1. A Bioinformatics Approach to Synthetic Lethal Interactions in Breast Cancer with Gene Expression Data

31 March 2017

# Methods, Techniques, and Resources

In this chapter, I will outline the various existing resources and methods utilised throughout this project. This includes public data repositories, stable and development releases of software packages (mostly for the R programming environment), and implementation of bioinformatics methods or statistical concepts with Shell or R scripts developed for this purpose. Methods and packages developed specifically for this project will be covered in more detail along with preliminary data to demonstrate and support their use in chapter 3.

## Bioinformatics Resources to Enable Genomics Research

### Public Data and Software Packages

Various bioinformatics resources, such as databases and methods, have become integral parts of genetics and genomics research. Reference genomes, genotyped variants, gene expression, and epigenetics profiles are among the most commonly used resources. Gene expression data in particular is widely available from many microarray and RNA-Seq studies, from repositories such as Gene Expression Omnibus (GEO) , caArray , and ArrayExpress . Such profiles serve as an excellent resource to examine the changes of gene expression occurring in cancers and the variation between samples. These microarray initiatives have set a precedent for data sharing, data mining, and the wider benefits of publicly available data for enabling the scientific community to further utilise the data rather than a single research group or consortium . The practice of integrating findings from publicly available genomics data with the research questions and experimental results of individual research groups has carried over into RNA-Seq datasets including the large-scale cancer genomics projects . This thesis is one such example of an investigation enabled by this wider movement and tools developed in various disciplines to generate, disseminate, and process genomic-scale data.

Along with databases, it is also becoming common practice for bioinformatics researchers to release their code as open-source or provide a software package to enable replication of the findings or further applications of the methods . This is part of a wider movement in software and data analysis with many tools to facilitate such work being released for use in Linux or the R programming environment . In addition to the R packages hosted on CRAN , the Bioconductor repositories also contain many packages specifically for applications in bioinformatics, and the GitHub site hosts many packages in various stages of development and early release. Packages from these various sources have been used throughout this project and cited where-ever possible. Several R packages have been developed during this thesis project and either publicly released on GitHub or prepared to accompany a publication.

#### Cancer Genome Atlas Data

Molecular profile data from normal and tumour samples was downloaded from publicly available sources, using the TCGA and the International Cancer Genome Consortium (ICGC) web portals . These include gene expression (RNA-Seq), somatic mutations, and anonymous clinical data. These versions downloaded were on Aug 6th 2015 (Release 19) and May 2nd 2016 (Release 20) for breast and stomach cancer respectively via the ICGC data portal (<https://dcc.icgc.org/>).

Performing a genomic alignment in remains a challenge in bioinformatics and methods to do so may yet be improved . However, the statistical and biological aspects of Bioinformatics are the focus of this thesis, comparing alignment methods is outside the scope of these investigations. The TCGA project used widely adopted tools: “Bowtie” for alignment , “mapslice” to detect splice sites , and the Reads Per Kilobase per Million mapped reads (RPKM) approach to qualify reads per transcript as a measure of gene expression . These are widely acceptable tools for processing RNA-Seq data and this is used to produce the raw counts of mapped reads (tier 1) and normalised expression data (tier 3) publicly available from TCGA.

Raw count and RSEM normalised TCGA expression data from Illumina RNA-Seq protocols were available from 1,177 samples (113 normal, 1,057 primary tumour, and 7 metastases) for 20,501 genes. TCGA somatic mutation data for 981 samples (976 primary tumours and 5 metastases) across 25,836 genes were available including 969 samples (964 primary tumours and 5 metastases) with corresponding RNA-Seq expression data and 19,166 genes mapped from Ensembl identifiers to gene symbols, of which 16,156 had corresponding gene expression information. Unless otherwise stated, the raw counts were used for further processing rather than the RSEM normalised data (provided by TCGA tier 3).

#### Reactome and Annotation Data

Unless otherwise specified, pathway analysis was performed for Human pathway annotation from the Reactome database (version 52) with pathway gene sets derived from the reactome.db R package. Entrez identifiers were mapped to gene symbols or aliases to match to TCGA expression and mutation data using the org.Hs.eg.db R package. Further pathway analysis used breast cancer gene signatures from Gatza and colleagues (Gatza *et al*., 2011; Gatza *et al*., 2014). These gene symbols were matched to the relevant dataset and used to construct a matrix of category membership using the safe R package .

## Data Handling

### Normalisation (voom)

Apart from the PAM50 subtyping procedure which required RSEM normalised data (J.S. Parker personal communication), the analysis of the RNA-Seq data presented here was based on raw read count data. Raw read counts were log-scaled, samples were checked for consistency with some removed (as described in section [methods:sampleqc]) based on a correlation matrix (Euclidean distance), and the final dataset was TMM normalised and then processed using the voom function in the limma R package .

[!ht]

[fig:density]

[!ht]

[fig:boxplot]

### Sample triage

The TCGA RNA-Seq data were assessed for batch effects using a correlation matrix of the log-transformed raw counts for which a heatmap (Euclidean distance, complete linkage) is shown in Figure [fig:corrmap]. While no major batch effects were detectable between the samples, 9 samples were excluded due to poor correlation with the remaining samples, as detailed in Table [tab:qc]. These samples showed unusual density plots compared to the rest of the dataset, and exhibited low mean read count in Figures [fig:density] and [fig:boxplot]. A heatmap showing key clinical properties of these excluded samples and their correlation with the remainder of the samples is shown in Figure [fig:corrmappart], and a full correlation heatmap (Figure [fig:corrmap]) shows these samples as relatively poorly correlated outliers in the bottom rows and left columns. After removal of these samples, the TCGA dataset used for analysis consisted of the remaining 1168 samples (from 1040 patients): 1049 tumour samples, 112 normal tissue for matched samples, and 7 metastases.

[!ht] [tab:qc]

### Pathway Metagenes and the Singular Value Decomposition

A “metagene” offers a consistent signal of pathway (expression) activation or inactivation by dimension reduction of a matrix, avoiding negatively correlated genes averaging out the signal of a mean-based centroid . Constructing these pathway metagenes used gene sets for Reactome and Gatza signatures (Gatza *et al*., 2011; Gatza *et al*., 2014) as specified above (see Section [methods:geneset]). The singular-value decomposition was performed (*X* = *UTDV* where *X* is the data matrix of the gene set with genes  ×  samples) and the leading eigenvector (first column of *V*) corresponding to the largest singular value was used as a metagene for the pathway gene set. To ensure consistent directionality of metagene signals, the median of the gene set in each sample was calculated and correlated against the metagene with the (arbitrary) metagene sign adjusted as needed to conform with the majority of the gene set (i.e., positive correlation between metagene and the median-based centroid). To ensure that genes and pathways were weighted equally, this was performed on a z-transformed dataset of gene expression and samples were scaled (by fractional ranking) for each metagene so that each metagene was on a comparable [0, 1] scale.

#### Candidate Triage and Integration with Screen Data

Candidate triage in combination with the experimental data was intended to integrate findings of the SLIPT analysis with an ongoing experiment project . The first procedure to compare the SLIPT gene candidates for *CDH1* with an siRNA experimental screen was a direct comparison of the overlapping candidates, presented in a Venn diagram and tested with the *χ*2 test. Since these candidates were not very comparable at the gene level (even when excluding genes not contained in both datasets), further gene set over-representation analysis was performed for pathways specific to each detection approach and the intersection of the two. The pathway composition of the intersection was further verified by a permutation resampling analysis (as described in section [methods:permutation]), the same number of genes detected by SLIPT were sampled randomly from the universe of genes tested by both approaches. These samplings were performed over 1 million iterations and the pathway over-representation was compared for each of the 1,652 reactome pathways. These over-representation scores (*χ*2) were compared the observed over-representation in the intersection of the SLIPT candidates, with the proportion of resamplings with higher *χ*2 values used for empirical p-values of pathway composition. Pathways for which no resamplings were observed as high as the observed were reported as *p* < 10− 6. These empirical p-values were adjusted for multiple comparisons (FDR). Intersection size was not assumed to be constant across resamplings so similarly with the proportion of resamplings with higher or lower intersection size were used to evaluate significance of enrichment or depletion respectively (of siRNA candidate among SLIPT candidate genes).

## Techniques

Various statisical, computational, and bioinformatics techniques were performed throughout this thesis. This section describes these techniques and gives the parameters used throughout this thesis unless otherwise specified. Where relevant the R package implementation which provided the technique will be acknowledged.

### Statistical Procedures and Tests

As described in sections [methods:heatmap] and [methods:metagene], the z-transform has been used to generate z-scores in various analyses in this thesis. Here each row of dataset (*x*) is transformed into a scores (*z*) using the mean (*μ*) and standard deviation (*σ*) of the data such that: $$z = \frac{x - \mu(x)}{\sigma(x)}$$. This generates data where each row (gene) has a mean of 0 and standard deviation of 1. Where plotted as aa heatmap, any data more than 3 standard deviations above or below the mean is plotted as 3 or  − 3 respectively.

Empirical Bayes differential expression analysis was performed using the limma R package . Where specified, the Fisher’s exact test, *χ*2 test, and correlation were used to measure associations between variables (as implemented in the stats R package ). Unless otherwise specified, Pearson’s correlation was used for correlation analyses (*r*) and coefficient of determination (*R*2). Where these comparisons are discussed in more detail, Fisher’s exact test and *χ*2 tests are supported by a table or Venn diagram, rendered with the limma R package . In some analyses, correlation is furter supported by a scatter plot and a line of best fit devied by least squares linear regression.

The t.test function has also been used to implement the t-test to compare pairs of data. Where relevant, an analysis of variance (ANOVA) has been performed to report significance of multivariate predictors of outcomes, or least squares linear regression performed for the adjusted coefficient of determination (*R*2) and F-statistic p-value to evaluate the fit of the predictor variables. For some analyses these are supported by boxplot or violinplot visualisation (rendered in R).

Multiple comparisons are adjusted by the Benjamini-Hochberg procedure to control the false discovery rate (FDR) unless otherwise specified . This procedure adjusts p-values to achieve an average of the proportion of false-positives among significant tests below a threshold, *α*. The more stringent Holm-Bonferroni (Holm) procedure was also applied in some cases to adjust for multiple comparisons and control the family-wise error rate which adjusts p-values so that the probability that any one of the tests is a false-positive (type-1 error) below a threshold, *α*.

### Gene Set Over-representation Analysis

Gene set enrichment over-representation tests whether there is an enrichment of a gene set (such as a biological pathway) among a group of input genes. Such input genes may be predicted synthetic lethal candidates or a subset defined by clustering (in section [methods:clustering]) or comparison with experimental candidates (see section [methods:vennanalysis]). Initially, these were performed using the GeneSetDB web tool hosted by the University of Auckland on the Reactome pathways . Since the GeneSetDB tool used an older version of Reactome (version 40), it was difficult to directly compare with the results of other analysis (see sections [methods:vennanalysis] and [methods:permutation]) performed on version 52 (as described in section [methods:geneset]). Thus an implementation of the hypergeometric test in R was used to test for over-representation against Reactome (version 52) pathways. Pathways containing less than 10 genes or more than 500 were excluded before adjusting for multiple comparisons.

### Clustering

Clustering analysis when performed uses unsupervised hierarchical clustering with complete linkage (distance calculated from the furthest possible pairing). For correlation matrices or multivariate normal parameters (e.g., Σ ), the distance metric used was Euclidean distance. For empirical or simulated gene and pathway expression data correlation distance was used, calculated by *distance* = 1 − *cor*(*t*(*x*)) where *cor* is Pearson’s correlation and *t*(*x*) is the transpose of the expression matrix.

### Heatmap

Standardised z-scores of the data were used to plot heatmaps on an appropriate scale. Raw (log-scale) read counts or voom normalised counts per gene (as specified) were plotted as normalised z-scores on a [ − 3,  + 3] blue-red scale. Similarly, correlations were plotted on a [ − 1, 1] blue-red scale. These heatmaps were performed using the linkage and distance specified for the clustering performed in Section [methods:clustering]. The gplots R package was used to generate many of the heatmaps throughout this thesis, along with a customised heatmap function (released as heatmap.2x). Where clearly specified, data have been split into subsets with clustering performed separately on each subset with these plotted alongside each other.

### Modeling and Simulations

Statistical modeling and simulations have been used to test various synthetic lethal detection procedures on simulated data. This involves constructing a statistical model of how synthetic lethality would appear in (continuous normally distributed) gene expression data. Where presented (in section [methods:SLModel]), the assumptions of the model are stated clearly. The model allows sampling from a multivariate normal distribution (using the mvtnorm R package) to generate simulated data with known underlying synthetic lethal partners (detailed in section [methods:simulating SL]). We can test whether statistical procedures, including those developed in this thesis (presented in section [methods:SLIPT]), are capable of detecting them upon this simulated data. This multivariate normal simulation procedure also enables the inclusion of correlation structure which is either given as correlated blocks of genes or derived from pathway structures (as detailed in section [methods:graphsim]).

If this multivariate normal distribution was sampled once and the procedure to add known synthetic lethal partners was performed, it generates a simulated dataset. Performing this simulation procedure and testing with a synthetic lethal detection procedure iteratively, these simulations can be used to assess the statistical performance of the detection procedure. The number of iterations (Reps) will be given for each simulation result. Typically, these are performed 1000 or 10,000 times depending on computational feasibility of doing so on larger datasets.

Several measures of statistical performance were used to assess the simulations. The following measures used the final classification of the detection procedure, statistical significance for *χ*2, significance and directional criteria met for SLIPT (see section [methods:SLIPT]), and an arbitrary threshold of  > 0. 2 and  <  − 0. 2 for correlation and negative correlation respectively. Sensitivity (or “true positive rate”) was measured as the proportion of known synthetic lethal partners predicted to be synthetic lethal. Specificity (or “true negative rate”) was measured as the proportion of known non-synthetic lethal partners predicted not to be synthetic lethal. The “false discovery rate” (also important in adjusting for multiple comparisons) was measured here as the proportion of known non-synthetic lethal partners out of all putative partners predicted by the detection procedure. Statistical “accuracy” is the proportion of true predictions for a detection procedure, in this case the total correctly predicted known synthetic lethal partners and correctly negative known non-synthetic lethal partners.

#### Receiver Operating Characteristic (Statistical Performance)

A more general procedure to measure the statistical performance of a simulation is the Receiver Operating Characteristic (ROC) curve which does not assume a threshold for classification of synthetic lethality but demonstrates the trade-off of sensitivity and specificity . These curves (implemented with the ROCR R package ) plot the True Positive Rate (sensitivity) against the False Positive Rate (1 −  specificity) as the prediction threshold is varied. An ideal detection method will have a true positive rate of 1 and a false positive rate of 0, hence the Area Under the ROC curve (AUC or AUROC) is a measure of statistical performance for a detection procedure accounting for this trade-off. AUROC values are typically frange from 0.5 the value expected by random chance to 1 for an optimal detection method, however it possible for an AUROC below 0.5 for a poor detection method that performs worse than random chance. An cancer biology, an AUROC of approximately 0. 8 is a predictive biomarker suitable for publication but predictors with lower AUROC values may still be informative depending on the context. In this thesis, the AUROC values varies widely across simulation parameters and a primarily used for comparisons across these parameters, although they can also be used to refine thresholds for optimal classification.

### Resampling Analysis

Resampling analyses (e.g., “permutation” analysis) are used to statistically test the significance of an observation without assuming the underlying distribution of expected test statistics . Instead these are derived from randomly shuffling test statistics or randomly sampling predicted candidates. For the purposes of this thesis, this involved randomly sampling genes from those tested to be analysed as putative synthetic lethal candidates. This was performed both for testing the significance of pathway composition in the intersection with experimental gene candidates (section [methods:vennanalysis]) and for assessing the significance of pathway structure among synthetic lethal candidates (section [methods:networkpermutation]).

These were analysed to compare the observed synthetic lethal genes against values derived from randomly sampling the same number of genes as observed by synthetic lethal from among the genes tested. Sampling iteratively across many resampling procedures, these resampling-based values form a null distribution that would be observed if the null hypothesis were true. Thus the proportion of resampling-based values across these iterations that are greater than or equal to that observed, forms an empirically derived p-value to test significance.

Resampling was performed for comparison (in section [methods:vennanalysis]) with fixed experimental screen candidates both resampling the number of genes overlapping with the screen candidates and test statistics for pathway enrichment. Resampling analysis was also applied to shortest paths and network metrics (in section [methods:networkpermutation]) to test significance of directional relationships between synthetic lethal candidate genes within pathway structures.

The number of iterations determines the accuracy of these p-values. For pathway composition (in section [methods:vennanalysis]), a million iterations were performed using high performance computing (as detailed in section [methods:HPC]) to provide sufficient accuracy after adjusting for multiple comparisons across pathways. For the purposes of network analysis (in section [methods:networkpermutation]), a thousand iterations were sufficient to reject the null hypothesis for the majority of pathways tested before adjusting for multiple comparisons, and thus further iterations were not performed.

## Pathway Structure Methods

### Network and Graph Analysis

Networks are important in considering the structure of relationships in molecular biology, including gene regulation, kinase cellular signaling, and metabolic pathways . Network theory is an interdisciplinary field which combines the approaches of computer science with the metrics and fundamental principles of graph theory, an area of pure mathematics dealing with relationships between sets of discrete elements. The vast amounts of molecular and cellular data from high-throughput technologies have enabled the application of network-based and genome-wide bioinformatics analysis to examine the complexity of a cell at the molecular level and understand aberrations in cancer. This thesis uses various metrics and analysis procedures developed in Graph and Network theory to analyse graph structure of biological pathways. Where feasible, these have been implemented using the igraph R package with such procedures described below . Custom R functions to perform more complex analysis and visualisation of igraph data objects will be described later.

Graph theory is a branch of pure mathematics which deals with the properties of sets of discrete objects (referred to as a ‘node’ or `vertex`) with some pairs are joined (by a ‘link’ or an `edge`). While a seemingly reductionist abstraction to mathematically study relationships, graph theory serves has applications in a wide range of studies including life sciences. Network theory is the sub-discipline of graph theory which deals with networks which has become popular due to the vast potential for applications of networks .

Applications vary depending on the situation modeled, particularly in how the edges between vertices are defined, whether they are directed or weighted, and whether multiple redundant edges between a pair of vertices (referred to as `parallel edges`) or edges connecting a vertex to itself (referred to as `loops`) are permitted in the model. Networks are defined such that the edges represent a relationship between the vertices and may be directed, weighted, or contain parallel edges or loops depending on the application. Unless otherwise stated, graph structures and networks in thesis will be unweighted and have no parallel edges or loops. Where a directional relationship is known or modeled, it will be represented with a directed edge in a digraph.

**Table 1.** Types of biological network based on molecular data

m4.0490003cm|m2.55cm|m4.301cm|m4.201cm & & &   
 **& & &**  
**Protein (Physical) & Protein & Protein-protein binding & Two-Hybrid**  
 **& & &**  
**Metabolic & Metabolite or cofactor & Involved in same reaction (links by reactions/enzymes) & Curate known pathways**  
 **& & &**  
**Regulation & Gene & Regulate each other by encoded proteins & Array, RNA-Seq, ChIP**  
 **& & &**  
**Functional & Gene & Shared gene function & Curate known pathways**  
 **& & &**

### Sourcing Graph Structure Data

Pathway Commons interaction data was sourced using the paxtools-4.3.0 Java application on October 6th 2015 . This utility was used to source ‘sif’ format interaction data into R , from which the human Reactome (version 52) dataset of interactions was imported , matching those used for pathway enrichment analysis. These interactions were used to construct an adjacency matrix for the Reactome network and subnetworks corresponding to each relevant biological pathway.

### Constructing Pathway Subgraphs

Subgraphs for each relevant pathway were constructed by matching the nodes in the complete Reactome network to the pathway gene sets (as derived in section [methods:geneset]). A subgraph with adjacent nodes was constructed by adding nodes which have an edge with a gene in the pathway gene set. The pathways these adjacent nodes belong to were added to form a “meta-pathway” to account for the possibility for nodes within the pathway being linked by the surrounding graph structure.

### Network Analysis Metrics

The existing network analysis measures which were applied in this thesis used an implementation in the igraph R package where it was available . Otherwise, custom features were developed for analysis of iGraph objects in R and released as igraph.extensions (as described in section [methods:igraphextensions]).

Vertex degree is the number of edges a node has and is a fundamental measure of the importance and connectivity of a network . More connected nodes, such as network hubs, will have a higher vertex degree relative to other nodes. For the purposes of this thesis, vertex degree ignored edge direction with loops (edges with itself) and double edges to the same node excluded.

A fundamental concept in network analysis is a “shortest path”, that is the shortest route via edges between any two particular nodes in a network. These are computed by Dijstra’s algorithm in the igraph R package . Where applicable paths will only use directed edges in a particular direction. Shortests paths are a useful measure of how close nodes are in a network. This is used to compute information centrality, and for further analysis of pathway structure (as described in section [methods:pathwaystr]).

Network centrality is an alternative measure of the importance or influence of a node to the graph structure . Various strategies are used to derive centrality, typically based on how connected the node is or the impact of node removal on the connectivity of the network. One of the most notable is the “PageRank” algorithm, a refinement of eigenvector centrality based on the eigenvectors of the adjacency matrix . This is implemented in the igraph R package .

Another network centrality measure that has been previously applied to biological protein interaction networks is the “information centrality”. The information centrality of a node is the relative impact on efficiency (transmission of information via shortest paths) of the network when the node is removed. That is the centrality (*C*) for node *n* in graph *G* is defined as: $$C\_n = \frac{E(G)-E(G')}{E(G)}$$ where *G*ʹ is the subgraph with the node removed and *E* is the efficiency derived from shortest paths (*dij* between nodes *i* and *j*) as: $$E(G) = \frac{2}{N(N-1} \sum\_{i<j \in G}^{} \frac{1}{d\_{ij}}$$ The efficiency of the network is implemented in the igraph R package and the iterative network centrality computation of each node has been released as an R package (info.centrality) and included in the igraph.extensions package.

**Table 5.** Data sources for Research

m2.299cm|m3.8009999cm|m3.552cm|m6.5490003cm & & & **Data Types Supported**  
 **& Cancer Genome Atlas & Cancer & Sequence, Mutation, CNV, SNP, expression, DNA Methylation, Clinical, RNA-Seq**  
**ICGC & International Cancer Genome Consortium & Cancer & Sequence, Mutation, CNV, SNP, expression, DNA Methylation, Clinical, RNA-Seq**  
 **& Encyclopaedia of DNA Elements &**

Cancer Cell Line &

RNA-Seq, ChIP-Seq, RIP-Seq  
CCLE & Cell Line Encyclopaedia & Cancer Cell Line & DNA CNV, expression, mutation, drug sensitivity  
 & Gene Expression Omnibus & Various & Gene Expression, RNA-Seq  
GENT & Gene Expression Atlas & Human Normal Tissue and Cancer & Gene Expression  
 & Biomolecular interaction network database &

Human &

Two-hybrid data, ChIP, CLIP, RIP  
STRING & Search Tool for retrieving interacting genes/proteins & Various & Protein-protein interactions: curated and predicted from MINT, PPRD, BIND, DIP, BioGRID, KEGG, Reactome, IntAct, EcoCyc, NCI, and GO  
 & Biological General Repository for Interaction Datasets &

,

, etc & Protein and Genetic Interaction  
Human Interactome & Human Interactome & Human & Protein Interactions  
 & Data Repository of Yeast Genetic Interactions & Yeast & Genetic Interactions  
SGD & Saccharomyces Genome Database & Yeast & Sequence, Expression, Protein and Genetic Interactions  
 & Gene Ontology & Various & Gene Function  
KEGG & Kyoto Encyclopedia of Genes and Genomes & Various & Gene Function and Pathway  
 &   & Various & Gene Function and Pathway  
DrugBank &   & Human & Chemical and drug target sequence/structure  
 & Therapeutic Target & Human & Protein and Nucleic Acid drug targets  
TDR & Targets database & Human & Chemical Genomics (Tropical Disease)  
 &   &   & Protein-drug binding affinity

## Implementation

### Computational Resources and Linux Utilities

Several computers were used to process and store data during this thesis (as summarised in Table [tab:computers]), running different versions of Linux operating systems, including a personal laptop computer, laboratory desktop machine, departmental server, and the New Zealand eScience Infrastructure Intel Pan high-performance computing cluster (a supercomputer based at the University of Auckland). Current workflows on local machines use Elementary OS (based on the Ubuntu versions given in Table [tab:computers]) and interacting with these via ZSH shell. However, Ubuntu OS and the Bourne Again SHell (bash) were used at the inception of this project and bash is continues to be used for running scripts. Various Linux applications and command-line utilities were used on these machines (as summarised in Table [tab:computerslinux]). As such, the workflows developed in this project should be backwards-compatible with Ubuntu Linux (and other derivatives). The majority of novel methodology and implementations were performed in R which is a cross-platform language, packages developed in R will be available for users of Linux, Mac, and Windows machines.

[!ht] [tab:computers]

[!ht] [tab:computerslinux]

[!ht] [tab:computersr]

### R Language and Packages

[!ht] [tab:computersrpackagesdev]

The R programming language has been used for the majority of this thesis. Current R installations across the machines used are given in Table [tab:computersr]. Local machines currently run the latest version of the R (at the time of writing) and remote machines run the versions and modules as managed by the system administrator. Various scripts and packages in this thesis were developed or run in previous versions of RStudio and R but these run without error in the current version of R (and the older versions on remote machines). The R packages developed during this thesis are given in Table [tab:computersrpackagesdev] with the relevant sections describing their implementation and use where appropriate. Various R packages were used throughout this thesis (as detailed in Table [tab:computersrpackages] with versions specified) these have not been updated if they would change the functionality of the scripts or functions in packages, in particular imported data from annotation packages (used to define gene sets) have been saved as local files to continue using stable versions of these pathway data (across machines). This is a summary of the key packages which (in addition to their dependencies) have been used throughout this project. Where a package implementation has been central to the methods applied, they are described in the relevant sections as well. A full table of packages used in this thesis can be found in the Appendix (Table [tab:computersrpackagesfull]).

|llll| [tab:computersrpackages]  
 & & &  
 colorspace & 1.3-2 & 3.3.1 & CRAN  
curl & 2.3 & 3.3.1 & CRAN  
 data.table & 1.9.6 & 3.3.1 & CRAN  
dendextend & 1.4.0 & 3.3.2 & CRAN  
 DBI & 0.5-1 & 3.3.1 & CRAN  
devtools & 1.12.0 & 3.3.1 & CRAN  
 dplyr & 0.5.0 & 3.3.1 & CRAN  
ggplot2 & 2.2.1 & 3.3.1 & CRAN  
 git2r & 0.18.0 & 3.3.1 & CRAN  
gplots & 3.0.1 & 3.3.1 & CRAN  
 gtools & 3.5.0 & 3.3.1 & CRAN  
igraph & 1.0.1 & 3.3.1 & CRAN  
 matrixcalc & 1.0-3 & 3.3.1 & CRAN  
mclust & 5.2.2 & 3.3.1 & CRAN  
 mvtnorm & 1.0-6 & 3.3.1 & CRAN  
org.Hs.eg.db & 3.1.2 & 3.1.2 & Bioconductor  
 openssl & 0.9.6 & 3.3.1 & CRAN  
plyr & 1.8.4 & 3.3.1 & CRAN  
 purrr & 0.2.2 & 3.3.1 & CRAN  
reactome.db & 1.52.1 & 3.2.1 & Bioconductor  
 RColorBrewer & 1.1-2 & 3.3.1 & CRAN  
Rcpp & 0.12.9 & 3.3.1 & CRAN  
 ROCR & 1.0-7 & 3.3.1 & CRAN  
roxygen2 & 6.0.1 & 3.3.2 & CRAN  
 shiny & 1.0.0 & 3.3.1 & CRAN  
snow & 0.4-2 & 3.3.1 & CRAN  
 testthat & 1.0.2 & 3.3.2 & CRAN  
tidyr & 0.6.1 & 3.3.2 & CRAN  
 tidyverse & 1.1.1 & 3.3.2 & GitHub (hadley)  
sm & 2.2-5.4 & 3.3.1 & CRAN  
 Unicode & 9.0.0-1 & 3.3.2 & CRAN  
vioplot & 0.2 & 3.3.1 & CRAN  
 viridis & 0.3.4 & 3.3.2 & CRAN  
xml2 & 1.1.1 & 3.3.2 & CRAN  
 xtable & 1.8-2 & 3.3.1 & CRAN  
zoo & 1.7-14 & 3.3.1 & CRAN  
 graphics & 3.3.2 & 3.3.2 & base  
grDevices & 3.3.2 & 3.3.2 & base  
 cluster & 2.0.5 & 3.3.1 & base  
graphics & 3.3.2 & 3.3.2 & base  
 grDevices & 3.3.2 & 3.3.2 & base  
Matrix & 1.2-8 & 3.3.1 & base  
 stats & 3.3.2 & 3.3.2 & base

### High Performance and Parallel Computing

Another enabling technology for bioinformatics is parallel computing, performing independent operations in separate cores, such “multithreading” is widely used to increase the time to compute results. Bioinformatics is particularly amenable to this since performing multiple iterations of a simulation or testing separate genes is often ``embarrassingly parallel``, being completely independent of the results of each other. As such parallel computing is offered by many high-performance “supercomputers” including national research infrastructure.

The New Zealand eScience Infrastructure (NeSI) is a computating resource providing the Intel Pan cluster hosted by the University of Auckland . The Pan cluster used throughout this thesis project to optimise and perform computations which would have otherwise been infeasible in the timeframe of thesis. Such technological developments and infrastructure initiatives have enabled bioinformatics research including this project. High performance computing on the Pan cluster was used extensively in this project including for resampling analysis (in sections [methods:permutation] and [methods:networkpermutation]), calculating information centrality (in section [methods:networkmetrics]), and in simulations (in sections [methods:simulation], [methods:simulationSLexpression], and [methods:graphsim])

Scripts and data were transferred between the Pan cluster and University of Otago computing resources by rsync and the Globus file transfer service . R scripts were run in parallel with the “simple network of workstations” snow R package . This utilised the “message passing interface” when it was feasible with memory requirements to run in parallel across multiple compute nodes, otherwise SOCKS was used to access multiple cores within an instance of R and pass input data to them. R jobs were submitted to queue for available resources and run on the Pan cluster via the Slurm workload manager . When running R scripts across many parameters or for memory-intensive jobs, slurm array job submission and independent submission of different parameters via shell commands with arguments passed to R. In some cases, this submission was automated across a range of parameters with Bash scripts.

# Methods Developed During Thesis

Here we outline the rationale and development of various methods used throughout this thesis to examine synthetic lethality in gene expression data, graph structures, models and simulations. First this describes the Synthetic Lethal Interaction Prediction Tool (SLIPT), a bioinformatics approach to synthetic lethal candidate triage. This is considered one of the main research outputs of the thesis, which is supported by comparisons to an experimental screen from a related project and performance on simulated data. These supporting data will be covered in further chapters but preliminary data to support the use and design of SLIPT will be provided alongside description of the method. This includes the construction of a statistical model of synthetic lethality in (continuous multivariate Guassian) gene expression data. This enables testing SLIPT upon simulated data with known synthetic lethal partners. Another key component of the simulation pipeline used later is the generation of simulated data from a known graph structure or simulated biological pathway. The development of this simulation procedure and other statistical treatment of graph and network structures will laos be covered here. Various R packages have been developed to support this project, most notably the slipt package to implement the SLIPT methodology. Numerous other R packages for handling graph structures, simulations, and custom plotting features will also be described as research outputs of this thesis, methods applied throughout, and contributions to the open-source software community that made this project feasible.

## A Synthetic Lethal Detection Methodology

The SLIPT methodology identifies gene expression patterns consistent with synthetic lethal interactions between a query gene and a panel of candidate interacting partners. Gene expression is called low, medium, or high by separating samples into tertiles (3-quantiles) for each gene. Genes with insufficient expression across all samples were excluded by requiring that the first tertile of raw counts is above zero. Then a *χ*2 test is performed between the query gene and each candidate partner, the p-values for the *χ*2 test were corrected for multiple testing using false discovery rate (FDR) error control to reduce false positives for large candidate gene panels . Significance was called only if FDR adjusted p-values were below the threshold *p* < 0. 05. A synthetic lethal interaction is predicted (as shown in Figure [fig:SLIPTMethod]) when (i) the *χ*2 test is significant; (ii) observed low-query, low-candidate samples are less frequent than expected; and (iii) observed low-query, high-candidate and high-query, low-candidate samples are more frequent than expected.

[!ht]

[fig:SLIPTMethod]

The synthetic lethal prediction procedure has also been adapted to utilise somatic mutation data for the query gene. This is intended to utilise a query gene known to be recurrently mutated in the disease (and dataset) with the majority of mutations disrupting gene function (such as null or frame shift mutations). A synthetic lethal interaction is predicted (as shown in Figure [fig:SLIPTMethodmtSL]) when (i) the *χ*2 test is significant; (ii) observed mutant-query, low-candidate samples are less frequent than expected; and (iii) observed mutant-query, high-candidate and wild-type-query, low-candidate samples are more frequent than expected. Where these are compared, we contrast those generated entirely from expression data (exprSL) with those using mutation data for the query gene (mtSL). Unless otherwise specified, computationally predicted synthetic lethal gene candidates from SLIPT used the exprSL procedure.

[!ht]

[fig:SLIPTMethodmtSL]

### Supporting Data

A series of experimental genome-wide siRNA screens have been performed on synthetic lethal partners of *CDH1* . The strongest candidates from a primary screen were subject to a further secondary screen with the strongest validation being independent replication with 4 gene knockdowns with different targeting siRNA. As shown in Table [tab:secondaryscreen], there is significant (*p* = 7. 49 × 10− 3 by Fisher’s exact test) association between SLIPT candidates and stronger validations of siRNA candidates. Since there were more SLIPT- genes among those not validated and more SLIPT+ genes among those validated with several siRNAs, this supports the use of SLIPT as a synthetic lethal discovery procedure which may augment such screening experiments.

[!ht] [tab:secondaryscreen]

## Simulations and Modelling Synthetic Lethality in Expression Data

A statistical model of Synthetic Lethality was developed upon which to test the SLIPT procedure on simulated data. This section will describe the synthetic lethal model and the simulation procedure for generating gene expression data with known synthetic lethal partners. Some preliminary results to support usage of the SLIPT methodology throughout this thesis will be presented here. The simulation procedure will be applied in more depth in a later chapter, including in combination with simulations from graph structures.

### A Model of Synthetic Lethality

We first construct a conceptual model of synthetic lethality (see Figure [fig:SLModel]). This will be used to build a statistical model of synthetic lethal gene expression from which to simulate expression data to test SLIPT and various potential synthetic lethal prediction methods upon. In the model, synthetic lethality arises from related functions and manifests in a cell death phenotype.

This is model suggests that synthetic lethality is detectable in measures of gene inactivation across a sample population, namely mutation, DNA copy number, DNA methylation, and suppression of expression. All of these potential mechanisms of gene inactivation could lead to synthetic lethality but expression data is readily available and changes in these alternative mechanisms are likely to impact on the amount of expressed (functional) RNA or protein detectable. There are several ways that functional relationships between genes could manifest in expression data, including coexpression, mutual exclusivity and directional shifts. Co-expression is overly simplistic and has performed poorly in previous studies as a predictor of synthetic lethality, this will still be tested with Correlation measures in later simulations. Here the alternative hypothesis is that synthetic lethality will lead to a detectable directional shift. This does not preclude mutual exclusivity, compensating expression, or co-loss under-representation as discussed in previous studies.

[!ht]

[fig:SLModel]

The first condition of the synthetic lethal model is that if there are only 2 synthetic lethal genes (1 true partner), then they will not both be non-functional in the same sample (in an ideal model). The frequency of gene function is an arbitrary level for non-functional and functional samples, for the purposes of modeling synthetic lethality we have used 30%, an estimate of the number of non-functional E-cadherin samples in the TCGA breast cancer dataset. In this ideal case, we would not expect to observe any samples lowly expressing both of these genes. While this is not observed, that is to be expected as it is unlikely that only 2 genes will have an exclusive synthetic lethal partnership.

[!ht]

[fig:SLModelHigher]

A synthetic lethal pair of genes is unlikely to act in isolation, higher-order synthetic lethal interactions (or 3 or more genes) must be considered in the model as shown in Figure [fig:SLModelHigher]. Even when testing pairwise interactions, modelling higher level interactions that may interfere with it is important. If there are additional synthetic lethal partners, there are two possibilities for adding these: 1) that they are independent partners of the query genes interacting pairwise (and not with each other) or 2) that it interacts with both of the synthetic lethal genes already in the system and any of the 3 (or more) are required to be functional for the cell to survive. The signal in gene expression data will be weaker for this latter case and thus we make the more stringent assumption that all synthetic lethal partner genes interact with one and other: that only one of these must be expressed to satisfy the model of synthetic lethality. In this model any of the synthetic lethal genes in a higher-order interaction is able to provide the missing function of the others, allowing for higher-level synthetic lethal partners to compensate for loss a synthetic lethal gene pair. While samples lowly expression synthetic lethal gene pairs will be under-represented, they may not be completely excluded from the dataset due to these higher-level interactions.

[!ht]

[fig:SLModelExpression]

To model synthetic lethal gene expression, several synthetic lethal genes are considered. In Figure [fig:SLModelExpression], an example of 3 synthetic lethal genes is presented, where any of the higher-order synthetic lethal interaction are required for cell viability. Expression data is assumed to be continuous, normally distributed data with an arbitrary threshold to define gene function. As discussed above this threshold is *pr* = 0. 3 throughout this thesis but is amenable to further modifications. Thus, if we consider synthetic lethal pairs, only a subset of these samples will be excluded, leading to an under-representation.

In practice, samples are not removed from a dataset, rather the query gene is added to satisfy the synthetic lethal condition as outlined in section [methods:simulating SL] below. This is performed to maintain a comparable samples size across simulations and the preserve the assumed (multivariate) normal distribution of the data.

### Simulation Procedure

Simulations were developed to simulate normal distributions of expression data and define function with a threshold cut-off. This is the reverse to the procedure of SLIPT to predict synthetic lethal partners (although the threshold is assumed to be unknown when testing upon simulated data). While gene function is used as an intermediary step in modelling synthetic lethal genes in expression data, the normal distribution gives a more comparable test of SLIPT (and other methods) on simulated data compared to normalised empirical gene expression data. This also has the added advantage of being amenable to simulating correlation structures with the Multivariate normal distribution. In Figure [fig:simulatefunction], an example of such a multivariate normal dataset is shown with the functional threshold applied.

[!ht]

[fig:simulatefunction]

Once we have generated a simulated dataset, the samples are compared by gene function (as derived from a functional threshold). Known underlying synthetic lethal partners are selected within the dataset and a query gene is generated by sampling from the normal distribution. These are matched (as shown for 2 synthetic lethal partners in Figure [fig:simulateaddquery]) such that the synthetic lethal condition is met: that at least one of the synthetic partner genes and the query gene are functional in any particular cell. This is done by ordering the samples by functional data (without assuming correlation of underyling expression values) with the query gene in one direction and the remaining dataset ordered by the selected synthetic lethal partner.

[!ht]

[fig:simulateaddquery]

This results in the simulated dataset where sample with non-functional query gene has loss of function in all of the synthetic lethal partners as at least one has been required to be functional in each sample, similarly the query gene is functional where the synthetic lethal partner genes are non-functional. Therefore we have generated a dataset with known synthetic lethal partners (see Figure [fig:simulateSL]) by as few assumptions about the relationships between the each synthetic lethal pair as possible (and allowing compensating functions from higher-order interactions). This has been designed to have the most stringent (least detectable) synthetic lethal relationships where higher-order interactions are possible for the purposes of testing pairwise detection procedures such as SLIPT.

[!ht]

[fig:simulateSL]

## Detecting Simulated Synthetic Lethal Partners

Test

### Multivariate Normal Simulation of Synthetic lethality

Test

#### Simulation with Directional Testing

Test

#### Simulation with Query-Correlated Pathways

Test

#### Simulated Expression Heatmaps

Test

### Replication Simulation Heatmap

Test

## Graph Structure Methods

Test

### Upstream and Downstream Gene Detection

Test

#### Permutation Analysis for Statisical Significance

Test

### Simulating Gene Expression from Graph Structures

A further refinement of the simulation procedure is to generate expression data with correlation structure derived from a known graph structure. This enables modelling of synthetic lethal partners within a biological pathway and the impact of pathway structure on synthetic lethal prediction. First we construct a simulated pathway with a graph structure with the igraph R package with the added provision of including the state of the edges, i.e., whether the activate or inhibit downstream pathway members. Here we consider purely whether biological pathway members would be expected to have correlated gene expression (higher than the background of genes in other pathways) but this framework is as applicable to modelling protein levels in a kinase regulation cascade or metabolic pathway with related substrates and products as it is to gene regulation as it has been applied here.

[!ht]

[fig:simplegraph]

First we must consider the graph structure upon which simulated data will be generated (by sampling a multivariate normal distribution). Throughout this section, the simulation procedure will be demonstrated with the relatively simple constructed graph structure shown in Figure [fig:simplegraph]. This graph structure visualisation was developed specifically for (directed) igraph objects in R and has been released in the plot.igraph package and igraph.extensions library. The plot\_directed function allows customisation of plot parameters for each node or edge and mixed (directed) edge types for indication activation or inhibition. These inhibition links (which often occur in biological pathways) are demonstrated in Figure [fig:simplegraph:second].

[!ht]

[fig:simulationactivating]

The simulation procedure is designed to use such graph structures to inform development of a “Sigma” correlation-covariate matrix for sampling a multivariate normal distribution (using the mvtnorm R package). Given a graph structure (or adjacency matrix), such as Figure [fig:simulationactivating:first], a relation matrix is calculated based on distance such that nearer nodes are weighted higher than farther nodes. For the purposes of this thesis a geometrically decreasing (relative) distance weighting is used, with each more distant node being related by $\frac{1}{2}$ compared to the next nearest as shown in Figure [fig:simulationactivating:second]. However, an arithmetically decreasing (absolute) distance weighting is also available in the graphsim R package release of this procedure.

A Sigma matrix is derived from this distance weighting matrix by creating a matrix where each node is correlated with itself (diagonal of 1), correlated to adjacent nodes by a correlation parameter (cor), and the remaining matrix based on weighting these correlations with by the distance matrix (or the nearest positive definite matrix). For the purposes of this thesis the correlation parameter, cor = 0. 8, unless otherwise specified as used for the example in Figure [fig:simulationactivating:third]. This Sigma matrix is used to sample from a multivariate normal distribution where each node has a mean of 0, standard deviation 1 and covariates within the range [0, 1]. This generates a simulated (continuous normally distributed) expression profile for each node (as shown in Figure [fig:simulationactivating:fourth]) with corresponding correlation structure (Figure [fig:simulationactivating:fifth]). This simulated correlation structure closely resembles the expected correlation structure (Sigma in [fig:simulationactivating:third]) even for the relatively modest sample size (*N* = 100) illustrated in [fig:simulationactivating]. Once a simulated gene expression dataset has been generated (as in Figure [fig:simulationactivating:fourth]) then a discrete matrix of gene function can be constructed with a functional threshold quintile (the parameter pr) to simulate functional relationships of synthetic lethality. For the purposes of this thesis the parameter pr is set to the 0.3 quintile which generates functional discrete matrices such as those used for synthetic lethal simulation in section [methods:simulating SL] (as shown Figure [fig:simulationactivating:sixth])

[!ht]

[fig:simulationinhibiting]

The simulation procedure discussed in Figure [fig:simulationactivating] is amenable to pathways containing inhibition links (as shown in Figure [fig:simulationinhibiting]) with a few refinements. With the inhibition links (as shown in Figure [fig:simulationinhibiting:first]), the distances are calculated in the same manner as before (Figure [fig:simulationinhibiting:second]) but the inhibitions are accounted for by iteratively multiplying downstream nodes by  − 1 to form blocks of negative correlations (as shown in Figures [fig:simulationinhibiting:third] and [fig:simulationinhibiting:fifth]). As before, a multivariate normal distribution with these negative correlations can be sampled to generate simulated data (as shown in Figures [fig:simulationinhibiting:fourth] and [fig:simulationinhibiting:sixth]).

These simulated datasets are amenable to simulating synthetic lethal partners of a query gene within a graph network. The query gene is assumed to be separate from the graph network pathway and is added to the dataset using the procedure in Section [methods:simulating SL]. Thus we can simulate known synthetic lethal partner genes within a synthetic lethal partner pathway structure.

*image* *image* *image* *image* *image*

*image* *image* *image* *image* *image* *image*

## Customised methods and R packages developed

test

### slipt

test

### plotting

test

### simulation from graph structures

test

### igraph methods

test

# Sample Correlation

[!ht]

[fig:corrmappart]

[!ht]

[fig:corrmap]

# Software Used for Thesis

|llllll| [tab:computersrpackagesfull]  
 & & & & &  
 base & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
abind & CRAN & & 1.4-5 & & 1.4-3  
 acepack & CRAN & & 1.4.1 & & 1.3-3.3  
ade4 & CRAN & & 1.7-5 & &  
 annaffy & Bioconductor & & 1.46.0 & &  
AnnotationDbi & Bioconductor & & 1.36.0 & 1.36.0 & 1.34.4  
 apComplex & CRAN & & 2.40.0 & &  
ape & CRAN & & 4 & & 3.4  
 arm & CRAN & & 1.9-3 & &  
assertthat & CRAN & 0.1 & 0.1 & 0.1 & 0.1  
 backports & CRAN & 1.0.5 & 1.0.4 & 1.0.5 & 1.0.2  
base64 & CRAN & & & 2 & 2  
 base64enc & CRAN & & 0.1-3 & & 0.1-3  
beanplot & CRAN & & 1.2 & 1.2 & 1.2  
 BH & CRAN & 1.60.0-2 & 1.62.0-1 & 1.62.0-1 & 1.60.0-2  
Biobase & Bioconductor & & 2.34.0 & 2.34.0 & 2.32.0  
 BiocGenerics & Bioconductor & & 0.20.0 & 0.20.0 & 0.18.0  
BiocInstaller & Bioconductor & & 1.24.0 & 1.20.3 & 1.22.3  
 BiocParallel & Bioconductor & & 1.8.1 & 1.8.1 &  
Biostrings & Bioconductor & & 2.42.1 & 2.42.0 &  
 BiSEp & Bioconductor & & 2.0.1 & 2.0.1 & 2.0.1  
bitops & CRAN & 1.0-6 & 1.0-6 & 1.0-6 & 1.0-6  
 boot & base & 1.3-18 & 1.3-18 & 1.3-18 & 1.3-18  
brew & CRAN & 1.0-6 & 1.0-6 & 1.0-6 & 1.0-6  
 broom & CRAN & 0.4.1 & & &  
caTools & CRAN & 1.17.1 & 1.17.1 & 1.17.1 & 1.17.1  
 cgdsr & CRAN & & 1.2.5 & &  
checkmate & CRAN & & 1.8.2 & & 1.7.4  
 chron & CRAN & 2.3-47 & 2.3-48 & 2.3-50 & 2.3-47  
class & base & 7.3-14 & 7.3-14 & 7.3-14 & 7.3-14  
 cluster & base & 2.0.5 & 2.0.5 & 2.0.5 & 2.0.4  
coda & CRAN & & 0.19-1 & & 0.18-1  
 codetools & base & 0.2-15 & 0.2-15 & 0.2-15 & 0.2-14  
colorRamps & CRAN & & 2.3 & &  
 colorspace & CRAN & 1.2-6 & 1.3-2 & 1.3-2 & 1.2-6  
commonmark & CRAN & 1.1 & & 1.2 &  
 compiler & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
corpcor & CRAN & & 1.6.8 & 1.6.8 & 1.6.8  
 Cprob & CRAN & & 1.2.4 & &  
crayon & CRAN & 1.3.2 & 1.3.2 & 1.3.2 & 1.3.2  
 crop & CRAN & & 0.0-2 & 0.0-2 &  
curl & CRAN & 1.2 & 2.3 & 2.3 & 0.9.7  
 d3Network & CRAN & & 0.5.2.1 & &  
data.table & CRAN & 1.9.6 & 1.10.0 & 1.10.1 & 1.9.6  
 data.tree & CRAN & & 0.7.0 & 0.7.0 &  
datasets & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 DBI & CRAN & 0.5-1 & 0.5-1 & 0.5-1 & 0.5-1  
dendextend & CRAN & 1.4.0 & 1.4.0 & 1.4.0 &  
 DEoptimR & CRAN & 1.0-8 & 1.0-8 & 1.0-8 & 1.0-4  
desc & CRAN & 1.1.0 & & 1.1.0 &  
 devtools & CRAN & 1.12.0 & 1.12.0 & 1.12.0 & 1.12.0  
DiagrammeR & CRAN & & 0.9.0 & 0.9.0 &  
 dichromat & CRAN & 2.0-0 & 2.0-0 & 2.0-0 & 2.0-0  
digest & CRAN & 0.6.10 & 0.6.11 & 0.6.12 & 0.6.9  
 diptest & CRAN & 0.75-7 & 0.75-7 & 0.75-7 &  
doParallel & CRAN & 1.0.10 & 1.0.10 & 1.0.10 & 1.0.10  
 dplyr & CRAN & 0.5.0 & 0.5.0 & 0.5.0 & 0.5.0  
ellipse & CRAN & & 0.3-8 & 0.3-8 & 0.3-8  
 evaluate & CRAN & & 0.1 & 0.1 & 0.9  
fdrtool & CRAN & & 1.2.15 & &  
 fields & CRAN & & 8.1 & &  
flexmix & CRAN & 2.3-13 & 2.3-13 & 2.3-13 &  
 forcats & CRAN & 0.2.0 & & &  
foreach & CRAN & 1.4.3 & 1.4.3 & 1.4.3 & 1.4.3  
 foreign & base & 0.8-67 & 0.8-67 & 0.8-67 & 0.8-66  
formatR & CRAN & & 1.4 & 1.4 & 1.4  
 Formula & CRAN & & 1.2-1 & & 1.2-1  
fpc & CRAN & 2.1-10 & 2.1-10 & 2.1-10 &  
 futile.logger & CRAN & & 1.4.3 & 1.4.3 & 1.4.1  
futile.options & CRAN & & 1.0.0 & 1.0.0 & 1.0.0  
 gdata & CRAN & 2.17.0 & 2.17.0 & 2.17.0 & 2.17.0  
geepack & CRAN & & 1.2-1 & &  
 GenomeInfoDb & Bioconductor & & 1.10.2 & 1.10.1 &  
GenomicAlignments & Bioconductor & & 1.10.0 & 1.10.0 &  
 GenomicRanges & Bioconductor & & 1.26.2 & 1.26.1 &  
ggm & CRAN & & 2.3 & &  
 ggplot2 & CRAN & 2.1.0 & 2.2.1 & 2.2.1 & 2.1.0  
git2r & CRAN & 0.15.0 & 0.18.0 & 0.16.0 & 0.15.0  
 glasso & CRAN & & 1.8 & &  
GO.db & Bioconductor & & 3.4.0 & 3.2.2 & 3.3.0  
 GOSemSim & Bioconductor & & 2.0.3 & 1.28.2 & 1.30.3  
gplots & CRAN & 3.0.1 & 3.0.1 & 3.0.1 & 3.0.1  
 graph & Bioconductor & & 1.52.0 & &  
graphics & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 graphsim &

[c]@l@GitHub  
TomKellyGenetics

& 0.1.0 & 0.1.0 & 0.1.0 & 0.1.0  
grDevices & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 grid & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
gridBase & CRAN & 0.4-7 & 0.4-7 & 0.4-7 & 0.4-7  
 gridExtra & CRAN & 2.2.1 & 2.2.1 & 2.2.1 & 2.2.1  
gridGraphics & CRAN & & 0.1-5 & &  
 gtable & CRAN & 0.2.0 & 0.2.0 & 0.2.0 & 0.2.0  
gtools & CRAN & 3.5.0 & 3.5.0 & 3.5.0 & 3.5.0  
 haven & CRAN & 1.0.0 & & &  
heatmap.2x &

[c]@l@GitHub  
TomKellyGenetics

& 0.0.0.9000 & 0.0.0.9000 & 0.0.0.9000 & 0.0.0.9000  
 hgu133plus2.db & Bioconductor & & 3.2.3 & &  
highr & CRAN & & 0.6 & 0.6 & 0.6  
 Hmisc & CRAN & & 4.0-2 & 4.0-2 & 3.17-4  
hms & CRAN & 0.2 & 0.3 & &  
 htmlTable & CRAN & & 1.8 & 1.9 &  
htmltools & CRAN & 0.3.5 & 0.3.5 & 0.3.5 & 0.3.5  
 htmlwidgets & CRAN & & 0.8 & 0.8 &  
httpuv & CRAN & 1.3.3 & & 1.3.3 &  
 httr & CRAN & 1.2.1 & 1.2.1 & 1.2.1 & 1.1.0  
huge & CRAN & & 1.2.7 & &  
 hunspell & CRAN & & 2.3 & & 2  
hypergraph & CRAN & & 1.46.0 & &  
 igraph & CRAN & 1.0.1 & 1.0.1 & 1.0.1 & 1.0.1  
igraph.extensions &

[c]@l@GitHub  
TomKellyGenetics

& 0.1.0.9001 & 0.1.0.9001 & 0.1.0.9001 & 0.1.0.9001  
 influenceR & CRAN & & 0.1.0 & 0.1.0 &  
info.centrality &

[c]@l@GitHub  
TomKellyGenetics

& 0.1.0 & 0.1.0 & 0.1.0 & 0.1.0  
 IRanges & Bioconductor & & 2.8.1 & 2.8.1 & 2.6.1  
irlba & CRAN & 2.1.1 & 2.1.2 & 2.1.2 & 2.0.0  
 iterators & CRAN & 1.0.8 & 1.0.8 & 1.0.8 & 1.0.8  
jpeg & CRAN & & 0.1-8 & &  
 jsonlite & CRAN & 1.1 & 1.2 & 1.3 & 0.9.20  
KEGG.db & Bioconductor & & 3.2.3 & &  
 kernlab & CRAN & 0.9-25 & 0.9-25 & 0.9-25 &  
KernSmooth & base & 2.23-15 & 2.23-15 & 2.23-15 & 2.23-15  
 knitr & CRAN & & 1.15.1 & 1.15.1 & 1.14  
labeling & CRAN & 0.3 & 0.3 & 0.3 & 0.3  
 lambda.r & CRAN & & 1.1.9 & 1.1.9 & 1.1.7  
lattice & base & 0.20-34 & 0.20-34 & 0.20-34 & 0.20-33  
 latticeExtra & CRAN & & 0.6-28 & & 0.6-28  
lava & CRAN & & 1.4.6 & &  
 lavaan & CRAN & & 0.5-22 & &  
lazyeval & CRAN & 0.2.0 & 0.2.0 & 0.2.0 & 0.2.0  
 les & CRAN & & 1.24.0 & &  
lgtdl & CRAN & & 1.1.3 & &  
 limma & Bioconductor & & 3.30.7 & 3.30.3 &  
lme4 & CRAN & & 1.1-12 & & 1.1-12  
 lubridate & CRAN & 1.6.0 & & &  
magrittr & CRAN & 1.5 & 1.5 & 1.5 & 1.5  
 maps & CRAN & & 3.1.1 & &  
markdown & CRAN & & 0.7.7 & 0.7.7 & 0.7.7  
 MASS & base & 7.3-45 & 7.3-45 & 7.3-45 & 7.3-45  
Matrix & base & 1.2-7.1 & 1.2-7.1 & 1.2-8 & 1.2-6  
 matrixcalc & CRAN & 1.0-3 & 1.0-3 & 1.0-3 & 1.0-3  
mclust & CRAN & 5.2 & 5.2.1 & 5.2.2 & 5.2  
 memoise & CRAN & 1.0.0 & 1.0.0 & 1.0.0 & 1.0.0  
methods & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 mgcv & base & 1.8-16 & 1.8-16 & 1.8-17 & 1.8-12  
mi & CRAN & & 1 & &  
 mime & CRAN & 0.5 & 0.5 & 0.5 & 0.4  
minqa & CRAN & & 1.2.4 & & 1.2.4  
 mnormt & CRAN & 1.5-5 & 1.5-5 & & 1.5-4  
modelr & CRAN & 0.1.0 & & &  
 modeltools & CRAN & 0.2-21 & 0.2-21 & 0.2-21 &  
multtest & Bioconductor & & 2.30.0 & 2.30.0 &  
 munsell & CRAN & 0.4.3 & 0.4.3 & 0.4.3 & 0.4.3  
mvtnorm & CRAN & 1.0-5 & 1.0-5 & 1.0-6 & 1.0-5  
 network & CRAN & & 1.13.0 & &  
nlme & base & 3.1-128 & 3.1-128 & 3.1-131 & 3.1-128  
 nloptr & CRAN & & 1.0.4 & & 1.0.4  
NMF & CRAN & 0.20.6 & 0.20.6 & 0.20.6 & 0.20.6  
 nnet & base & 7.3-12 & 7.3-12 & 7.3-12 & 7.3-12  
numDeriv & CRAN & & 2016.8-1 & & 2014.2-1  
 openssl & CRAN & 0.9.4 & 0.9.6 & 0.9.6 & 0.9.4  
org.Hs.eg.db & Bioconductor & & 3.1.2 & & 3.3.0  
 org.Sc.sgd.db & Bioconductor & & 3.4.0 & &  
parallel & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0

[c]@l@pathway.structure  
.permutation

&

[c]@l@GitHub  
TomKellyGenetics

& 0.1.0 & 0.1.0 & 0.1.0 & 0.1.0  
pbivnorm & CRAN & & 0.6.0 & &  
 PGSEA & Bioconductor & & 1.48.0 & &  
pkgmaker & CRAN & 0.22 & 0.22 & 0.22 & 0.22  
 PKI & CRAN & & 0.1-3 & &  
plogr & CRAN & & 0.1-1 & 0.1-1 &  
 plot.igraph &

[c]@l@GitHub  
TomKellyGenetics

& 0.0.0.9001 & 0.0.0.9001 & 0.0.0.9001 & 0.0.0.9001  
plotrix & CRAN & & 3.6-4 & &  
 plyr & CRAN & 1.8.4 & 1.8.4 & 1.8.4 & 1.8.3  
png & CRAN & & 0.1-7 & & 0.1-7  
 prabclus & CRAN & 2.2-6 & 2.2-6 & 2.2-6 &  
praise & CRAN & 1.0.0 & 1.0.0 & & 1.0.0  
 pROC & CRAN & & 1.8 & 1.9.1 &  
prodlim & CRAN & & 1.5.7 & &  
 prof.tree & CRAN & & 0.1.0 & &  
proftools & CRAN & & 0.99-2 & &  
 progress & CRAN & & & 1.1.2 &  
psych & CRAN & 1.6.12 & 1.6.12 & &  
 purrr & CRAN & 0.2.2 & 0.2.2 & 0.2.2 & 0.2.2  
qgraph & CRAN & & 1.4.1 & &  
 quadprog & CRAN & & 1.5-5 & 1.5-5 & 1.5-5  
R.methodsS3 & CRAN & & 1.7.1 & & 1.7.1  
 R.oo & CRAN & & 1.21.0 & & 1.20.0  
R.utils & CRAN & & 2.5.0 & &  
 R6 & CRAN & 2.1.3 & 2.2.0 & 2.2.0 & 2.1.3  
RBGL & CRAN & & 1.50.0 & &  
 RColorBrewer & CRAN & 1.1-2 & 1.1-2 & 1.1-2 & 1.1-2  
Rcpp & CRAN & 0.12.7 & 0.12.9 & 0.12.9 & 0.12.7  
 RcppArmadillo & CRAN & & & 0.7.700.0.0 & 0.6.700.6.0  
RcppEigen & CRAN & & 0.3.2.9.0 & & 0.3.2.8.1  
 RCurl & CRAN & & 1.95-4.8 & 1.95-4.8 & 1.95-4.8  
reactome.db & Bioconductor & & 1.52.1 & 1.52.1 &  
 reactometree &

[c]@l@GitHub  
TomKellyGenetics

& & 0.1 & &  
readr & CRAN & 1.0.0 & 1.0.0 & &  
 readxl & CRAN & 0.1.1 & & &  
registry & CRAN & 0.3 & 0.3 & 0.3 & 0.3  
 reshape2 & CRAN & 1.4.1 & 1.4.2 & 1.4.2 & 1.4.1  
rgexf & CRAN & & 0.15.3 & 0.15.3 &  
 rgl & CRAN & & & 0.97.0 & 0.95.1441  
Rgraphviz & CRAN & & 2.18.0 & &  
 rjson & CRAN & & 0.2.15 & &  
RJSONIO & CRAN & & 1.3-0 & &  
 rmarkdown & CRAN & & 1.3 & 1.3 & 1  
Rmpi & CRAN & & 0.6-6 & & 0.6-5  
 rngtools & CRAN & 1.2.4 & 1.2.4 & 1.2.4 & 1.2.4  
robustbase & CRAN & 0.92-7 & 0.92-7 & 0.92-7 & 0.92-5  
 ROCR & CRAN & 1.0-7 & 1.0-7 & 1.0-7 & 1.0-7  
Rook & CRAN & & 1.1-1 & 1.1-1 &  
 roxygen2 & CRAN & 6.0.1 & 5.0.1 & 6.0.1 & 5.0.1  
rpart & base & 4.1-10 & 4.1-10 & 4.1-10 & 4.1-10  
 rprojroot & CRAN & 1.2 & 1.1 & 1.2 &  
Rsamtools & Bioconductor & & 1.26.1 & 1.26.1 &  
 rsconnect & CRAN & & 0.7 & &  
RSQLite & CRAN & & 1.1-2 & 1.1-2 & 1.0.0  
 rstudioapi & CRAN & 0.6 & 0.6 & 0.6 & 0.6  
rvest & CRAN & 0.3.2 & & &  
 S4Vectors & Bioconductor & & 0.12.1 & 0.12.0 & 0.10.3  
safe & Bioconductor & & 3.14.0 & 3.10.0 &  
 scales & CRAN & 0.4.0 & 0.4.1 & 0.4.1 & 0.4.0  
selectr & CRAN & 0.3-1 & & &  
 sem & CRAN & & 3.1-8 & &  
shiny & CRAN & 0.14 & & 1.0.0 &  
 slipt &

[c]@l@GitHub  
TomKellyGenetics

& 0.1.0 & 0.1.0 & 0.1.0 & 0.1.0  
sm & CRAN & 2.2-5.4 & 2.2-5.4 & &  
 sna & CRAN & & 2.4 & &  
snow & CRAN & 0.4-1 & 0.4-2 & 0.4-2 & 0.3-13  
 sourcetools & CRAN & 0.1.5 & & 0.1.5 &  
SparseM & CRAN & & 1.74 & & 1.7  
 spatial & base & 7.3-11 & 7.3-11 & 7.3-11 & 7.3-11  
splines & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 statnet.common & CRAN & & 3.3.0 & &  
stats & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 stats4 & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
stringi & CRAN & 1.1.1 & 1.1.2 & 1.1.2 & 1.0-1  
 stringr & CRAN & 1.1.0 & 1.1.0 & 1.2.0 & 1.0.0

[c]@l@Summarized  
Experiment

& Bioconductor & & 1.4.0 & 1.4.0 &  
 survival & base & 2.39-4 & 2.40-1 & 2.40-1 & 2.39-4  
tcltk & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 testthat & CRAN & 1.0.2 & 1.0.2 & & 1.0.2  
tibble & CRAN & 1.2 & 1.2 & 1.2 & 1.2  
 tidyr & CRAN & 0.6.1 & 0.6.1 & 0.6.1 &  
tidyverse &

[c]@l@GitHub  
hadley

& 1.1.1 & & &  
 timeline & CRAN & & 0.9 & &  
tools & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 tpr & CRAN & & 0.3-1 & &  
trimcluster & CRAN & 0.1-2 & 0.1-2 & 0.1-2 &  
 Unicode & CRAN & 9.0.0-1 & 9.0.0-1 & 9.0.0-1 &  
utils & base & 3.3.2 & 3.3.2 & 3.3.1 & 3.3.0  
 vioplot & CRAN & & 0.2 & &  
vioplotx &

[c]@l@GitHub  
TomKellyGenetics

& 0.0.0.9000 & 0.0.0.9000 & &  
 viridis & CRAN & 0.3.4 & 0.3.4 & 0.3.4 &  
visNetwork & CRAN & & 1.0.3 & 1.0.3 &  
 whisker & CRAN & 0.3-2 & 0.3-2 & 0.3-2 & 0.3-2  
withr & CRAN & 1.0.2 & 1.0.2 & 1.0.2 & 1.0.2  
 XML & base & 3.98-1.3 & 3.98-1.1 & 3.98-1.5 & 3.98-1.4  
xml2 & CRAN & 1.1.1 & & 1.1.1 & 1.0.0  
 xtable & CRAN & 1.8-2 & 1.8-2 & 1.8-2 & 1.8-2  
XVector & Bioconductor & & 0.14.0 & 0.14.0 &  
 yaml & CRAN & & 2.1.14 & 2.1.14 & 2.1.13  
zlibbioc & CRAN & & 1.20.0 & 1.20.0 &  
 zoo & CRAN & 1.7-13 & 1.7-14 & & 1.7-13

# Source Code for thesis.dvi

## thesis.tex

### abstract.tex

### acknowledgements.tex

### intro.tex

### literature.tex

### new\_ideas.tex

### implementation.tex

### results.tex

### conclusion.tex

### appendices.tex