease related event (such as a death or a flare which have promising applications in the fields of personalised medicine and precision medicine[77, 78, 79]. Whilst the full potential of these models has likely not yet been fully realised, it is almost certain either new methods will be developed or older methods will be revisited which will have newfound utility when being used with the wealth of biological data being generated by modern technology.

Finally, it seems even less likely that there will ever be a lack of biological questions which need to be answered both due to the real-world importance of such questions and due to the inquisitive nature of humankind.

(14981 total words)

Appendices

Appendix A

Installing and Using the `CircadianTools` Package

Steps before installing `CircadianTools`

Firstly, R version 3.4.4 or later should be installed: ideally on eithewr a Linux or Mac OS system.

Installing the `CircadianTools` R package is best done through the use of the `devtools` package as this will ensure all of the packages which CircadianTools depends on will also be installed.

Running the below code in R will install the `devtools` package.

```
install.packages("devtools")
```

Before `CircadianTools` can be installed, dependencies from the Bioconductor repository must also be installed. This can be done by using the following R command:

```
# Install BiocManager if not already installed
if (!requireNamespace("BiocManager", quietly = TRUE)){
  install.packages("BiocManager")
}

BiocManager::install("rain", "seqinr") # Install Bioconductor dependencies
```

Installing `CircadianTools`

Installing from GitHub

The most up-to-date version of `CircadianTools` can be installed from GitHub by entering the below line in to the R console:

```
devtools::install_github("nathansam/CircadianTools")
```

Installing from local files

Included with this document is the most recent version of the `CircadianTools` at the time of submission. This version of the package can be installed by the `devtools::install_local("path")` function where "path" is the path to the included "CircadianTools_1.0.0.tar.gz" file For example, if the "CircadianTools_1.0.0.tar.gz" file is saved to the user's Linux home directory then `CircadianTools` can be installed by the following command:

```
devtools::install_local("/home/USER/CircadianTools_1.0.0.tar.gz")
```

where USER is replaced with the name of the user account.

Documentation

Documentation is provided for `CircadianTools` via the R documentation system. Typing

```
library(CircadianTools)
?CircadianTools
```

will produce a list of all of the functions provided via `CircadianTools` and provide links to the documentation for these functions.

The documentation of each function provides a short description for the function, explains the arguments for the function, and provides an example of using the function. If user is unsure about a function used in one of the subsequent appendices, referring to the '`CircadianTools` 'documentation may alleviate this uncertainty.

Appendix B

Cosinor Models Code Appendix

Preliminaries

This appendix is the first of multiple appendices which present the code used to produce the results discussed in this document. There is a code appendix for each of the applicable chapters. This appendix presents the code used for the cosinor models chapter.

Firstly, the `CircadianTools` package was loaded. The count data (Laurasmappings) was read in and the CT18.4 column was removed. This column was removed as it is mostly zeroes due to a technical problem which occurred during the experiment. Any genes which showed zero activity for all readings were removed: as is customary. There were 91,311 genes in the dataset after removal.

```
library(CircadianTools)
# Read in count data
Laurasmappings <- read.csv(
  "~/MEGA/Uni/Masters/Diss/Stats/Raw_Data/Laurasmappings.csv",
  stringsAsFactors = FALSE)

Laurasmappings$CT18.4 <- NULL # Remove column of zeroes
Laurasmappings <- GeneClean(Laurasmappings) # Remove genes which show no activity
nrow(Laurasmappings) # Print the number of genes in the reduced dataset
```

```
## [1] 91311
```

Cosinor 24 Hour (Circadian) Period

Cosinor models with 24 hour periods were fitted and F-tests were carried out on these models. The resultant p-values are saved alongside gene names in the `cosinor.24` object. The `system.time` function is used to show the speed of fitting cosinor models to all 91,311 genes.

```
# Fit cosinor models with 24 hour period and calculate time taken.
system.time(cosinor.24 <- CosinorAnalysis(Laurasmappings, period = 24,
                                          progress = FALSE, print = FALSE)
            )
```

```
##    user  system elapsed
## 257.550   0.923 259.421
```

The p-values needed to be adjusted using a bonferroni correction due to the large number of genes being tested. After the adjustment, the genes which are significant at the 5% level were found.

```
# Apply bonferroni correction
cosinor.24$pVal <- p.adjust(cosinor.24$pVal, method = "bonferroni")
# Sort by smallest p-value
cosinor.24 <- cosinor.24[order(cosinor.24$pVal, decreasing = FALSE), ]
# Find genes significant at the 5% level
sig.24 <- subset(cosinor.24, pVal <= 0.05)
sig.24 # Print these genes
```

```
##                    sample         pVal
## 90568   comp99801_c1_seq1 2.246277e-06
## 57405   comp80445_c1_seq1 3.783256e-05
## 66624   comp89211_c0_seq2 1.469563e-04
## 81165   comp96806_c0_seq1 4.191294e-04
## 86918   comp98714_c0_seq2 6.070344e-04
## 81     comp100026_c0_seq2 6.175014e-04
## 8815   comp102333_c0_seq8 1.638949e-03
## 8813   comp102333_c0_seq2 2.707901e-03
## 28969   comp23718_c0_seq1 5.662856e-03
```

```
## 8814  comp102333_c0_seq21 6.844270e-03
## 6583   comp101772_c1_seq2 1.034375e-02
## 50942   comp71855_c0_seq1 1.053243e-02
## 86530   comp98599_c2_seq1 1.281385e-02
## 43801     comp606_c0_seq1 1.745201e-02
## 8816   comp102333_c0_seq9 4.192379e-02
```

A FASTA file was created using the FastaSub function in `CircadianTools`. This FASTA file was created by including only the sequences for the samples which were found to be significant with 24 hour cosinor models from a FASTA file which detailed the transcriptome of *Talitrus saltaor*. This FASTA file was then 'BLASTed' to search for proteins.

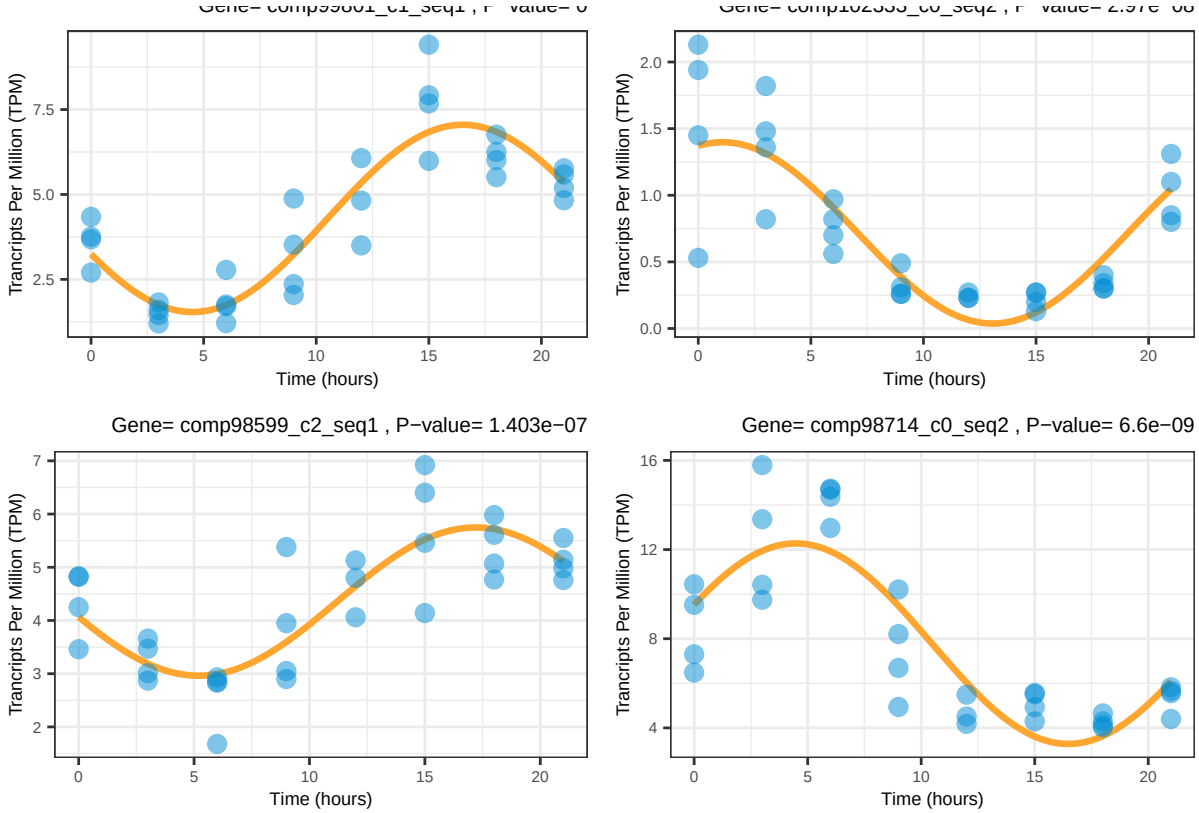
```
# Read in main FASTA file containing transcriptome
main.fasta <- seqinr::read.fasta(
  "~/MEGA/Uni/Masters/Diss/Stats/Raw_Data/JoesTranscriptomeMin300bp 2.fasta")

# Create a FASTA file which only contains sequences for the samples which were
# found to be significant with 24 hour cosinor models
sig.24.fasta <- FastaSub(gene.names = sig.24$sample, fasta.file = main.fasta,
         filename = "sig.24")
```

The below code saves plots of the cosinor models for these genes to a folder in the working directory called "cosinor_24".

```
CosinorSignificantPlot(cosinor.24, Laurasmappings, number = nrow(sig.24),
                       print = FALSE, save = TRUE, path = "cosinor_24")
```

After viewing the plots, four plots were chosen for inclusion in the main document and were saved as print quality png files.

```
gene.names <- c("comp99801_c1_seq1", "comp102333_c0_seq2", "comp98599_c2_seq1",
                "comp98714_c0_seq2" )

# Change the size of the text in the plots
txt.size <- ggplot2::theme(text = ggplot2::element_text(size = 7))


p1 <- CosinorPlot(gene.names[1], Laurasmappings) + txt.size
p2 <- CosinorPlot(gene.names[2], Laurasmappings) + txt.size
p3 <- CosinorPlot(gene.names[3], Laurasmappings) + txt.size
p4 <- CosinorPlot(gene.names[4], Laurasmappings) + txt.size

p <- gridExtra::grid.arrange(p1, p2, p3, p4, ncol = 2 , nrow = 2)
```



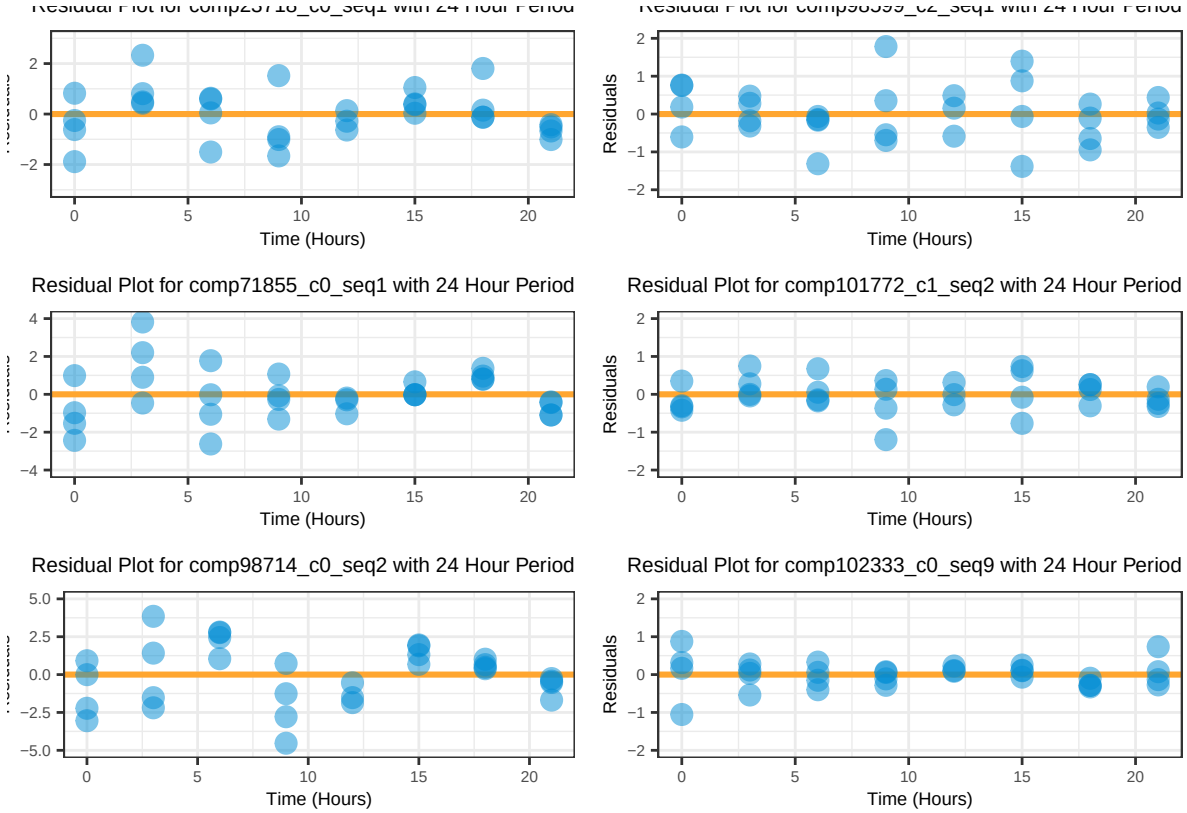

```
ggplot2::ggsave("cosinor_24.png", plot = p, width = 12, height = 10,
                units = "in")
```

The residuals for these significant cosinor models were plotted and saved to a folder in the working directory called “residuals”.

```
genes.sig <- GeneSub(sig.24, Laurasmappings)
CosinorResidualDatasetPlot(genes.sig, print = FALSE, save = TRUE,
                           path = "residuals")
```

After investigating the plots of the residuals, some genes appeared to have sinusoidal residuals whilst others did not. As a result, the plots of residuals for 6 genes were were saved for use as a figure. 3 of the genes were judged to have sinusoidal residuals whilst the remaining three were judged to not have this attribute.

```
p1 <- CosinorResidualPlot("comp23718_c0_seq1", Laurasmappings) + txt.size
p2 <- CosinorResidualPlot("comp71855_c0_seq1", Laurasmappings) + txt.size
p3 <- CosinorResidualPlot("comp98714_c0_seq2", Laurasmappings) + txt.size
p4 <- CosinorResidualPlot("comp98599_c2_seq1", Laurasmappings) + txt.size
p5 <- CosinorResidualPlot("comp101772_c1_seq2", Laurasmappings) + txt.size
p6 <- CosinorResidualPlot("comp102333_c0_seq9", Laurasmappings) + txt.size

arranged <- gridExtra::grid.arrange(p1,p4,p2,p5,p3,p6, ncol=2)
```




```
ggplot2::ggsave("residuals.png", arranged, width=12, height=10, units="in",
                dpi=300)
```

Investigating Genes with Sinusoidal Residuals

The three genes which were judged to have sinsoidal characteristics were then fitted with cosinor models with both circadian and circatidal terms. ANOVA was used to see if the addition of these extra terms resulted in a signficantly better fit.

Comp23718_c0_seq1

```
MultiCosinorTest("comp23718_c0_seq1", Laurasmappings, period.1 = 24,
                 period.2 = 12.4 )
```

```
## The anova table for the simple model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 50.411  50.411  50.044 1.073e-07 ***
## sss.1      1 13.673  13.673  13.573 0.0009732 ***
## Residuals 28 28.205   1.007
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The anova table for the more complex model is given by:
## Analysis of Variance Table
##
## Response: activity
```

```
##           Df Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 50.411  50.411 65.7513 1.372e-08 ***
## sss.1      1 13.673  13.673 17.8338 0.0002608 ***
## rrr.2      1  0.076   0.076  0.0997 0.7546551
## sss.2      1  8.195   8.195 10.6886 0.0030321 **
## Residuals 26 19.934   0.767
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Carrying out ANOVA on both models produces the following table :
## Analysis of Variance Table
##
## Model 1: activity ~ rrr.1 + sss.1
## Model 2: activity ~ rrr.1 + sss.1 + rrr.2 + sss.2
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     28 28.205
## 2     26 19.934  2    8.2714 5.3942 0.01098 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The ANOVA results have been saved.
```

Comp71855_c0_seq1

```
MultiCosinorTest("comp71855_c0_seq1", Laurasmappings, period.1 = 24,
                 period.2 = 12.4)
```

```
## The anova table for the simple model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 104.718 104.718 55.1337 4.364e-08 ***
## sss.1      1   8.569   8.569  4.5117   0.04263 *
## Residuals 28  53.182   1.899
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The anova table for the more complex model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 104.718 104.718 65.4088 1.441e-08 ***
## sss.1      1   8.569   8.569  5.3525   0.02886 *
## rrr.2      1   1.284   1.284  0.8020   0.37872
## sss.2      1  10.272  10.272  6.4163   0.01768 *
## Residuals 26  41.625   1.601
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Carrying out ANOVA on both models produces the following table :
## Analysis of Variance Table
##
## Model 1: activity ~ rrr.1 + sss.1
## Model 2: activity ~ rrr.1 + sss.1 + rrr.2 + sss.2
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     28 53.182
## 2     26 41.625  2    11.556 3.6091 0.04138 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The ANOVA results have been saved.
```

Comp98714_c0_seq2

```
MultiCosinorTest("comp98714_c0_seq2", Laurasmappings, period.1 = 24,
                 period.2 = 12.4)
```

```
## The anova table for the simple model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df Sum Sq Mean Sq F value   Pr(>F)
## rrr.1      1 274.78 274.776  68.132 5.54e-09 ***
## sss.1      1  45.69  45.690  11.329  0.00223 **
## Residuals 28 112.92   4.033
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The anova table for the more complex model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df  Sum Sq Mean Sq  F value    Pr(>F)
## rrr.1      1 274.776 274.776 106.0460 1.147e-10 ***
## sss.1      1  45.690  45.690  17.6335 0.0002776 ***
## rrr.2      1  12.303  12.303   4.7482 0.0385810 *
## sss.2      1  33.252  33.252  12.8332 0.0013752 **
## Residuals 26  67.369   2.591
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Carrying out ANOVA on both models produces the following table :
## Analysis of Variance Table
##
## Model 1: activity ~ rrr.1 + sss.1
## Model 2: activity ~ rrr.1 + sss.1 + rrr.2 + sss.2
##   Res.Df     RSS Df Sum of Sq      F   Pr(>F)
## 1     28 112.924
## 2     26  67.369  2    45.555 8.7907 0.001213 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The ANOVA results have been saved.
```

Investigating Genes With Non-Sinusoidal Residuals

The Three genes which were judged to not have sinusodial residuals were then investigated.

Comp98599_c2_seq1

```
MultiCosinorTest("comp98599_c2_seq1", Laurasmappings, period.1 = 24,
                 period.2 = 12.4)
```

```
## The anova table for the simple model is given by:
## Analysis of Variance Table
##
```

```
## Response: activity
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 29.6726 29.6726 56.0481 3.735e-08 ***
## sss.1      1  1.2592  1.2592  2.3784    0.1342
## Residuals 28 14.8236  0.5294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The anova table for the more complex model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 29.6726 29.6726 56.7985 5.317e-08 ***
## sss.1      1  1.2592  1.2592  2.4103    0.1326
## rrr.2      1  1.1592  1.1592  2.2189    0.1484
## sss.2      1  0.0815  0.0815  0.1560    0.6960
## Residuals 26 13.5829  0.5224
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Carrying out ANOVA on both models produces the following table :
## Analysis of Variance Table
##
## Model 1: activity ~ rrr.1 + sss.1
## Model 2: activity ~ rrr.1 + sss.1 + rrr.2 + sss.2
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     28 14.824
## 2     26 13.583  2    1.2407 1.1875  0.321
## The ANOVA results have been saved.
```

Comp101772_c1_seq2

```
MultiCosinorTest("comp101772_c1_seq2", Laurasmappings, period.1 = 24,
                 period.2 = 12.4)
```

```
## The anova table for the simple model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 4.5071  4.5071  22.872 5.027e-05 ***
## sss.1      1 7.2688  7.2688  36.887 1.502e-06 ***
## Residuals 28 5.5176  0.1971
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The anova table for the more complex model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 4.5071  4.5071 24.3783 3.964e-05 ***
## sss.1      1 7.2688  7.2688 39.3165 1.231e-06 ***
## rrr.2      1 0.0211  0.0211  0.1142   0.73808
## sss.2      1 0.6896  0.6896  3.7301   0.06441 .
## Residuals 26 4.8069  0.1849
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Carrying out ANOVA on both models produces the following table :
## Analysis of Variance Table
##
## Model 1: activity ~ rrr.1 + sss.1
## Model 2: activity ~ rrr.1 + sss.1 + rrr.2 + sss.2
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     28 5.5176
## 2     26 4.8069  2   0.71074 1.9222 0.1665
## The ANOVA results have been saved.
```

Comp102333_c0_seq9

```
MultiCosinorTest("comp102333_c0_seq9", Laurasmappings, period.1 = 24,
                 period.2 = 12.4)
```

```
## The anova table for the simple model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 0.2780  0.2780  2.0105    0.1672
## sss.1      1 6.8318  6.8318 49.3998 1.208e-07 ***
## Residuals 28 3.8723  0.1383
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## The anova table for the more complex model is given by:
## Analysis of Variance Table
##
## Response: activity
##           Df Sum Sq Mean Sq F value    Pr(>F)
## rrr.1      1 0.2780  0.2780  1.9827    0.1710
## sss.1      1 6.8318  6.8318 48.7158 2.068e-07 ***
## rrr.2      1 0.2071  0.2071  1.4768    0.2352
## sss.2      1 0.0190  0.0190  0.1356    0.7157
## Residuals 26 3.6462  0.1402
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Carrying out ANOVA on both models produces the following table :
## Analysis of Variance Table
##
## Model 1: activity ~ rrr.1 + sss.1
## Model 2: activity ~ rrr.1 + sss.1 + rrr.2 + sss.2
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     28 3.8723
## 2     26 3.6462  2   0.22611 0.8062 0.4574
## The ANOVA results have been saved.
```

Cosinor 12.4 (Circatidal) Period

Cosinor models with periods of 12.4 hours were then be fitted. The 25 best fitting models were then found.

```
cosinor.12.4 <- CosinorAnalysis(Laurasmappings, period = 12.4, progress = FALSE,
                                print= FALSE )
cosinor.12.4 <- cosinor.12.4[order(cosinor.12.4$pVal, decreasing = FALSE), ]
```

```
sig.12.4 <- cosinor.12.4[1:25, ]
sig.12.4
```

```
##                    sample         pVal
## 84017  comp97780_c0_seq11 0.0001188788
## 40770   comp53017_c0_seq2 0.0001813181
## 47742   comp66784_c0_seq1 0.0001975727
## 32217   comp28197_c0_seq1 0.0002257871
## 13449 comp1187601_c0_seq1 0.0003098192
## 40902  comp533893_c0_seq1 0.0003403444
## 60758   comp84247_c0_seq1 0.0003462262
## 20107   comp15545_c0_seq1 0.0003619025
## 82934   comp97430_c3_seq8 0.0003650143
## 39427  comp489773_c0_seq1 0.0003777484
## 89458   comp99487_c0_seq2 0.0003795692
## 57101   comp80084_c0_seq1 0.0003968281
## 36320   comp37769_c0_seq1 0.0003979793
## 71437   comp92563_c0_seq4 0.0004666427
## 23690   comp17963_c0_seq1 0.0004721746
## 54310   comp76855_c0_seq2 0.0005164138
## 88158   comp99093_c0_seq2 0.0006021586
## 78647   comp95889_c0_seq2 0.0006182605
## 85027  comp98126_c0_seq33 0.0006260849
## 13526 comp1189457_c0_seq1 0.0006328214
## 63453   comp86536_c0_seq1 0.0006437563
## 46227   comp64680_c0_seq1 0.0006444333
## 16688 comp1372561_c0_seq1 0.0006836282
## 44567  comp617341_c0_seq1 0.0007331367
## 86925   comp98716_c0_seq1 0.0007550429
```

A FASTA file was created using the FastaSub function in `CircadianTools` similar to the signficant circadian genes was used for a BLAST search.

```
sig.12.4.fasta <- FastaSub(gene.names = sig.12.4$sample, fasta.file = main.fasta,
         filename = "sig.24")
```

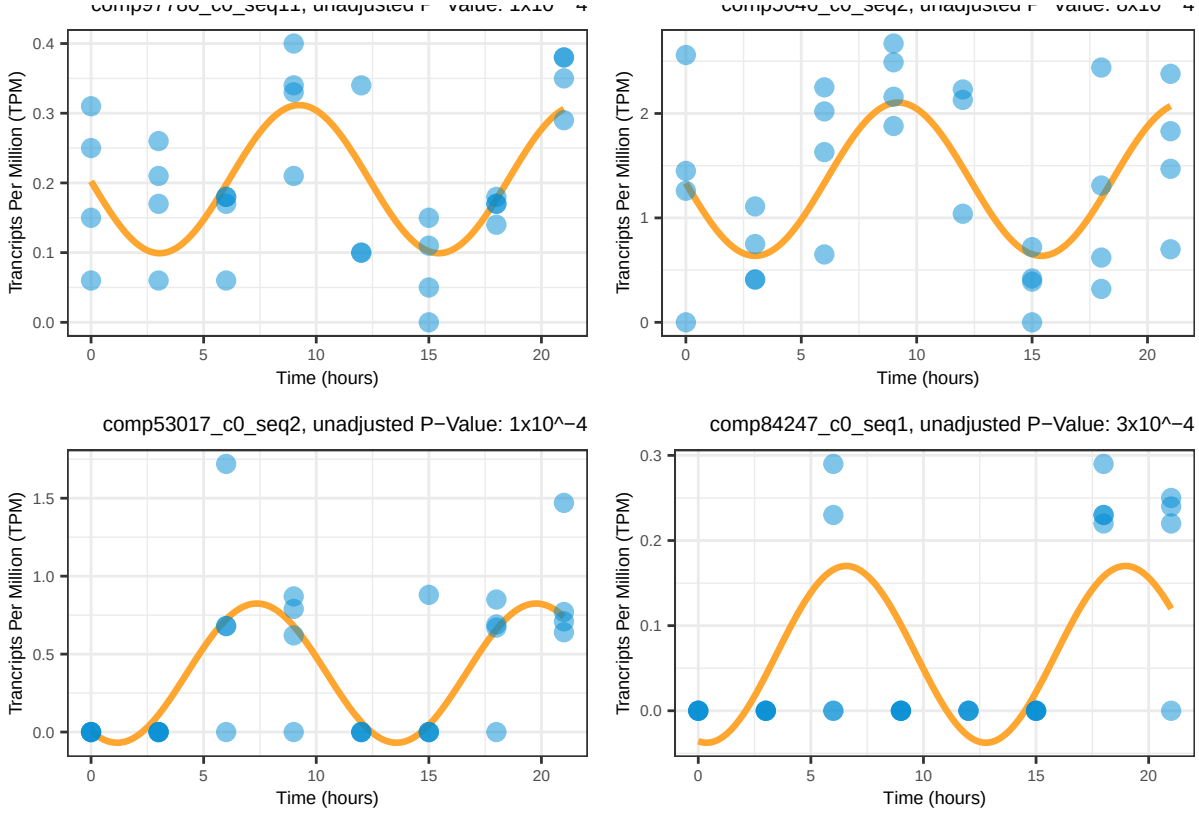
Plots of the best fitting genes were plotted and saved to the `cosinor_12.4` folder in the working directory so they could be judged for circatidal characteristics.

```
CosinorSignificantPlot(cosinor.12.4, Laurasmappings, period = 12.4 ,number = 25,
                       print = FALSE, save = TRUE, path = "cosinor_12.4")
```

After viewing the plots, 4 were chosen for inclusion in the main document.

```
p1 <- CosinorPlot("comp97780_c0_seq11", Laurasmappings, period=12.4) +
        ggplot2::ggtitle("comp97780_c0_seq11, unadjusted P-Value: 1x10^-4")  +
        txt.size
p2 <- CosinorPlot("comp5046_c0_seq2", Laurasmappings, period=12.4) +
        ggplot2::ggtitle("comp5046_c0_seq2, unadjusted P-Value: 8x10^-4") +
        txt.size
p3 <- CosinorPlot("comp53017_c0_seq2", Laurasmappings, period=12.4) +
        ggplot2::ggtitle("comp53017_c0_seq2, unadjusted P-Value: 1x10^-4") +
        txt.size
p4 <- CosinorPlot("comp84247_c0_seq1", Laurasmappings, period=12.4) +
        ggplot2::ggtitle("comp84247_c0_seq1, unadjusted P-Value: 3x10^-4") +
        txt.size
```

```
arranged <- gridExtra::grid.arrange(p1,p2,p3,p4, ncol=2)
```




```
ggplot2::ggsave("cosinor_12.4.png", arranged,width=12, height=10, units="in",
                dpi=300)
```

Appendix C

Filtering Code Appendix

Preliminaries

The initial set-up for filtering was similar to the Cosinor code appendix. The `CircadianTools` package was loaded. The count data (Laurasmappings) was read in and the CT18.4 column was removed due to this column showing no activity. Any genes which show zero activity for all readings were removed: as is customary.

```
library(CircadianTools)
Laurasmappings <- read.csv("~/MEGA/Uni/Masters/Diss/Stats/Raw_Data/Laurasmappings.csv",
                           stringsAsFactors = FALSE) # Read in count data
Laurasmappings$CT18.4 <- NULL # Remove column of zeroes
Laurasmappings <- GeneClean(Laurasmappings) # Remove genes which show no activity
```

For this code appendix, an additional preliminary steps was taken: `circadian.csv`, which contains the names of genes found to be circadian via BLAST, was read in.

```
circadian <- read.csv("~/MEGA/Uni/Masters/Diss/Stats/circadian.csv", sep="",
                      stringsAsFactors=FALSE)
```

ANOVA Filtering

Threshold of 5%

Firstly, the genes were filtered using ANOVA with a signficance level of 5%.

```
system.time(a.filter <- AnovaFilter(dataset = Laurasmappings, threshold = 0.05))
```

```
##    user  system elapsed
##  35.170   3.937  81.870
```

```
anova.no <- nrow(a.filter) # Number of genes in the reduced dataset
circadian.subset <- GeneSub(circadian, a.filter)
circadian.no <- nrow(circadian.subset)



cat(paste("There are", anova.no,
          "genes in the dataset filtered via ANOVA with a threshold of 5%.\n",
          circadian.no, "of the", nrow(circadian),
         "genes found to be circadian via BLAST are in the reduced dataset.\n
         These genes are: \n")
    )
```

```
## There are 11186 genes in the dataset filtered via ANOVA with a threshold of 5%.
##  5 of the 18 genes found to be circadian via BLAST are in the reduced dataset.
##
##          These genes are:
```

```
circadian.subset$sample
```

```
## [1] "comp97405_c0_seq1"  "comp99101_c0_seq3"  "comp102279_c0_seq7"
## [4] "comp102609_c0_seq3" "comp939723_c0_seq1"
```

Threshold of 2.5%

As 11,186 genes is still too many genes for some computational methods, a threshold of 2.5% was considered.

```
system.time(a.filter <- AnovaFilter(dataset = Laurasmappings,
                                    threshold = 0.025)
            )
```

```
##    user  system elapsed
##  34.934   3.679  81.315
```

```
anova.no <- nrow(a.filter) # Number of genes in the reduced dataset
circadian.subset <- GeneSub(circadian, a.filter)
circadian.no <- nrow(circadian.subset)



cat(paste("There are", anova.no,
          "genes in the dataset filtered via ANOVA with a threshold of 2.5%.\n",
          circadian.no, "of the", nrow(circadian),
         "genes found to be circadian via BLAST are in the reduced dataset.\n
         These genes are: \n")
    )
```

```
## There are 7124 genes in the dataset filtered via ANOVA with a threshold of 2.5%.
##  3 of the 18 genes found to be circadian via BLAST are in the reduced dataset.
##
##          These genes are:
```

```
circadian.subset$sample
```

```
## [1] "comp97405_c0_seq1"  "comp99101_c0_seq3"  "comp102279_c0_seq7"
```

The reduced dataset was saved as a csv file for use in later chapters.

```
write.csv(a.filter,"a_filter.csv", row.names = FALSE )
```

T-Test Filtering

Filtering via t-tests, as presented in the main document was then used to filter the count data.

```
system.time(t.filter <- TFilter(Laurasmappings, maxdifference = 1,
                                minchanges = 2, psignificance = 0.05)
            )
```

```
##    user  system elapsed
##  34.960   4.088  82.536
```

```
circadian.subset <- GeneSub(circadian, t.filter)
circadian.no <- nrow(circadian.subset)

t.filter.no <- nrow(t.filter) # Number of genes in the reduced dataset
circadian.no <- nrow(GeneSub(circadian, t.filter))

cat(paste("There are",t.filter.no,
          "genes in the dataset filtered via t-tests. \n",
          circadian.no, "of the", nrow(circadian),
         "genes found to be circadian via BLAST are in the reduced dataset.\n
         These genes are: \n"))
```

```
## There are 6294 genes in the dataset filtered via t-tests.
##  2 of the 18 genes found to be circadian via BLAST are in the reduced dataset.
```

```
##
##          These genes are:
```

```
circadian.subset$sample
```

```
## [1] "comp100937_c0_seq1" "comp939723_c0_seq1"
```

The reduced dataset was saved as a csv file for use in later chapters.

```
write.csv(t.filter,"t_filter.csv", row.names = FALSE )
```

T-Test Experimentation

As filtering by using the t-tests methood involves three important paramters, these parameters were each varied whilst fixing the other two in order to see the effect of varying the paramters.

p

```
p.values <- c(0.05, 0.025, 0.01)

p.value.results <- data.frame()
 for (i in p.values){
  filtered <- TFilter(Laurasmappings, maxdifference = 1, minchanges = 2,
                      psignificance = i)

  p.value.results <- rbind (p.value.results,
                            data.frame(p = i, genecount = nrow(filtered)
                                       )
                            )
}

print(p.value.results)
```

```
##       p genecount
## 1 0.050      6294
## 2 0.025      2643
## 3 0.010       805
```

Minimum Significant Changes

```
changes <- 1 : 5

sig.change.results <- data.frame()

for (i in changes){
  filtered <- TFilter(Laurasmappings, maxdifference = 1, minchanges = i,
                      psignificance = 0.05)
  sig.change.results <- rbind(sig.change.results,
                              data.frame(min.changes = i,
                                         genecount = nrow(filtered)
                                         )
                              )
}

print(sig.change.results)
```

```
##   min.changes genecount
```

```
## 1           1     22614
## 2           2      6294
## 3           3      1775
## 4           4       259
## 5           5        55
```

Maximum Difference Between Signficant Changes

```
max.diff <- 0 : 5

max.diff.results <- data.frame()

for (i in max.diff){
  filtered <- TFilter(Laurasmappings, maxdifference = i, minchanges = 2,
                      psignificance = 0.05)
  max.diff.results <- rbind(max.diff.results,
                            data.frame(max.diff = i,
                                       genecount = nrow(filtered)
                                       )
                            )
}


print(max.diff.results)
```

```
##   max.diff genecount
## 1        0      4724
## 2        1      6294
## 3        2      7297
## 4        3      7312
## 5        4      7312
## 6        5      7312
```

Comparing T-Test and ANOVA Filtering

The number of genes found in both the ANOVA filtered and t-test filtered datasets were then found.

```
shared.no <- nrow(GeneSub(a.filter, t.filter))
cat(paste( "There are", shared.no,
           "genes which can be found in both reduced datasets.\n"))
```

```
## There are 3097 genes which can be found in both reduced datasets.
```

Appendix D

Cluster Analysis Code Appendix

Preliminaries

The `CircadianTools` package was loaded first. The ANOVA and t-test filtered datasets from the previous chapter were read in. Both datasets were scaled and centered.

```
library(CircadianTools)
a.filter <- read.csv(
  "~/MEGA/Uni/Masters/Diss/Stats/R Markdown/filtering/a_filter.csv",
  stringsAsFactors=FALSE)
a.filter <-GeneScale(a.filter)
t.filter <- read.csv(
  "~/MEGA/Uni/Masters/Diss/Stats/R Markdown/filtering/t_filter.csv",
  stringsAsFactors=FALSE)
t.filter <-GeneScale(t.filter)
```

The seed was set to 123 for reproducibility purposes.

```
set.seed(123)
```

Cluster Validation

Plots of cluster validation metric scores against k (the number of clusters in the partition) were plotted for PAM, agglomerative hierarchical clustering and DIANA for the ANOVA and t-test filtered datasets. Both Euclidean distance and absolute Pearson's correlation were considered.

Due to the huge amount of computational power required to create the cluster validation plots, the plots were generated using Google Cloud virtual machines with 24 cores. As a result, the cluster validation code will not be run again for this appendix. However, the code used has been included.

The values of k considered were first created.

```
k.params <- c(seq(2,5), seq(10, 300, 5))
```

ANOVA Filtered with Euclidean Distance

```
# Use the maximum number of processor threads for calculations
nthreads <- parallel::detectCores()

ClusterParamSelection(a.filter, k = k.params, metric="euclidean",
                      nthreads = nthreads, path = "anova_euclid_validation")
```

ANOVA Filtered with Absolute Pearson's Correlation

```
ClusterParamSelection(a.filter, k = k.params, metric="abs.correlation",
                      nthreads = nthreads, path = "anova_abscor_validation")
```

T-Test Filtered with Euclidean Distance

```
ClusterParamSelection(t.filter, k = k.params, metric="euclidean",
                      nthreads = nthreads, path = "ttest_euclid_validation")
```

T-Test Filtered with Absolute Pearson's Correlation

```
ClusterParamSelection(t.filter, k = k.params, metric="abs.correlation",
                      nthreads = nthreads, path = "ttest_abscor_validation")
```

Generating the Clusters For Use in Future Methods

After consulting the cluster validation plots, k = 95 was decided to be the best value to use. Clustering partitions were then generated and saved to csv files.

ANOVA Filtered

The distance matrices using either Euclidean distance or absolute Pearson's Correlation were calculated for the ANOVA filtered dataset.

```
a.euc <- DistanceGen(a.filter, metric = "euclidean")
a.cor <- DistanceGen(a.filter, metric = "abs.correlation")
```

PAM Clustering

```
system.time(a.pam.euc <- PamClustering(distance = a.euc, k = 95, scale = TRUE)
            )
```

```
##     user   system  elapsed
## 4877.809    0.418 4888.552
```

```
a.pam.euc <- cbind(a.filter, cluster = a.pam.euc$cluster)
write.csv(a.pam.euc, file = "a_pam_euc.csv", row.names = FALSE)
a.pam.cor <- PamClustering(distance = a.cor, k = 95, scale = TRUE)
a.pam.cor <- cbind(a.filter, cluster = a.pam.cor$cluster)
write.csv(a.pam.cor, file = "a_pam_cor.csv", row.names = FALSE)
```

Agglomerative Hierarchical Clustering

```
system.time(a.agglom.euc <- AgglomClustering(distance = a.euc, k = 95,
                                             scale = TRUE)
            )
```

```
##    user  system elapsed
##   2.032   0.155   7.634
```

```
a.agglom.euc <- cbind(a.filter, cluster = a.agglom.euc$cluster)
write.csv(a.agglom.euc, file = "a_agglom_euc.csv", row.names = FALSE )
a.agglom.cor <- AgglomClustering(distance = a.cor, k = 95, scale = TRUE)
a.agglom.cor <- cbind(a.filter, cluster = a.agglom.cor$cluster)
write.csv(a.agglom.cor, file = "a_agglom_cor.csv", row.names = FALSE)
```

DIANA Clustering

```
system.time(a.diana.euc <- DianaClustering(distance = a.euc, k = 95,
                                           scale = TRUE)
                  )
```

```
##    user  system elapsed
## 628.248   0.342 629.667
```

```
a.diana.euc <- cbind(a.filter, cluster = a.diana.euc$cluster)
write.csv(a.diana.euc, file = "a_diana_euc.csv", row.names = FALSE)
a.diana.cor <- DianaClustering(distance = a.cor, k = 95, scale = TRUE)
a.diana.cor <- cbind(a.filter, cluster = a.diana.cor$cluster)
write.csv(a.diana.cor, file = "a_diana_cor.csv", row.names = FALSE)
```

T-Test Filtered

Similar calculations were then run for t-test filtered dataset. The distance matrices for Euclidean distance and absolute Pearson's Correlation were calculated.

```
t.euc <- DistanceGen(t.filter, metric = "euclidean")
t.cor <- DistanceGen(t.filter, metric = "abs.correlation")
```

PAM

```
t.pam.euc <- PamClustering(distance = t.euc, k = 95, scale = TRUE)
t.pam.euc <- cbind(t.filter, cluster = t.pam.euc$cluster)
write.csv(t.pam.euc, file = "t_pam_euc.csv", row.names = FALSE)
t.pam.cor <- PamClustering(distance = t.cor, k = 95, scale = TRUE)
t.pam.cor <- cbind(t.filter, cluster = t.pam.cor$cluster)
write.csv(t.pam.cor, file = "t_pam_cor.csv", row.names = FALSE)
```

Agglomerative Hierarchical

```
t.agglom.euc <- AgglomClustering(distance = t.euc, k = 95, scale = TRUE)
t.agglom.euc <- cbind(t.filter, cluster = t.agglom.euc$cluster)
write.csv(t.agglom.euc, file = "t_agglom_euc.csv", row.names = FALSE)
t.agglom.cor <- AgglomClustering(distance = t.cor, k = 95, scale = TRUE)
t.agglom.cor <- cbind(t.filter, cluster = t.agglom.cor$cluster)
write.csv(t.agglom.cor, file = "t_agglom_cor.csv", row.names = FALSE)
```

DIANA

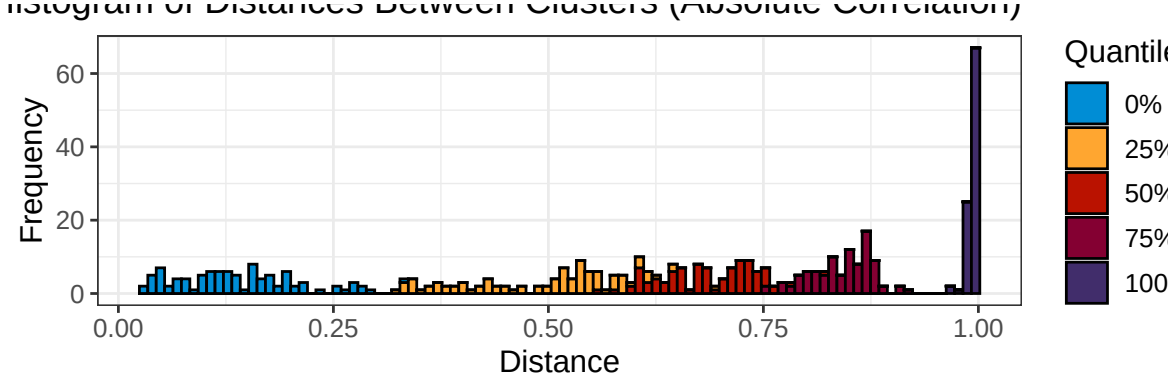
```
t.diana.euc <- DianaClustering(distance = t.euc, k = 95,
                                          scale = TRUE)
t.diana.euc <- cbind(t.filter, cluster = t.diana.euc$cluster)
write.csv(t.diana.euc, file = "t_diana_euc.csv", row.names = FALSE)
t.diana.cor <- DianaClustering(distance = t.cor, k = 95, scale = TRUE)
t.diana.cor <- cbind(t.filter, cluster = t.diana.cor$cluster )
write.csv(t.diana.cor, file = "t_diana_cor.csv", row.names = FALSE)
```

Comparing Distance Metrics

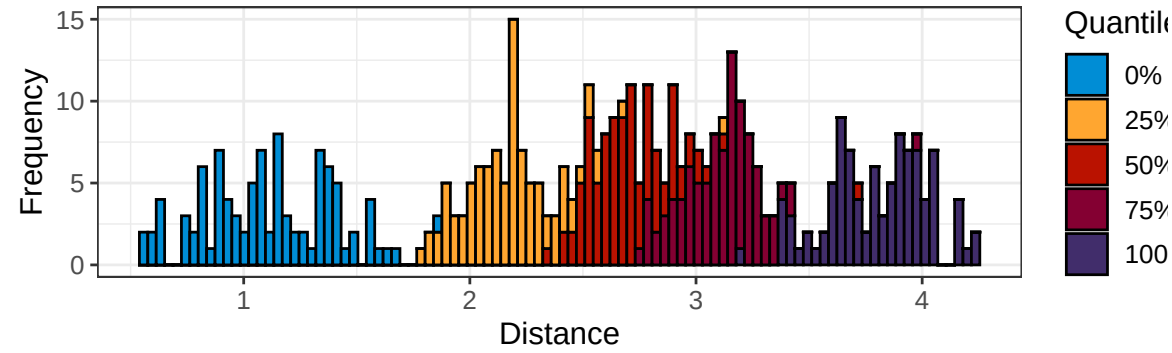
In order to compare the effect of using different distance metrics, the distances between the centers of clusters generated via using the two metrics were plotted used histograms.

```
quantiles <- CircadianTools::FindClusterDistanceQuantiles(
  cluster.dataset = a.agglom.cor, metric = "abs.correlation",
  nthreads = 12)

quantiles.plot <- reshape2::melt(data = quantiles,
                                 measure.vars = c("0%", "25%", "50%", "75%", "100%")
                                 )
colnames(quantiles.plot) <- c("results", "Quantile", "Distance")
# Vector of colours used in package
colours.vector <- c("#008dd5", "#ffa630", "#ba1200", "#840032", "#412d6b")

p1 <- ggplot2::ggplot(data = quantiles.plot, ggplot2::aes(x = Distance,
                                                          fill = Quantile)
                      )
p1 <- p1 + ggplot2::geom_histogram(color = "black", bins = 100)
```

```
p1 <- p1 + ggplot2::scale_fill_manual(values = colours.vector)
p1 <- p1 + ggplot2::theme_bw() + ggplot2::ylab("Frequency")
p1 <- p1 + ggplot2::ggtitle("Histogram of Distances Between Clusters (Absolute Correlation)")
p1 <- p1 + ggplot2::theme(plot.title = ggplot2::element_text(hjust = 1))
p1 <- p1 + ggplot2::theme(text = ggplot2::element_text(size = 12))




quantiles <- CircadianTools::FindClusterDistanceQuantiles(
  cluster.dataset = a.agglom.euc, metric = "euclidean", nthreads = 12)

quantiles.plot <- reshape2::melt(data = quantiles,
                                 measure.vars = c("0%", "25%", "50%",
                                                  "75%", "100%")
                                 )
colnames(quantiles.plot) <- c("results", "Quantile", "Distance")


p2 <- ggplot2::ggplot(data = quantiles.plot, ggplot2::aes(x = Distance,
                                                          fill = Quantile)
                      )
p2 <- p2 + ggplot2::geom_histogram(color = "black", bins = 100)
p2 <- p2 + ggplot2::scale_fill_manual(values = colours.vector)
p2 <- p2 + ggplot2::theme_bw() + ggplot2::ylab("Frequency")
p2 <- p2 +  ggplot2::ggtitle("Histogram of Distances Between Clusters (Euclidean Distance)")
p2 <- p2 + ggplot2::theme(plot.title = ggplot2::element_text(hjust = 1))
p2 <- p2 + ggplot2::theme(text = ggplot2::element_text(size = 12))

p <- gridExtra::grid.arrange(p1,p2, nrow = 2)
```



Histogram of Distances Between Clusters (Euclidean Distance)




```
ggplot2::ggsave("Quantile_comp.png", plot = p, width = 8.27,
                height = 11.69, units = "in")  # Save the plot
```

Appendix E

Correlation Analysis Code Appendix

Preliminaries

First, the `CircadianTools` package was loaded.

```
library(CircadianTools)
```

In order to keep this appendix concise, only the code used to produce the correlation networks included in the main document will be included. The below code can be easily expanded to include all of the correlation networks which were visualised (which can still be found at users.aber.ac.uk/nsc/cytoscape.html).

Gene correlation network

The below code generates the file read into Cytoscape to create the correlation networks between genes.

```
# Read in the cluster dataset
t.filter <- read.csv("~/Stats/R Markdown/filtering/t_filter.csv",
                     stringsAsFactors = FALSE)


# Find lagged correlations (returns an object in short form format)
cor.lag.1 <- CorAnalysisDataset(t.filter, lag = 1, save = FALSE)
# Convert to long form format
cyto.cor.lag.1 <- CytoscapeFile(cor.lag.1, save = FALSE)
# Filter out weak correlations and write data as a csv file
filtered <- CytoscapeFilter(cyto.cor.lag.1, threshold = 0.95, save = TRUE,
                            filename = "t.filter.lag1")
# Clear objects in the workspace to reduce RAM usage
rm(list=ls())
```

Cluster correlation networks

For each visualisation, the required clustering results from the clustering appendix were read in. The cluster profiles were then found and correlated with an applied lag. The results were initally in a short form format so were converted to a long form format as this is what Cytoscape expects. The files were then filtered to include correlations greater than 0.95 or less that -0.95. The files were saved and objects in the workspace were then cleared in order to minimise RAM usage.

ANOVA Filtered, Agglomerative Clustering, Absolute Correlation, Lag of 1

```
# Read in the cluster dataset
a.agglom.cor <- read.csv( "~/Stats/R Markdown/clustering/a_agglom_cor.csv",
                          stringsAsFactors=FALSE)

# Find lagged correlations (returns an object in short form format)
cor.lag.1 <- CorAnalysisClusterDataset(a.agglom.cor, lag = 1, save = FALSE)
# Convert to long form format
cyto.cor.lag.1 <- CytoscapeFile(cor.lag.1, save = FALSE)
# Filter out weak correlations and write data as a csv file
filtered <- CytoscapeFilter(cyto.cor.lag.1, threshold = 0.95, save = TRUE,
                            filename = "a.agglom.cor.lag1")
# Clear objects in the workspace to reduce RAM usage
rm(list=ls())
```

ANOVA Filtered, DIANA Clustering, Absolute Correlation, Lag of 2

```
# Read in the cluster dataset
a.diana.cor <- read.csv("~/Stats/R Markdown/clustering/a_diana_cor.csv",
                        stringsAsFactors=FALSE)

# Find lagged correlations (returns an object in short form format)
cor.lag.2 <- CorAnalysisClusterDataset(a.diana.cor, lag = 2, save = FALSE)
# Convert to long form format
cyto.cor.lag.2 <- CytoscapeFile(cor.lag.2, save = FALSE)
# Filter out weak correlations and write data as a csv file
filtered <- CytoscapeFilter(cyto.cor.lag.2, threshold = 0.95, save = TRUE,
                            filename = "a.diana.cor.lag2")
# Clear objects in the workspace to reduce RAM usage
rm(list=ls())
```

ANOVA Filtered, PAM Clustering, Euclidean Distance, Lag of 2

```
# Read in the cluster dataset
a.pam.euc <- read.csv("~/Stats/R Markdown/clustering/a_pam_euc.csv",
                      stringsAsFactors=FALSE)

# Find lagged correlations (returns an object in short form format)
cor.lag.2 <- CorAnalysisClusterDataset(a.pam.euc, lag = 2, save = FALSE)
# Convert to long form format
cyto.cor.lag.2 <- CytoscapeFile(cor.lag.2, save = FALSE)
# Filter out weak correlations and write data as a csv file
filtered <- CytoscapeFilter(cyto.cor.lag.2, threshold = 0.95, save = TRUE,
                            filename = "a.pam.euc.lag2")
# Clear objects in the workspace to reduce RAM usage
rm(list=ls())
```

T-Test Filtered, Agglomerative Clustering, Euclidean distance, Lag of 1

```
t.agglom.euc <- read.csv("~/Stats/R Markdown/clustering/t_agglom_euc.csv",
                         stringsAsFactors=FALSE)

# Find lagged correlations (returns an object in short form format)
cor.lag.1 <- CorAnalysisClusterDataset(t.agglom.euc, lag = 1, save = FALSE)
# Convert to long form format
cyto.cor.lag.1 <- CytoscapeFile(cor.lag.1, save = FALSE)
# Filter out weak correlations and write data as a csv file
filtered <- CytoscapeFilter(cyto.cor.lag.1, threshold = 0.95, save = TRUE,
                            filename = "t.agglom.euc.lag1")
# Clear objects in the workspace to reduce RAM usage
rm(list=ls())
```

These files were then opened in Cytoscape and networks were produced. The Cytoscape session files can also be found alongside the other supporting files provided.

Bibliography

[1] A. Sboner, X. J. Mu, D. Greenbaum, R. K. Auerbach, and M. B. Gerstein, "The real cost of sequencing: higher than you think!," *Genome Biology*, vol. 12, p. 125, Aug 2011.

[2] J. F. O'Grady, L. S. Hoelters, M. T. Swain, and D. C. Wilcockson, "Identification and temporal expression of putative circadian clock transcripts in the amphipod crustacean *Talitrus saltator*," *PeerJ*, vol. 4, p. e2555, Oct. 2016.

[3] W. Huang, S. Richards, M. A. Carbone, D. Zhu, R. R. H. Anholt, J. F. Ayroles, L. Duncan, K. W. Jordan, F. Lawrence, M. M. Magwire, C. B. Warner, K. Blankenburg, Y. Han, M. Javaid, J. Jayaseelan, S. N. Jhangiani, D. Muzny, F. Ongeri, L. Perales, Y.-Q. Wu, Y. Zhang, X. Zou, E. A. Stone, R. A. Gibbs, and T. F. C. Mackay, "Epistasis dominates the genetic architecture of drosophila quantitative traits," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 109, pp. 15553–15559, Sep 2012. 22949659[pmid].

[4] L. Zhang, M. Hastings, E. Green, E. Tauber, M. Sladek, S. Webster, C. Kyriacou, and D. Wilcockson, "Dissociation of circadian and circatidal timekeeping in the marine crustacean eurydice pulchra," *Current Biology*, vol. 23, no. 19, pp. 1863 – 1873, 2013.

[5] R Core Team, *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019.

[6] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

[7] I. Korf, M. Yandell, and J. Bedell, *BLAST*, ch. 1, p. 3. Sebastopol, CA: O'Reilly & Associates, 1st ed. ed., 2003.

[8] Microsoft and S. Weston, *foreach: Provides Foreach Looping Construct for R*, 2017. R package version 1.4.4.

[9] M. Corporation and S. Weston, *doParallel: Foreach Parallel Adaptor for the 'parallel' Package*, 2018. R package version 1.0.14.

[10] R. F. Gesteland and J. F. Atkins, *The RNA world : the nature of modern RNA suggests a prebiotic RNA world*. Monograph series (Cold Spring Harbor Laboratory of Quantitative Biology) ; 24, Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press, 1993.

[11] D. P. Clark, *Molecular Biology : Understanding the Genetic Revolution*, ch. 3, p. 63. Amsterdam ; Boston: Academic Press/Elsevier, 2009.

[12] P. A. McGettigan, "Transcriptomics in the rna-seq era," *Current Opinion in Chemical Biology*, vol. 17, no. 1, pp. 4 – 11, 2013. Omics.

[13] Z. Wang, M. Gerstein, and M. Snyder, "RNA-seq: a revolutionary tool for transcriptomics," *Nature Reviews Genetics*, vol. 10, pp. 57–63, jan 2009.

[14] S. Zhao, W.-P. Fung-Leung, A. Bittner, K. Ngo, and X. Liu, "Comparison of RNA-seq and microarray in transcriptome profiling of activated t cells," *PLoS ONE*, vol. 9, p. e78644, jan 2014.

[15] A. Kulkarni, A. G. Anderson, D. P. Merullo, and G. Konopka, "Beyond bulk: a review of single cell transcriptomics methodologies and applications," *Current Opinion in Biotechnology*, vol. 58, pp. 129 – 136, 2019. Systems Biology • Nanobiotechnology.

[16] F. Finotello and B. Di Camillo, "Measuring differential gene expression with rna-seq: challenges and strategies for data analysis," *Briefings in Functional Genomics*, vol. 14, no. 2, pp. 130–142, 2015.

[17] G. C. Budd, "A sand hopper (talitrus saltator)," *Marine Life Information Network: Biology and Sensitvity Key Reviews*, 2005.

[18] P. K. Bregazzi and E. Naylor, "The locomotor activity rhythm of talitrus saltator (montagu) (crustacea, amphipoda)," *Journal of Experimental Biology*, vol. 57, no. 2, pp. 375–391, 1972.

[19] A. Ugolini, L. S. Hoelters, A. Ciofini, V. Pasquali, and D. C. Wilcockson, "Evidence for discrete solar and lunar orientation mechanisms in the beach amphipod, talitrus saltator montagu (crustacea, amphipoda)," *Scientific Reports*, vol. 6, p. 35575, Oct. 2016.

[20] M. Zimmer, "Effects of temperature and precipitation on a flood plain isopod community: a field study," *European Journal of Soil Biology*, vol. 40, no. 3, pp. 139 – 146, 2004.

[21] *Transcription factors : a practical approach*. Practical approach series ; 201, Oxford ; New York: Oxford University Press, 2nd ed. ed., 1999.

[22] L. A. Pennacchio, W. Bickmore, A. Dean, M. A. Nobrega, and G. Bejerano, "Enhancers: five essential questions," *Nature Reviews Genetics*, vol. 14, pp. 288 EP –, Mar 2013. Perspective.

[23] S. Karni, "Analysis of biological networks : Transcriptional networks-promoter sequence analysis," 2007.

[24] A. Sehgal, ed., *Circadian rhythms and biological clocks. Part A*. Methods in enzymology ; volume 551, 2015.

[25] P. Cohen, "The origins of protein phosphorylation," *Nature Cell Biology*, vol. 4, no. 5, pp. E127–E130, 2002.

[26] A. C. Charles and C. Guilleminault, "250 Years Ago: Tribute to a New Discipline (1729-1979)," *Sleep*, vol. 2, pp. 155–160, 09 1979.

[27] P. B. Kidd, M. W. Young, and E. D. Siggia, "Temperature compensation and temperature sensation in the circadian clock," *Proceedings of the National Academy of Sciences*, vol. 112, no. 46, pp. E6284–E6292, 2015.

[28] M. H. Vitaterna, J. S. Takahashi, and F. W. Turek, "Overview of circadian rhythms.," *Alcohol Research & Health*, vol. 25, no. 2, pp. 85–93, 2001.

[29] R. J. Konopka and S. Benzer, "Clock mutants of drosophila melanogaster," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 68, pp. 2112–2116, Sep 1971. 5002428[pmid].

[30] W. A. Zehring, D. A. Wheeler, P. Reddy, R. J. Konopka, C. P. Kyriacou, M. Rosbash, and J. C. Hall, "P-element transformation with period locus dna restores rhythmicity to mutant, arrhythmic drosophila melanogaster," *Cell*, vol. 39, no. 2, Part 1, pp. 369 – 376, 1984.

[31] T. A. Bargiello, F. R. Jackson, and M. W. Young, "Restoration of circadian behavioural rhythms by gene transfer in drosophila," *Nature*, vol. 312, no. 5996, pp. 752–754, 1984.

[32] "The nobel prize in physiology or medicine 2017," Oct 2017. `https://www.nobelprize.org/prizes/medicine/2017/press-release/` Accessed on 01/09/2019 @ 13:19.

[33] P. E. Hardin, J. C. Hall, and M. Rosbash, "Feedback of the drosophila period gene product on circadian cycling of its messenger rna levels," *Nature*, vol. 343, no. 6258, pp. 536–540, 1990.

[34] A. Sehgal, J. Price, B. Man, and M. Young, "Loss of circadian behavioral rhythms and per rna oscillations in the drosophila mutant timeless," *Science*, vol. 263, no. 5153, pp. 1603–1606, 1994.

[35] J. L. Price, J. Blau, A. Rothenfluh, M. Abodeely, B. Kloss, and M. W. Young, "double-time is a novel drosophila clock gene that regulates period protein accumulation," *Cell*, vol. 94, no. 1, pp. 83 – 95, 1998.

[36] J. J. Tyson, C. I. Hong, C. D. Thron, and B. Novak, "A simple model of circadian rhythms based on dimerization and proteolysis of per and tim," *Biophysical Journal*, vol. 77, no. 5, pp. 2411 – 2417, 1999.

[37] P. Emery, W. So, M. Kaneko, J. C. Hall, and M. Rosbash, "Cry, a drosophila clock and light-regulated cryptochrome, is a major contributor to circadian rhythm resetting and photosensitivity," *Cell*, vol. 95, no. 5, pp. 669 – 679, 1998.

[38] T. Ishikawa, A. Matsumoto, T. Kato Jr, S. Togashi, H. Ryo, M. Ikenaga, T. Todo, R. Ueda, and T. Tanimura, "Dcry is a drosophila photoreceptor protein implicated in light entrainment of circadian rhythm," *Genes to Cells*, vol. 4, no. 1, pp. 57–65, 1999.

[39] A. Reinberg and F. Halberg, "Circadian chronopharmacology," *Annual Review of Pharmacology*, vol. 11, no. 1, pp. 455–492, 1971. PMID: 5004942.

[40] G. Cornelissen, "Cosinor-based rhythmometry," *Theoretical biology & medical modelling*, vol. 11, pp. 16–16, Apr. 2014.

[41] M. Sachs, *cosinor: Tools for estimating and predicting the cosinor model*, 2014. R package version 1.1.

[42] A. Mutak, *cosinor2: Extended Tools for Cosinor Analysis of Rhythms*, 2018. R package version 0.2.1.

[43] I. C. Graham Upton, *A Dictionary of Statistics 3e*, p. 126. Oxford University Press, 2014.

[44] I. C. Graham Upton, *A Dictionary of Statistics 3e*, p. 379. Oxford University Press, 2014.

[45] G. N. Lance and W. T. Williams, "A generalized sorting strategy for computer classifications," *Nature*, vol. 212, pp. 218–218, Oct. 1966.

[46] L. L. McQuitty, "Hierarchical linkage analysis for the isolation of types," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 55–67, 1960.

[47] P. Arabie, L. J. Hubert, and G. De Soete, *Clustering and classification*. Singapore ; London: World Scientific, 1996.

[48] A. de Vries and B. D. Ripley, *ggdendro: Create Dendrograms and Tree Diagrams Using 'ggplot2'*, 2016. R package version 0.1-20.

[49] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data : An Introduction to Cluster Analysis*, ch. 6, pp. 253–259. John Wiley & Sons, Inc., 1990.

[50] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data : An Introduction to Cluster Analysis*, ch. 2, pp. 68–72. John Wiley & Sons, Inc., 1990.

[51] W. S. Manjoro, M. Dhakar, and B. K. Chaurasia, "Operational analysis of k-medoids and k-means algorithms on noisy data," in *2016 International Conference on Communication and Signal Processing (ICCSP)*, pp. 1500–1505, IEEE, 2016.

[52] G. Brock, V. Pihur, S. Datta, and S. Datta, "clValid: An R package for cluster validation," *Journal of Statistical Software*, vol. 25, no. 4, pp. 1–22, 2008.

[53] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome research*, vol. 13, pp. 2498–2504, Nov. 2003.

[54] P. Wu, L. Bao, R. Zhang, Y. Li, L. Liu, Y. Wu, J. Zhang, Z. He, and W. Chu, "Impact of short-term fasting on the rhythmic expression of the core circadian clock and clock-controlled genes in skeletal m uscle of crucian carp (carassius auratus)," *Genes*, vol. 9, p. 526, Oct. 2018.

[55] M. E. Hughes, J. B. Hogenesch, and K. Kornacker, "Jtk_cycle: an efficient nonparametric algorithm for detecting rhythmic components in genome-scale data sets," *Journal of biological rhythms*, vol. 25, pp. 372–380, Oct. 2010.

[56] P. F. Thaben and P. O. Westermark, "Detecting rhythms in time series with rain," *Journal of biological rhythms*, vol. 29, pp. 391–400, Dec 2014. 25326247[pmid].

[57] L. J. Heyer, "Exploring expression data: Identification and analysis of coexpressed genes," *Genome Research*, vol. 9, pp. 1106–1115, nov 1999.

[58] M. Jeanmougin, A. de Reynies, L. Marisa, C. Paccard, G. Nuel, and M. Guedj, "Should we abandon the t-test in the analysis of gene expression microarray data: A comparison of variance modeling strategies," *PLoS ONE*, vol. 5, p. e12336, sep 2010.

[59] L. de Torrenté, S. Zimmerman, M. Suzuki, M. Christopeit, J. M. Greally, and J. C. Mar, "The shape of gene expression distributions matter: how incorporating distribution shape improves the interpretation of cancer transcriptomic data," mar 2019.

[60] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 96, pp. 6745–6750, June 1999.

[61] F. D. Gibbons and F. P. Roth, "Judging the quality of gene expression-based clustering methods using gene annotation," *Genome research*, vol. 12, pp. 1574–1581, Oct. 2002.

[62] Feng Luo, Kun Tang, and L. Khan, "Hierarchical clustering of gene expression data," in *Third IEEE Symposium on Bioinformatics and Bioengineering, 2003. Proceedings.*, pp. 328–335, March 2003.

[63] P. A. Jaskowiak, R. J. G. B. Campello, and I. G. Costa, "Evaluating correlation coefficients for clustering gene expression profiles of cancer," in *Advances in Bioinformatics and Computational Biology* (M. C. de Souto and M. G. Kann, eds.), (Berlin, Heidelberg), pp. 120–131, Springer Berlin Heidelberg, 2012.

[64] P. D'haeseleer, "How does gene expression clustering work?," *Nature Biotechnology*, vol. 23, pp. 1499–1501, Dec. 2005.

[65] P. Langfelder and S. Horvath, "Wgcna: an r package for weighted correlation network analysis," *BMC Bioinformatics*, vol. 9, p. 559, Dec. 2008.

[66] J. Liu, L. Jing, and X. Tu, "Weighted gene co-expression network analysis identifies specific modules and hub genes related to coronary artery disease," *BMC Cardiovascular Disorders*, vol. 16, p. 54, Mar. 2016.

[67] L. Miao, R.-X. Yin, S.-L. Pan, S. Yang, D.-Z. Yang, and W.-X. Lin, "Weighted gene co-expression network analysis identifies specific modules and hub genes related to hyperlipidemia," 2018.

[68] L. Korslund, "Activity of root voles (microtus oeconomus) under snow: social encounters synchronize individual activity rhythms," *Behavioral Ecology and Sociobiology*, vol. 61, p. 255, Sep 2006.

[69] F. Dündar, L. Skrabanek, P. Zumbo, and P. O. Westermark, "Introduction to differential gene expression analysis using rna-seq.," *Journal of biological rhythms*, pp. 1–67, 2015.

[70] T. Scharl and F. Leisch, "Jackknife distances for clustering time – course gene expression data," 2006.

[71] E. Schubert and P. J. Rousseeuw, "Faster k-medoids clustering: Improving the pam, clara, and CLARANS algorithms," *CoRR*, vol. abs/1810.05691, 2018.

[72] "What is systems biology." `https://isbscience.org/about/what-is-systems-biology/` Accessed on 15/09/2019 @ 11:45.

[73] U. Sauer, M. Heinemann, and N. Zamboni, "Getting closer to the whole picture," *Science*, vol. 316, no. 5824, pp. 550–551, 2007.

[74] B. N. Kholodenko, F. J. Bruggeman, and H. M. Sauro, *Mechanistic and modular approaches to modeling and inference of cellular regulatory networks*, pp. 143–159. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[75] "Predicct website." `https://www.predicct.co.uk` Accessed on 15/09/2019 @ 11:51.

[76] A. Pomerantz, N. Peñafiel, A. Arteaga, L. Bustamante, F. Pichardo, L. A. Coloma, C. L. Barrio-Amorós, D. Salazar-Valenzuela, and S. Prost, "Real-time dna barcoding in a rainforest using nanopore sequencing: opportunities for rapid biodiversity assessments and local capacity building," *GigaScience*, vol. 7, p. giy033, Apr. 2018.

[77] S. L. Brilleman, M. J. Crowther, M. Moreno-Betancur, J. B. Novik, J. Dunyak, N. Al-Huniti, R. Fox, J. Hammerbacher, and R. Wolfe, "Joint longitudinal and time-to-event models for multilevel hierarchical data," *Statistical Methods in Medical Research*, vol. 28, no. 12, pp. 3502–3515, 2019. PMID: 30378472.

[78] D. P. Walsh, V. J. Dreitz, and D. M. Heisey, "Integrated survival analysis using an event-time approach in a bayesian framework," *Ecology and Evolution*, vol. 5, no. 3, pp. 769–780, 2015.

[79] A. Bellot and M. van der Schaar, "A hierarchical bayesian model for personalized survival predictions," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, pp. 72–80, Jan 2019.