Internal Report 2011–number September 2011

Leiden University

Track Bioinformatics

A Gaussian Random Field Algorithm used to Analyse Microarray Data from Multiple Species

Orr Shomroni

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Evolutionary conservation implies that similar biological mechanisms occur in similar species, and those mechanisms can be studied using cross-species studies. Cross-species studies use multiple microarray experiments from different species to study gene expressions in those species simultaneously, where these studies can either analyse the microarrays separately and compare the results between species, or combine the data from all microarrays and analyse all gene expressions together. However, challenges arise during cross-species analyses, such as varying gene expressions in different species, noisy data and discrete homology assignments. This study is therefore aimed at suggesting an algorithm that is able to deal with all those challenges in cross-species analyses and obtain biologically relevant results from microarray experiments. The algorithm suggested here is a new and promising method that so far has only been applied to a cross-species study on immune response genes. This study will therefore use this algorithm for a different biological case study (brain ageing), and the sensitivity (choice of parameters) and scalability (convergence and runtime) of the algorithm will be assessed.

Acknowledgments

I would like to thank my supervisor Michael Emmerich from Leiden Institute of Advanced Computer Science (LIACS) for helping me better understand the algorithm and giving me many comments about the thesis itself. I would like to thank my second supervisor Erwin Bakker from LIACS for giving me feedback on the thesis and some useful comments. I would also like to thank Prof. Jan Hendrik Hoogeboom from LIACS for assisting me in creating the alignment matrix needed for the algorithm. In addition, I would like to thank Edmondo Trentin from Universita di Siena, Italy, who gave me some pointers on graph theory that made me think about the principles behind the GRF. Finally, I would like to thank Yong Lu from Harvard Medical School, USA, who helped me better understand some parts of his paper ”Cross species expression analysis of innate immune response” from 2010, which was used to design the algorithm in this thesis.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Cross-species data analysis

1.1 Background of cross-species data analysis

Due to evolutionary conservation, certain biological mechanisms are conserved in evolutionary-similar organisms, and so many of the genes in those mechanisms are conserved as well[54]. Early cross-species analyses involved analysing sequence similarity between species, using those similarities to identify core interaction modules and compare regulatory mechanisms in related species[42, 47]. Nevertheless, sequence similarity involves static data (DNA sequences), and since interactions and expressions of genes change across time and in different conditions, sequence analysis is not powerful enough to study such interactions[24, 31]. For this reason, researchers use microarrays to measure dynamic, condition-specific gene expressions in order to analyse these interactions and identify deregulated genes that are influenced by transcriptional changes (genes that change in statistically significant terms across different biological conditions). By studying certain transcriptional changes and deregulated genes in multiple species, it is possible to identify core groups of deregulated genes that are conserved across those species, thus showing major players in conserved biological mechanisms.

1.2 Challenges in cross-species analysis

When studying a microarray from a single species for a single condition, challenges that need to be addressed include searching in large datasets and noisy data. However, when considering multiple microarrays for multiple species, additional challenges include (1) handling homology assignments between species, (2) comparing continuous gene expressions in different species (sequence data is only based on 4 letters, while expressions of homologous genes might vary largely in different species), (3) a wide range of conditions makes it difficult to compare data between species (extensive biological variations), and (4) different microarray studies use different analysis techniques. In addition, since cross-species analysis often involves comparing data from different laboratories, the data is even noisier than when dealing with a single microarray. To overcome these issues, there are 3 general approaches that can be applied to perform cross-species analysis of microarray data: (1) studying individual microarrays and then combining their results (meta-analysis); (2) using the same microarray to study different species; and (3) analysing the data from all species concurrently (combined analysis)[36]. Since bioinformatics deals with the analysis of microarray data rather than with the microarray experiment design itself, approach (2) will not be dealt with here, and approaches (1) and (3) are discussed in depth.

1.3 State of the art

1.3.1 Meta-analysis

The meta-analysis of microarray data detects deregulated genes in different species with different gene expression ranges, finds common expression patterns in genes from multiple species and explains the evolution of gene expressions and co-regulation. This approach involves two general methods: co-expression meta-analysis and expression meta-analysis.

Co-expression meta-analysis searches for gene modules with similar expression patterns in individual species, and then finds modules of homologous genes that are co-expressed in other species. This technique can use data from different microarray studies with different experimental conditions, allowing genes to be studied in groups rather than individuals with noisy and varied expressions. This technique deals with challenges (2) and (3) in Section 1.2, as it does not compare between gene expressions from different species directly, and therefore does not deal with the extensive biological variations, but rather compares overall patterns that occur between the genes[36].

Expression meta-analysis directly analyses expression profiles of homologous genes in order to find differentially expressed genes (DEGs) conserved in multiple species. The DEGs are often derived from published papers on different microarrays in different species, and expression meta-analysis searches for overlapping DEGs between different species. Again, since this technique does not compare gene expressions from different species directly, it tackles challenges (2) and (3) in Section 1.2. Another variant of expression meta-analysis, called indirect comparisons, uses the DEGs in each species to find functional annotations (gene ontology (GO) categories, KEGG pathways) that are over-represented in this species, and then the overlap between functional annotations across different species can be found. Since this technique does not test for overlap between homologous genes, it allows to compare even distant species with few homologous genes, thus it also tackles challenge (1) in Section 1.2. In addition, indirect comparisons generate a more biologically significant answer than simply deriving overlapping homologous genes, as specific biological mechanisms and gene groups can be observed[36].

One challenge that meta-analysis still faces is the fact that different published papers use different analyses for their microarrays, and therefore they are inconsistent and difficult to compare. Nevertheless, this can be corrected by using published microarrays rather than results, and analysing them with the same technique to find overlapping deregulated genes.

1.3.2 Combined analysis

From Section 1.3.1, it is seen that indirect comparisons can account for all 4 challenges in cross-species analysis. However, it might still be necessary to search for 'core' groups of deregulated genes across different species, and in this case, meta-analysis does not sufficiently account for homology assignments between genes. Meta-analysis has a binary homology assignment (genes are either homologous or not), and this limits the comparison of differentially expressed genes to the homology annotations, without considering similarity measures of a continuous nature (e.g. E-value in BLAST). The combined analysis combines the microarray datasets from different species first, and then analyses all the data together, where it is also capable to integrate continuous homology scores in the analysis.

One algorithm which employs such continuous homology score approach was developed by Lu et al.[37]. This technique used Markov Random Fields (MRFs), an undirected graphical model that analysed cell cycle data from human and budding yeast. The model used genes from both species (and their expression patterns) as nodes, and similarity between sequences as edges. The scores along the edges allowed a more flexible threshold for the homology, which affected borderline scores by increasing scores for similarly expressed homologous genes and decreasing scores for non-homologous genes[37]. The technique was later adapted to study genes involved in immune responses in different species, different cell types and different bacteria types. The study used Gaussian Random Fields (GRFs), which are similar to MRFs, except that gene nodes in GRFs are represented by continuous random variables with Gaussian distributions, whereas MRFs are represented by discrete random variables that do not necessarily have Gaussian distributions (see Figure 1.1)[38]. The GRF technique deals with all challenges in Section 1.2: to deal with challenge (1), it uses continuous homology scores, which are more lenient than binary ones; to deal with challenges (2) and (3), the technique compares overall patterns of deregulated genes between both species (human and mouse), thus avoiding dealing with variable gene expressions; and to deal with challenge (4), all genes from different species are brought under the same graph and analysed with the same technique.

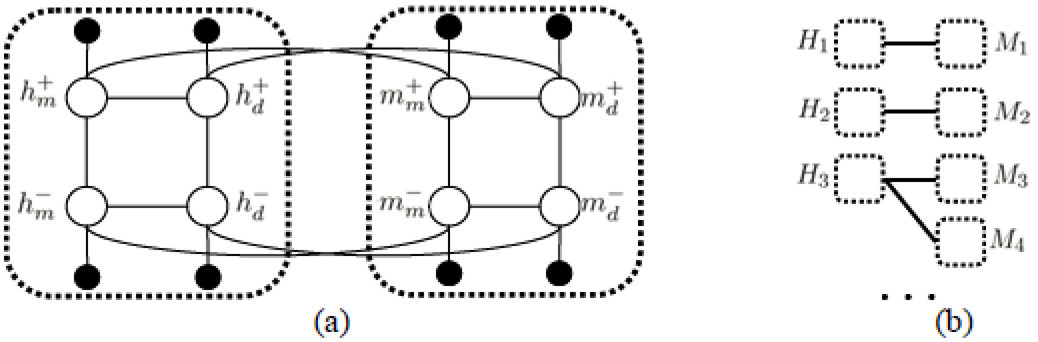


Figure 1.1: Gaussian random field model implemented by Lu et al. 2010
This Gaussian random field model was used to find genes involved in immune responses in different species (human/mouse marked as h/m respectively), different cell types (macrophage/dendritic marked as m/d respectively) and against different bacteria types (gram positive/negative marked as +/- respectively)[38]. Figure (a) shows white (latent) nodes representing class labels of genes under different conditions, and the black nodes represent expression scores of genes in those conditions. Figure (b) shows genes with edges between them to show they are similar to a certain extent.

1.4 Aim of study

As can be seen from Section 1.3, there have been multiple techniques suggested to account for the challenges of cross-species analysis, but the two most beneficial techniques suggested are indirect comparisons and combined analysis (specifically the GRF technique used by Lu et al. from 2010). Nevertheless, indirect comparisons can only account for functional annotations in a general sense, whereas combined analysis is able to account for specific deregulated genes in different species that can later be studied for deregulated functional annotations as well. Therefore, this study focuses on implementing a combined analysis algorithm using the GRF technique and describing the pseudocode concisely so the implementation is clear and allows readers to implement the code themselves. Another aim of this study would be to assess the sensitivity (how changing the parameters affects the algorithm) and scalability (how fast the algorithm is, how efficiently it can deal with large datasets) of the algorithm by modifying its parameters and measuring its runtime. A secondary aim of this study involves finding deregulated genes between two opposing biological conditions in two evolutionarily-similar species, where the results (set of deregulated genes found) will be evaluated by comparing them with previous knowledge of the biological conditions. Since the model constructed here would be very similar to that by Lu et al., the hypothesis is that this model could be applied to any biological conditions for two evolutionary-similar species to find conserved deregulated genes in both species, an assertion made by Lu et al.[38]. The model used for this study is the same as that shown in Figure 1.1(b).

Section 1.5 discusses biological background of ageing (the biological condition chosen to be studied), Chapter 2 gives an explanation of how the GRF algorithm works (including its complexity), Chapter 3 shows the results of the algorithm when run on synthetic data (scalability and sensitivity of the algorithm) and biological data (finding genes deregulated in ageing) and Chapter 4 gives the conclusion of the results and an evaluation of the performance of the algorithm.

1.5 Biological background of ageing

1.5.1 Ageing in general

Ageing is a progressive, irreversible process that can be divided into 3 stages: metabolism, damage and pathology of cells. To sustain life, metabolism takes place in different types of cells, and at the same time produces toxins, which accumulate in cells to generate toxin biological products. The toxins are stored in specific storage cellular organelles, where they accumulate up to a certain threshold when they shift the balance of metabolism. During the lifetime of an organism, mitochondria produces much ATP as cell energy and few reactive oxygen species (which are related to ageing pathology). When enough toxins accumulates, ATP-deficient mitochondria begin to accumulate, while more reactive oxygen species are produced. These changes trigger apoptosis (cell death) pathways, leading to cell death, failing of organs and finally organism death[46].

Since ageing in humans involves gene perturbations and large environmental variances, full experiments on the ageing mechanisms in humans are very difficult, which is why model organisms are studied. Model organisms, including *Caenorhabditis elegans* (*C. elegans*), *Drosophila* and mice, have genes orthologous to human genes, which allows to study ageing mechanisms in organisms with low environmental variation, and derive information about ageing mechanisms in humans[53]. Studying ageing-associated genes in model species, it is possible to find conserved genes and biological pathways in humans, as well as to further strengthen the evidence of human genes found to be ageing-associated[5]. Such studies have already shown that reduced activity of gene daf-2 (homologous of insulin growth factor receptors) causes slower ageing in *C. elegans*[33], mutant line Methuselah gene (homologous to G-protein coupled receptors) causes 35 percent increase in average life-span in *Drosophila*[34] and Cav-1 gene is an important control for healthy neuronal ageing in mice[25].

1.5.2 Ageing in cross-species studies

As mentioned in Section 1.5.1, studying ageing in model organisms allows to derive information about ageing mechanisms in humans. Furthermore, when studying ageing in multiple species simultaneously, there is additional support for the existence of conserved ageing mechanisms. One such study was done by McElwee et al., and focused on the insulin/insulin-like growth factor-like signalling (IIS) pathway in mutants of mice, flies and worms. Using expression meta-analysis, they searched and compared deregulated genes between datasets they generated themselves and publicly available ones, but they couldn't find any significant conservation at the gene level. However, using indirect comparisons, they identified several GO categories as evolutionarily conserved, including sugar catabolism, energy generation, glutathionine-S-transferases and other processes linked to cellular detoxification[40]. Another experiment conducted by McCarroll et al. studied ageing-deregulated genes in humans, yeast, fly and worm using expression meta-analysis as well, and found mitochondrial metabolism, DNA repair and cellular transport as evolutionarily conserved processes involved in ageing[39]. These studies therefore show some functional annotations that can be considered as ageing-related, and can be useful to evaluate the final results of this study.

Since ageing is a process involved in many biological mechanisms such as various signalling pathways, cell cycle and late-onset diseases, there are many ageing-associated genes to discover. Some such ageing-associated genes are consistent over different cell types, involved in mechanisms intrinsic to cells, while others are cell specific. It is important to mention that in the process of ageing, some cells die, other grow and yet other simply remain quiescent[13]. From a physiological perspective, ageing causes accumulation of damage in cells belonging to muscle tissue (skeletal muscles and heart), liver, kidney and brain, and so cell types of those organs/tissues are highly interesting in the study of ageing-deregulated genes. The studies above look at several microarrays from several species, but they do not consider the genetic difference in tissues as much. The study by Magalhães et al., on the other hand, studies exactly that, comparing microarrays for human, mice and rats in different tissues (such as skeletal muscles, lungs, kidney, heart, hippocampus, frontal cortex and eye), and showed the data was associated with GO categories of mitochondria, metabolism and apoptosis[13]. After a quick look at the datasets, it was decided to use brain datasets, due to the important role ageing plays in increasing susceptibility to certain mental disorders (such as Alzheimer's disease), as well as its effect on certain higher brain activities such as learning and memory. Section 3.3.1 elaborates on the brain datasets used for human and mouse.

1.5.3 Brain ageing

The brain is considered a highly crucial organ for many organisms, and is considered especially complex and intricate in humans. Humans are especially distinguished from other organisms due to large brains and cognitive and behavioural abilities that transcend those of other animals. In addition, humans have a disease profile that does not occur in other organisms, not even in primates, including vulnerability to neurodegenerative disorders (such as Alzheimer's disease), AIDS and certain epithelial cancers. Humans also have longer lifespan than other primates and commonly used model organisms (*C. elegans*, *Drosophila* and mice), which means humans have higher susceptibility to ageing-related diseases, such as certain neurodegenerative disorders[43]. Therefore, ageing-related neurodegenerative disorders can only be studied in humans. Nevertheless, for the sake of studying gene expressions in ageing brains, this is actually beneficial: the study of Nutrition, Ageing and Memory in the Elderly (NAME)[45] showed that dementia increases in humans with age, particularly after the age of 80, but even within the sample of individuals over 80, only 40% were diagnosed with dementia. Therefore, it is necessary to acknowledge that some deregulated genes in old individuals result from late-onset neurodegenerative disorders, and not by ageing itself. With this in mind, cross-species analysis of genes from brain cells allows to downplay the statistical significance of human genes deregulated in neurodegenerative disorder , since their homologs will probably not appear deregulated in other species.

Ageing in relation to the brain is mostly associated with brain atrophy, where neurons die out, losing the connectivity between them, and often resulting in smaller brain size. This decline can be seen in humans at a rate of about 5% per decade after the age of 40 years, and the rate increases over the age of 70 years, where the highest decline can be seen in the gray matter of the frontal cortex (higher mental functions involving moral judgement, social behaviour and predicting consequences of actions) and parietal cortex (involving integrating sensory information and visuospatial processing)[46].

Other regions affected by ageing are the hippocampus and prefrontal cortex, which take part (among other things) in memory, thus showing elderly people as having decline in spatial and episodic memory. Particular associations have also been drawn between age, reduction in prefrontal cortex volume and decrease in ability to perform executive functions (functions involving organising incoming stimuli data, processing it and planning response to it)[46].

On the gene and protein level, protein synthesis is crucial to maintain neural networks and electrical potentials for acquiring and storing memory. A particular set of genes known as immediate early genes (IEGs) are expressed to allow input and processing of data from the environment. Those include transcription factors, which control the regulation of other neuronal-related genes, and effector genes, such as neuronal activity regulated pentraxin (NARP), hypothesised to be related to processing sensory-specific information[30], and activity regulated cytoskeletal gene (*Arc*) necessary for maintenance of long-term memory[46]. When researching other genes related to ageing, it was found that some up-regulated genes in the brain (increasing with age) related to Ca^{2+} pathways contribute to formation of fibrillar Aβ protein in Alzheimer's disease, and some down-regulated genes related to energy metabolism contribute to memory deficiency[46]. Therefore, in analysing microarrays of brain cells while looking at ageing as the different biological conditions (young and old), IEGs and genes related to metabolism and Ca^{2+} pathways are of particular interest.

Chapter 2

Algorithm design

2.1 Algorithm elaboration

2.1.1 Algorithm overview

As mentioned in Section 1.4, the aim of this project is to implement the GRF combined analysis technique used by Lu et al.[38], and use it to find deregulated genes from microarray datasets of two evolutionary-related species. The algorithm represents maximum likelihood estimation (MLE) aimed to estimate certain parameters to maximise the overall likelihood of the GRF, where the algorithm incorporates belief propagation as it is faster and more efficient than standard MLE. The input portion of the algorithm consists of gene expression scores calculated from certain microarrays, some class association for each gene an a matrix with alignment scores between proteins related to those genes. The processing step involves first the initialisation of node potential functions and global distribution parameters (see Section 2.1.4), and then recalculation of those variables using belief propagation nested loops that run until the variables converge. In the processing step, the inner loop calculates messages from each node to each neighbouring node and converge when the messages do not change largely between iterations, and the outer loop calculates beliefs for each gene and the global distribution parameters until the global distribution parameters converge (see Section 2.1.5). This algorithm finally maximises a likelihood function and obtains for each gene a posterior probability to indicate which class it most likely belongs to (see Section 2.1.6). The diagram of the algorithm can be seen in Figure 2.1.

Table 2.1 gives a list of supporting algorithms (such as calculating posterior probabilities and deriving normal distributions), and Table 2.2 shows the procedure used to derive the weight matrix for the genes. In addition, the algorithms at the end of this chapter give a full elaboration of the pseudocode, where Algorithm 1 shows how the GRF is initialised, Algorithm 2 shows the belief propagation, Algorithm 3 shows how the global distribution scores are updated, and Algorithm 4 shows how the algorithms are combined and run until the global distribution scores converge.

2.1.2 Input

Microarrays allow to study multiple gene expressions at the same time in multiple samples, where certain genes are deregulated under different biological conditions. As such, deregulated genes will have a larger expression ratio between samples with different conditions than non-deregulated genes. This fact allows to derive the expression score s_i for each gene, which represents the expression

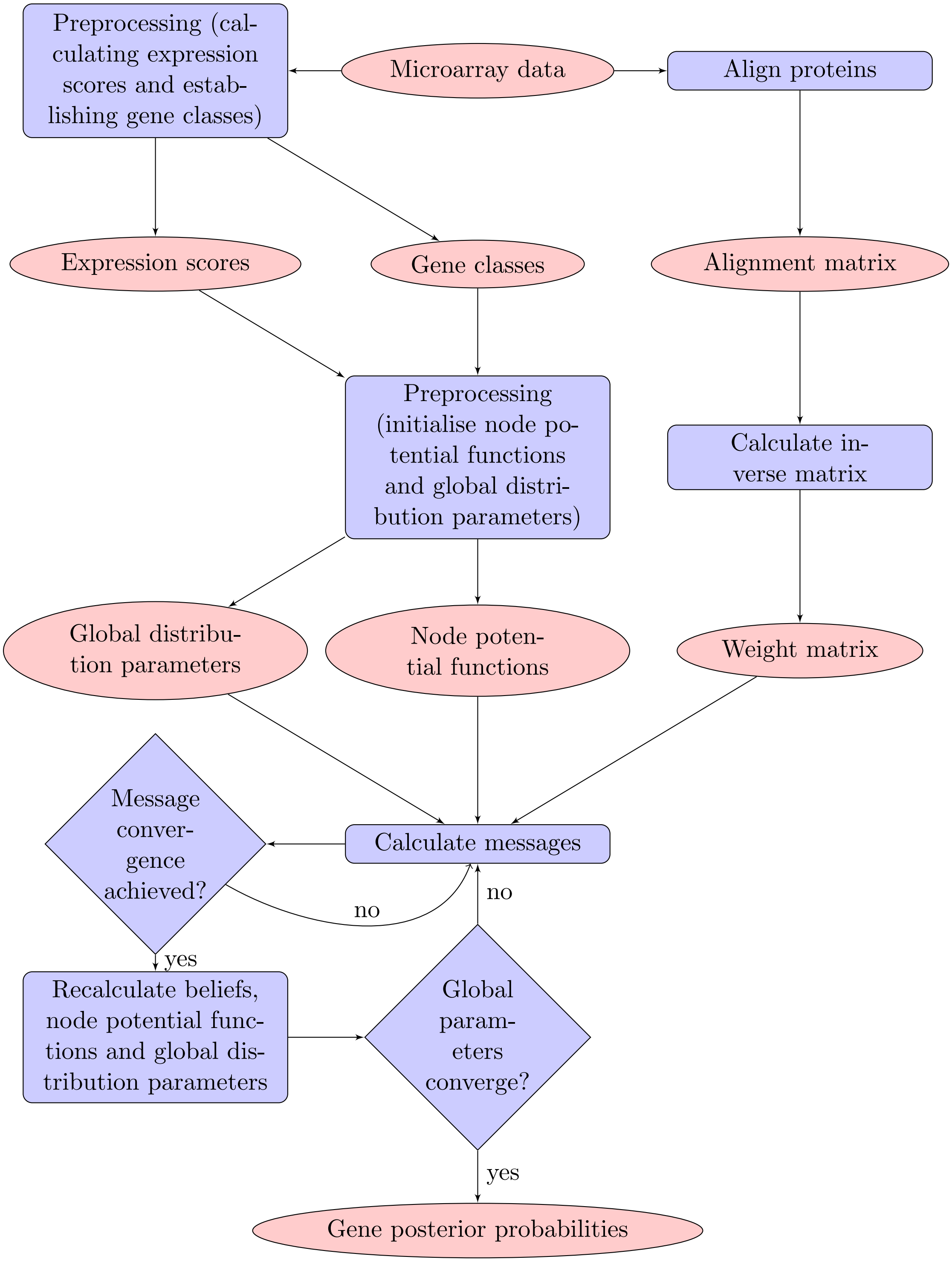

Preprocessing (calculating expression scores and establishing gene classes)
Microarray data
Align proteins
Expression scores
Gene classes
Alignment matrix
Preprocessing (initialise node potential functions and global distribution parameters)
Calculate inverse matrix
Global distribution parameters
Node potential functions
Weight matrix
Message convergence achieved?
Calculate messages
no
no
yes
Recalculate beliefs, node potential functions and global distribution parameters
Global parameters converge?
yes
Gene posterior probabilities

Figure 2.1: Gaussian random field (GRF) algorithm diagram
The diagram shows data in ellipses, calculations in rectangles and convergence checkpoints in diamonds.

profile of gene i in all samples. In addition, in order to distinguish between the deregulated and non-deregulated genes, it is necessary to divide them into classes, so each gene has a label indicating it belongs to one of the classes. The initialisation of the labels depends on the biological condition and previous knowledge about it: for example, when studying the biological condition of aging, it is necessary to label genes that are known to be aging-related as belonging to class 1, while the rest belong to class 0.

Each gene is related to a protein, and each protein might have similar sequences with other proteins, be it within the same species (paralogs) or between different species (orthologs). It is important to mention that genes/proteins with highly similar sequences might have similar expressions in organisms, and possibly similar functions. Therefore, to account for the similarity between genes, the second input for the algorithm is a weight matrix of scores between homologous proteins (see Section 2.1.3 for an elaboration on the matrix calculation).

The final input to the algorithm is a positive hyperparameter λ used for calculating messages between gene nodes (see Algorithm 2).

2.1.3 Computing the weight matrix

As asserted by Zhu et al.[55], the weight matrix of the GRF is equal to the inverse of a marginal covariance matrix, where the marginal dependency between genes is captured by alignment scores between their equivalent proteins. As such, by computing the alignment score matrix between proteins, it is possible to calculate its inverse to find the weight matrix for the equivalent genes in the GRF model.

The first step involves using the annotation packages in R for both species that are studied, in order to retrieve all the unique Entrez gene IDs. Those gene IDs are then mapped to Uniprot IDs using the official Uniprot website, and their sequences are retrieved directly from the NCBI (National Centre for Biotechnology Information) using a Matlab command that matches the Uniprot ID with its sequence. Given all the protein sequences, they are aligned with each other using pairwise local sequence alignment to generate an alignment score matrix, whereas a score below a certain threshold is replaced with 0 as indication for no homology between the proteins in question. For alignment score matrices that are not too large, simple matrix inversion can be done in either Matlab or R. However, if the matrix is too large (comparing tens of thousands of genes to each other), it is converted into diagonal block matrices by Markov clustering algorithm[17]. The diagonal block matrices, which are small clusters of the original matrix, are inverted using the Sparse Approximate Inverse Preconditioner[21], a computationally cheap approach to find matrix inverses. The inverse matrices for all blocks are then combined, and are used as the weight matrix for the GRF. The overview of this procedure can be seen in Table 2.2 and it is further elaborated in Section 3.2.1.

2.1.4 Initialisation

Computing node potentials

The node potential indicates, based on the gene expression data, whether a gene is deregulated between differing biological conditions (e.g. young and old) or not. To begin with, each gene node in the GRF model has a label c_i, and in this study the labels used are 0 and 1. The initialisation of the labels is elaborated in Sections 3.2.2 and 3.3.3 for the synthetic and biological data, respectively. Given these initial labels for all genes and their expression scores, the naive Bayes algorithm derives global distribution parameters (mean and standard deviation) for each species and class combination (there are 2 species and 2 classes, so 8 parameters are calculated). Given the distribution parameters and expression scores, the prior probabilities for each gene to be in either class, given the expression score, is calculated. Combined with the class probabilities, the posterior probability p_i for each gene to be in class 1 is calculated (see Algorithm 1). The posterior probabilities are then transformed using the probit function (inverse cumulative distribution function), and those probit transformations are used to construct probability density functions (PDFs) used as the node potentials in the GRF:

$$\psi_i(Y_i) = \phi(Y_i|\mu = \Phi^{-1}(p_i), \sigma^2 = 1)$$

where $\psi_i(Y_i)$ is the PDF of the normally distributed random variable Y_i and $\Phi^{-1}(p_i)$ is the probit function of the posterior probability p_i when Y_i is considered in isolation. It is important to mention that the posterior probabilities may be 0 or 1 if the prior probability of class 1 or class 0 are very small, respectively, and when these values are transformed with the probit function, it would generate node potential functions with means of $-\infty$ or ∞, respectively. Since the means of the node potential functions cannot be infinity or negative infinity (the algorithm cannot work with those values), the posterior probabilities are adapted so they cannot take values of 0 or 1. Any posterior probability below ϵ is replaced with ϵ, and any posterior probability over $1 - \epsilon$ is replaced with $1 - \epsilon$.

The relation between Y_i and s_i allows to calculate the random variable probability of being in a certain range instead of using naive Bayes for the class and expression scores:

$$\Pr(c_i = 1|s_i) = \Pr(Y_i > 1) \quad \Pr(c_i = 0|s_i) = \Pr(Y_i \leq 1)$$

Theoretically, the node potential functions are calculated for random variables. Practically, the construction of random variables can be circumvented to save computation time, and this is done by representing each node potential function as a mean and variance from the random variable PDF. The following sections in this chapter will display the algorithm techniques from both theoretical (random variables) and practical perspectives (means and variances).

2.1.5 Processing

Inference by belief propagation

Given the node potentials for all genes and their similarities (based on the weight matrix), it is possible to construct messages between genes, such that each gene i neighbouring gene j can send a message to gene j regarding what it "believes" its distribution is. Those messages are constructed between each two neighbouring genes in the GRF, and is based on a marginal distribution for all messages coming to the receiver from its neighbourhood:

$$m_{ij}(Y_j) \leftarrow \int \psi_{ij}(Y_i, Y_j)\psi_i(Y_i) \prod_{k \in N_G(i) \setminus j} m_{ki}(Y_i) \cdot dY_i$$

where $N_G(i)$ is the neighbourhood of node i on the GRF graph and $\psi_{ij}(Y_i, Y_j)$ is the edge potential function calculated as

$$\psi_{ij}(Y_i, Y_j) = \begin{cases} \exp\{-\lambda|W_{ij}|(Y_i - Y_j)^2\} & \text{if } W_{ij} \geq 0 \\ \exp\{-\lambda|W_{ij}|(Y_i + Y_j)^2\} & \text{if } W_{ij} < 0 \end{cases}$$

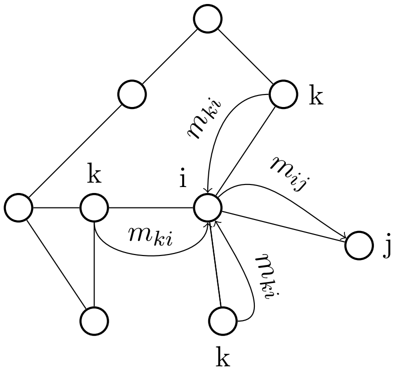


Figure 2.2: Gaussian random field (GRF) message passing
The message passed from gene i to gene j (m_{ij}) is based on the node potential of gene i and the messages from neighbour nodes k to i (m_{ki}).

where W_ij is the weight matrix entry for genes i and j. The importance of the edge potential function is that it reflects on the effect of different λ values on the algorithm, such that a large λ value would indicate a low edge potential function, and a small λ value would indicate a high edge potential function. The edge potential function reflects on the similarity between genes, such that a large λ value would mean only genes with large alignment scores are emphasised, making the algorithm stricter, and when λ is small, even genes with low alignment scores are emphasised, and the algorithm becomes more lenient. This is particularly important for the study of λ values in Section 3.2.2.

The integration operation above is computationally heavy, and it is possible to avoid it using direct calculations of means and variances of messages (See Algorithm 2). The messages from each gene to each other gene are calculated iteratively, and are tested for convergence using Euclidian distances between messages of a certain iteration and messages of a previous iteration. It is important to note that the messages from node i to each neighbouring node j depend on the messages going from the other neighbouring nodes of node i into it, as shown in Figure 2.2.

By marginalising bivariate Gaussian distributions, it is possible to derive mean v and variance ρ^2 for further calculations:

$$f_{ij}(Y_i) = \psi_i(Y_i) * \prod_{k \in N_G(i) \setminus j} m_{ki}(Y_i) \sim N(v_{ij}, \rho_{ij}^2)$$

This can be further simplified using the general rule for deriving means and variances from a product of Gaussian distributions:

$$\mu = \frac{\sum_i (\mu_i / \sigma_i^2)}{1/\sigma_i^2} \tag{2.1a}$$

$$\sigma^2 = (\sum_i (1/\sigma_i^2))^{-1} \tag{2.1b}$$

This is followed by further calculations (see Algorithm 2), which then derive the mean and variance of all messages.

Once the messages converge, the belief of each random variable Y_i is updated:

$$b_i(Y_i) \leftarrow (1/z_i)\psi_i(Y_i) \prod_{k \in N_G(i)} m_{ki}(Y_i)$$

Again, using Equations 2.1, it is possible to calculate the mean and variance directly, where this time the shortcut also circumvents the necessity to normalise the belief using normalising factor z, and uses the means and variances of beliefs as they are. Finally, it is important to state that the belief of each gene is not the node potential function, but a random variable that is used to derive the new posterior probability for each gene.

Updating the score distribution

Given the belief $b_i(Y_i)$ of each gene i (where in this study, a gene is either ageing-related or not), it is possible to calculate the posterior probability for that gene to be in class 1 or in class 0:

$$p_i^{(t)} = \Pr(C_i = 1|\Theta^{(t)}) = \int_0^{+\infty} b_i(Y_i)\,\mathrm{d}Y_i \quad q_i^{(t)} = \Pr(C_i = 0|\Theta^{(t)}) = 1 - p_i^{(t)}$$

The posterior probabilities for genes to belong to class 1 is first used to recalculate $\psi_i(Y_i)$ by using the probit of the new posterior probabilities, such that $\psi_i(Y_i) = N(\mathrm{probit}(p_i^{(t)}), 1)$, where t is the outer loop iteration. The posterior probabilities for genes to be in either class are then used to calculate the global distribution parameters:

$$\begin{aligned} \mu_0^{(t+1)} &= \Sigma_i q_i^{(t)} s_i / \Sigma_i q_i^{(t)} & \mu_1^{(t+1)} &= \Sigma_i p_i^{(t)} s_i / \Sigma_i p_i^{(t)} \\ \sigma_0^{2(t+1)} &= \frac{\Sigma_i q_i^{(t)} (s_i - \mu_0^{(t+1)})^2}{\Sigma_i q_i^{(t)}} & \sigma_1^{2(t+1)} &= \frac{\Sigma_i p_i^{(t)} (s_i - \mu_1^{(t+1)})^2}{\Sigma_i p_i^{(t)}} \end{aligned}$$

2.1.6 Outcome

The aim of the algorithm is to maximise a likelihood function L, and it does so by calculating the global distribution parameters until they converge. The likelihood can be expressed as the following function:

$$L = \frac{1}{Z} \prod_i \psi_i(Y_i) \prod_i \psi_{ij}(Y_i, Y_j)$$

where Z is the normalising term. Although the likelihood function is maximised, it is not explicitly calculated, and the output obtained from the algorithm is a list of posterior probabilities for all genes and global distribution parameters for each species and each class. The probabilities are used to predict the class each gene belongs to, and the classification can be compared with the actual gene classes to see the error rate of the algorithm.

2.1.7 Algorithm complexity

The GRF algorithm constitutes of two main convergence loops: an inner message convergence loop, which converges when the Euclidian distance of the messages is below ϵ, and an outer overall convergence loop, which converges when the Euclidian distance of the global distribution parameters is below ϵ. The inner loop calculates the messages from node i to every neighbour j, so there is a

Method	**Explanation**
naiveBayes (class,predictor_vars)	computes the conditional a-posterior probabilities of a categorical class variable given independent predictor variables using the Bayes rule
probit(Y)	computes the probit transformation of probabilities in Y
dnorm(X, μ, σ^2)	density of values in X within normal distribution $N(\mu, \sigma^2)$
pnorm(n, μ, σ^2)	area under the curve for the lower tail of a normal distribution $N(\mu, \sigma^2)$ from $-\infty$ to n; this can be expressed as a cummulative distribution function (CDF), where $\Pr(X \leq x) = CDF_{\mu,\sigma^2}(x)$

Table 2.1: List of supporting functions in R-base

Input	**Process**	**Tool/source used**	**Output**
Microarray probe IDs from two species (e.g. mouse and human)	Retrieve Entrez gene IDs	Annotation packages in R	Unique genes from both species
Unique genes from both species	Map Entrez genes to protein IDs	Website *www.uniprot.org*	Unique proteins from both species
Unique proteins	Retrieve protein sequences	Matlab, getgenpept command	tab-delimited file with protein sequnces
Protein sequences	Pairwise local sequence alignment for all sequences against each other	Alignment command and scoring system 2.1.3	Matrix with alignment scores for all sequences
Matrix with alignment scores	Calculate an inverse matrix	see Section 2.1.3 for elaboration	Inverse alignment score matrix (weight matrix for GRF)

Table 2.2: Procedure to create GRF weight matrix

Algorithm 1 Initialise GRF

```
**Input:** matrix containing Entrez gene IDs, and for each gene ID its equivalent Uniprot protein IDs, score expressions, binary class association (e.g. non-aging-related or aging-related) and binary organism association (e.g. human or mouse)

    Extract score expressions, class and organism associations from input matrix
    $\mathbf{s} \leftarrow$ score expressions
    $\mathbf{c} \leftarrow$ class assiciation
    $\mathbf{o} \leftarrow$ organism association
    Retrieve global distribution parameters for organism 0 using Naive Bayes model, where $[\mathbf{c}_0, \mathbf{s}_0]$ are subvectors of $[\mathbf{c},\mathbf{s}]$ $\forall o_i = 0$
    $params_m = [\mu_{0m}, \sigma_{0m}, \mu_{1m}, \sigma_{1m}]=$ naiveBayes($\mathbf{c}_0, \mathbf{s}_0$) (See Table 2.1)
    Retrieve global distribution parameters for organism 1 using Naive Bayes model, where $[\mathbf{c}_1, \mathbf{s}_1]$ are subvectors of $[\mathbf{c},\mathbf{s}]$ $\forall o_i = 1$
    $params_h = [\mu_{0h}, \sigma_{0h}, \mu_{1h}, \sigma_{1h}]=$ naiveBayes($\mathbf{c}_1, \mathbf{s}_1$) (See Table 2.1)
    Derive class probabilities $\Pr(c_i = 0)$ and $\Pr(c_i = 1)$ for both organisms from the respective naiveBayes models
    Derive prior probabilities $\Pr(s_i|c_i = 0)$ and $\Pr(s_i|c_i = 1)$ for both organisms from the respective naiveBayes models
    **for** $i \in \mathbf{c}$ **do**
        $\Pr(s_i|c_i = 0) = \mathrm{dnorm}(s_i, \mu_{0h}, \sigma^2_{0h}) \oplus \mathrm{dnorm}(s_i, \mu_{0m}, \sigma^2_{0m})$(See Table 2.1)
        $\Pr(s_i|c_i = 1) = \mathrm{dnorm}(s_i, \mu_{1h}, \sigma^2_{1h}) \oplus \mathrm{dnorm}(s_i, \mu_{1m}, \sigma^2_{1m})$(See Table 2.1)
        $p_i = \Pr(c_i = 1|s_i) = \frac{\Pr(s_i|c_i=1)\Pr(c_i=1)}{\Pr(s_i|c_i=1)\Pr(c_i=1)+\Pr(s_i|c_i=0)\Pr(c_i=0)}$
            Adapt posterior probabilities when they are equal to 0 or 1 (see Section 2.1.4)
    $\mu$=probit($\mathbf{p}$)
    **for** i=1 **to** length($\mathbf{p}$) **do**
        $\psi_i(Y_i)$ is a probability density function for random variable $Y_i$
        $\psi_i(Y_i) \sim N(\mu_i, 1)$

**Output:** $params_h$, $params_m$,$\psi(\mathbf{Y})$
```

Algorithm 2 Belief propagation

```
    Input:
      (a) $\psi(\mathbf{Y})$: node potential function for random variables (See Algorithm 1)
      (b) $\mathbf{W}$: weight matrix
      (c) $\lambda$: positive hyperparameter
    t and g are counters for convergence of outer loop and inner loop, respectively
    Messages are initialised as normal distributions N(0,1), but are not used for the first inner loop, i.e. when t=1 and g=1
(1) while $\sqrt{\sum_{j \in N_G(i)} (\mathrm{E}(m_{ij}) - \mathrm{E}(old\_m_{ij}))^2} > \epsilon \quad \forall i \in m$ do
        Obtain all messages for node $Y_i$ from its neighbours
        for $i = 1$ to length($\psi(\mathbf{Y})$) do
            Initialise old messages as current messages
        (2) $old\_m_{ij} = m_{ij}$
            Calculate the message distribution for each neighbour of $Y_i$
        (3) for j $\in N_G(i)$ do
                if g=1 & t=1 do
                In very first iteration, messages are initialised with $\psi(\mu, \sigma^2)$
            (a) $v_{ij} = \mathrm{E}(\psi_i(Y_i))$
            (b) $\rho_{ij}^2 = \mathrm{Var}(\psi_i(Y_i))$
                else
            (a) $v_{ij} = \frac{\sum_{k \in N_G(i) \backslash j} (\mathrm{E}(m_{ki})/\mathrm{Var}(m_{ki})) + \mathrm{E}(\psi_i(Y_i))/\mathrm{Var}(\psi_i(Y_i))}{\sum_{k \in N_G(i) \backslash j} (1/\mathrm{Var}(m_{ki})) + 1/\mathrm{Var}(\psi_i(Y_i))}$
            (b) $\rho_{ij}^2 = 1/(\sum_{k \in N_G(i) \backslash j} (1/\mathrm{Var}(m_{ki})) + 1/\mathrm{Var}(\psi_i(Y_i)))$
                Derive mean and variance scores for messages from gene $i$ to gene $j$
            (d) $\alpha_{ij} = 2\lambda |W_{ij}^*|$, where $W_{ij}$ is the weight matrix entry for nodes $i$ and $j$
            (e) $r_{ij} = 1/\rho_{ij}^2$
            (f) $\hat{\mu}_j = sign(W_{ij}^*) v_{ij}$
            (g) $\hat{\sigma}_j^2 = \frac{1}{r_{ij}} + \frac{1}{\alpha_{ij}}$
                Update messages from node $i$ to each node $j$ in the GRF, where $\hat{\mu}_j$ is the mean of the message and $\hat{\sigma}_j^2$ is the variance of the message
            (i) $m_{ij}(Y_i) = [\hat{\mu}_j, \hat{\sigma}_j^2]$
    Update belief of each node
(4) for $i = 1$ to length($\psi(\mathbf{Y})$) do
      (a) $b_i(Y_i(\mu)) = \frac{\sum_{k \in N_G(i)} (\mathrm{E}(m_{ki})/\mathrm{Var}(m_{ki})) + \mathrm{E}(\psi_i(Y_i))/\mathrm{Var}(\psi_i(Y_i))}{\sum_{k \in N_G(i)} (1/\mathrm{Var}(m_{ki})) + 1/\mathrm{Var}(\psi_i(Y_i))}$
      (b) $b_i(Y_i(\sigma^2)) = 1/(\sum_{k \in N_G(i)} (1/\mathrm{Var}(m_{ki})) + 1/\mathrm{Var}(\psi_i(Y_i)))$
    Output $\mathbf{b}(\mathbf{Y})$: belief distribution for each node $i$
```

Algorithm 3 Update global distribution parameters

```
Input:
  (a) $params_h$ and $params_m$: global distribution parameters in both species in different groups
  (b) $\mathbf{b}(\mathbf{Y})$: belief distribution, derived from Algorithm 2
Update score distributions
  (1) for i=1 to length($\mathbf{b}(\mathbf{Y})$) do
          Calculate the probability of the label of node $i$ being 0 or 1 given parameters at time $t$
      (a) $p_i(t) = \Pr(c_i = 1|params(t)) = \int_0^{+\infty} b_i(Y_i)\,\mathrm{d}Y_i$=1-pnorm(0,E($b_i(Y_i)$),Var($b_i(Y_i)$))
          Adapt posterior probabilities when they are equal to 0 or 1 (see Section 2.1.4)
      (b) $q_i(t) = 1 - p_i(t)$
      The posterior probability that any gene is in class 1 is used to recalculate the node potential functions
  (2) $\psi_i(Y_i)(t) = [\mathrm{probit}(p_i(t)), 1]$
      Parameters are stored for all iterations of the outer loop
  (3) Calculate new parameters
      $\mu_{0m} = \Sigma_i q_i s_i / \Sigma_i q_i$   where $i \in o_i = 0$   AND   $c_i = 0$
      $\sigma^2_{0m} = \sqrt{\frac{\Sigma_i q_i (s_i - \mu_{0m})^2}{\Sigma_i q_i}}$   where $i \in o_i = 0$   AND   $c_i = 0$
      $\mu_{1m} = \Sigma_i p_i s_i / \Sigma_i p_i$   where $i \in o_i = 0$   AND   $c_i = 1$
      $\sigma^2_{1m} = \sqrt{\frac{\Sigma_i p_i (s_i - \mu_{1m})^2}{\Sigma_i p_i}}$   where $i \in o_i = 0$   AND   $c_i = 1$
      $\mu_{0h} = \Sigma_i q_i s_i / \Sigma_i q_i$   where $i \in o_i = 1$   AND   $c_i = 0$
      $\sigma^2_{0h} = \sqrt{\frac{\Sigma_i q_i (s_i - \mu_{0m})^2}{\Sigma_i q_i}}$   where $i \in o_i = 1$   AND   $c_i = 0$
      $\mu_{1h} = \Sigma_i p_i s_i / \Sigma_i p_i$   where $i \in o_i = 1$   AND   $c_i = 1$
      $\sigma^2_{1h} = \sqrt{\frac{\Sigma_i p_i (s_i - \mu_{1m})^2}{\Sigma_i p_i}}$   where $i \in o_i = 1$   AND   $c_i = 1$
Output $params_h(t)$ and $params_m(t)$: updated global distribution parameters for outer iteration $t$
```

loop for each gene i and a nested loop for each of its neighbours j. If there are n genes in the data, the loop for neighbours j (which contains basic operations) is at most of time complexity $O(n-1)$ (depending on threshold for alignment scores, not every node may be interconnected with every other node), and the loop for the genes i is therefore of time complexity $O(n(n-1))$, which means it has a quadratic time complexity $O(n^2)$. However, the convergence complexities for the inner and outer loop are difficult to calculate, as they depends on multiple factors (such as convergence threshold and λ values). Since it is yet unclear which factors affect the convergence complexities, the complexity can be written as $O(N_{outer}, N_{inner}, n^2)$, where N_{outer} is the complexity of the outer loop, and N_{inner} is the convergence of the inner loop. The algorithm complexity is further studied and expanded on in Section 3.2.

Algorithm 4 GRF convergence algorithm

```
    Input:
      (a) matrix containing Entrez gene IDs, and for each gene ID its equivalent Uniprot protein IDs, score expressions, binary class association and binary organism association
      (b) weight matrix $W$
      (c) positive hyperparameter $\lambda$
(1) Initialise node potential functions and global distribution parameters for both species (See Algorithm 1)
    $params = [params_m, params_h]$
    while $\sqrt{\sum(params - old\_params)^2} > \epsilon$ do    {Main Loop}
      (2) Use belief propagation to derive belief for each node (See Algorithm 2)
      (3) Update global distribution parameters and node potential functions (See Algorithm 3)
(4) From updated node potential functions, derive inverse probit functions for the means to get ageing posterior distributions
    Output $p$ (ageing posterior probabilities for each gene) and $params$ (final global distribution parameters)
```

Chapter 3

Results

In the GRF algorithm, there are three important inputs: the expression scores of genes in different samples, their association with a certain biological condition and the weight matrix indicating the similarity score between proteins synthesised from the genes. The algorithm was run on synthetic data in order to study its sensitivity and scalability, and it was run on biological data in order to obtain some biological results regarding brain ageing (see Section 1.5.3). Since the synthetic data is used to study the algorithm performance, the data must be small and easy to handle, so 20 homologous genes were chosen from human and mouse (see Section 3.2). The biological data will be using complete microarrays from human and mouse, resulting in tens of thousands of genes (see Section 3.3.1).

3.1 Hardware and software architecture

All computations were run using Intel(R) Core(TM)2 Duo CPU model E7400 with 2.80 GHz, 2 GB RAM and Operating system of Windows 7 Ultimate 2009. The software used were R 2.13, Matlab R2011a and Microsoft Excel 2007.

3.2 Synthetic data

3.2.1 Weight matrix selection

Since the synthetic data is small, it is used to study different possible alignment techniques and their runtime. Choosing which technique to use is particularly critical for the biological data, as the scores should be obtained quickly and efficiently for a large dataset. In sequence alignment, 3 options must be considered:

(1) Pairwise or multiple alignment - pairwise alignment compares two sequences directly with each other, giving a specific alignment score, whereas multiple sequence alignment searches for conserved sequence regions in a group of sequences that are evolutionarily related

(2) Local or global alignment - local alignment is used for dissimilar sequences with different lengths, but with similar sequence motifs, whereas global alignment compares sequences that are roughly of equal size

(3) Substitution matrix - a generally agreed-upon matrix accounting for the alignment scores between specific amino acids within protein sequences

Since the microarrays contain many various genes, it is feasible to assume they differ largely in length (preferable local rather than global alignment), and most likely do not have overall conserved sequence regions (preferable pairwise rather than multiple alignment). Two types of local pairwise alignment techniques that are often used are the Smith-Waterman algorithm and BLAST (Basic Local Alignment Search Tool) algorithm, where BLAST is considered faster than Smith-Waterman because BLAST finds the approximate optimum score, while Smith-Waterman finds the actual optimum. Both algorithms were compared to choose which one is better in terms of results and runtime. For the substitution matrix, several options are available, and those are discussed in depth in Section 3.2.1.

Smith-Waterman algorithm

Smith-Waterman (SW) algorithm is based on dynamic programming (DP), which is a technique used to maximise the similarity between two sequences using scores for matches, mismatches and gaps. DP relies on a recursive definition of the optimal score, a DP matrix that remembers multiple optimal scores for subproblems, a bottom-up approach that solves the subproblems from smallest to largest, and a traceback that recovers the optimal solution from the matrix. Unlike DP, though, SW focuses on the optimal local score without considering other subproblems.

Given two sequences (DNA, RNA or protein) x and y of lengths M and N respectively, the SW algorithm initialises a DP matrix D such that $D_{i0} = D_{0j} = 0$ for $0 \leq i \leq M$ and $0 \leq j \leq N$, meaning that the first row and first column of the DP matrix are equal to 0. For each following matrix elements in position (i, j), any of 3 possible cases may occur: (1) x_i is aligned with y_j, (2) x_i is aligned with a gap, or (3) y_j is aligned with a gap. The optimal alignment would be the highest score from these cases, unless all 3 cases return a negative score, in which case the score is 0. Avoiding negative score creates a situation where the algorithm focuses on finding similarities between sequences and avoids considering dissimilarities. The matrix score $S(i, j)$ is a score based on all previous scores, and can be expressed mathematically as follows:

$$S(i,j) = max \begin{cases} S(i-1, j-1) + \sigma(x_i, y_j) \\ S(i-1, j) - \gamma \\ S(i, j-1) - \gamma \\ 0 \end{cases}$$

where $\sigma(x_i, y_j)$ is the alignment score between the character x_i and y_j and γ is a positive number representing a gap penalty. Once all scores are obtained, the algorithm finds the highest score in matrix D, and finds the optimal alignment by traceback of the sequence until it reaches a score of 0. The SW algorithm is also able to find a local optimal alignment that is not necessarily as long as the sequences, deeming it as a local alignment technique[48, 15].

The time complexity of this algorithm can be calculated by looking at its 3 main steps: (1) initialisation, (2) matrix filling, and (3) traceback. The initialisation includes simple operations of filling the first and column of the matrix with 0, so it has a time complexity of $O(M + N)$. The matrix filling (calculating scores for each position in the matrix) is also based on simple operations (finding max value from 4 values), so filling an $M * N$ matrix has a time complexity of $O(MN)$.

Finally, traceback of the optimal score involves a reverse version of the matrix filling, with a time complexity not higher than $O(MN)$ (can be lower if the optimal alignment is local). As such, the total complexity of this algorithm is $O(M+N)+O(MN)+O(MN)=O(MN)$.

BLAST 2 sequences

The BLAST 2 sequences (bl2seq) algorithm is a pairwise local alignment that is based on the BLAST algorithm, which is similar to SW algorithm. Nevertheless, BLAST is a heuristic algorithm, meaning it runs faster and generates approximate results. To speed up BLAST compared to SW algorithm, BLAST contains a pre-alignment step that generates a database of one sequence aligned with several smaller sub-sequences, and then compares them with another sequence. Given a protein sequence x, the positions of all possible words within it of length w (3 by default settings) are generated in a hash table (speeds up the algorithm), and each word is aligned with a database containing all possible word combinations of proteins of length w (i.e. 20^w words). Using a certain statistical significance threshold, all statistically significant word matches, also known as High-scoring Sequence Pairs (HSPs), are kept. Each HSP is then extended to the left and right until its alignment with the segment of another sequence y generates a score not below threshold X. This generates a set of sub-sequences in y that align with sub-sequences in x, which rather than studying the optimal alignment alone, allows biologists to observe the multiple motifs that are similar in both sequences[49]. For an elaboration on the complexity of BLAST, see [6].

Substitution matrix selection

Substitution matrices reflect on how likely it is that 2 residues align with each other based on their frequency of appearance apart and together. The substitution matrix scores often rely on log-odds score: given 2 hypotheses (null hypothesis "the residues aligned are uncorrelated" and alternative hypothesis "the residues aligned are correlated"), the log-odds score is the logarithm of the ratio of the likelihoods of both hypotheses. The log-odds score can be represented as such:

$$\mathrm{s}(a,b) = 1/\alpha \log \frac{P_{ab}}{f_a f_b}$$

where α is a scaling factor, P_{ab} is the likelihood (probability) that residues a and b are observed as aligned in homologous sequence alignment (alternative hypothesis), and f_a and f_b the frequencies that amino acids a and b are overall observed on average in any protein sequence (null hypothesis). Therefore, if there is a higher probability that residues a and b are aligned together in homologous sequences than by chance, than $P_{ab} > f_a f_b$, so the log-odds score is positive[16].

There are two commonly used substitution matrices known as PAM (Point Accepted Mutation) and BLOSUM (BLOck SUbstitution Matrix), which use different techniques to calculate the log-odds score presented in the substitution matrices. Nevertheless, they both rely on the same principle in sample data selection, which dictates that the evolutionary distance should be small so that amino acid frequencies can be studied in closely aligned homologous proteins[11].

Point accepted mutations (PAMs) are protein mutations by single amino acids, where one amino acid is replaced by another amino acid. When these mutations occur in nature, it is preferable for proteins to change in small degrees, and so preferable PAMs occur between amino acids with similar chemical and physical properties in order to keep the structure and function of the mutated protein similar to those of the original protein. PAM matrices are used to calculate the frequencies of each amino acid changing into another amino acid within multiple proteins with a certain number of PAM units. For example, PAM250 gives scores of amino acids mutating to other amino acids in proteins which are 250 point accepted mutations apart. The first step to calculate a PAM250 matrix is by taking multiple proteins with 250 PAM units distance, and then calculate M, a matrix such that M_{ij} is the probability that amino acid in position i in one sequence is replaced by amino acid in position j in another sequence. Using probability matrix M, the PAM250 matrix is calculated as a log odds matrix of the substitution probability and frequency of the substituted amino acid. Therefore, entry of PAM250 at position (i, j) can be represented as follows:

$$\text{PAM250}(i, j) = \log_{10} \frac{M^{250}(i, j)}{f(i)}$$

where $f(i)$ is the probability that amino acid at position i occurs in the other sequence by chance. The commonly used PAM250 is multiplied by 10, such that if a score in the PAM250 matrix is 10, $\text{PAM250}(i, j) = 1$, and the substitution of the amino acids in positions i and j within similar proteins would occur 10 times more frequently than at random. The construction of the PAM250 matrix was done by Dayhoff and Schwartz using phylogenetic trees and related sequences[11].

BLOSUM calculates log odds matrix in a way similar to PAM, except that whereas a specific PAM matrix is based only on proteins with specific PAM units, BLOSUM creates blocks of sequences that are clustered if their alignment is above a certain percentage threshold. For example, BLOSUM62 uses a threshold of 62%, so if sequences A and B have 62% or more aligned positions, they would be clustered together. If another sequence C has 62% or more aligned positions with either A or B (not necessarily with both), it is added to the cluster of A and B. The advantage of such a technique is that it represents scores of relatively similar sequences, and not just of sequences with a fixed distance, as is the case in PAM matrices. The BLOSUM62 matrix was constructed by Henikoff and Henikoff using 504 groups of non-redundant proteins catalogued in Prosite and keyed in SWISS-PROT[26].

Alignment results

To keep the synthetic data small, it was decided to use 20 pairs of homologous genes from two evolutionary similar species, in this case a human and a mouse. The 20 pairs of homologous genes are shown in Table 3.1, along with their lengths. The use of proteins of different lengths shows how the alignment algorithms compare sequences of varying lengths, from very short to very long. Both algorithms include various parameters, including penalty costs for gaps and extensions, penalty for mismatches, word-sizes (for bl2seq) and score thresholds, and the default settings were used in most cases. The only exception was the expect value threshold for bl2seq, which is the probability of an alignment to be significant (default is E=10, i.e. 10 matches in the alignment are found by chance). The default returned several scores equal to 0, as their alignment resulted in an expect score higher than 10, so to retrieve as many scores as possible, bl2seq was run once with default settings and once with expect value threshold E=1000.

When generating alignment score matrices, it is possible to use complete alignment matrix (alignment of all proteins with each other, including each protein with itself) or a distance matrix (score between a protein sequence and itself is 0). This distinction has an effect on the diagonal of the score matrices, and thus on the inverses (weight matrices for the GRF), since the distance matrix

Protein name	**Uniprot ID (human)**	**Length human protein (amino acids)**	**Uniprot ID (mouse)**	**Length mouse protein (amino acids)**
Adra1a	P35348	466	Q8BV77	466
APEX1	P27695	318	Q544Z7	317
ALDOC	P09972	364	P05063	363
Ifi27l2a	Q9H2X8	130	Q8R412	90
SHC1	P29353	583	P98083	579
Spectrin	Q13813	2472	B2RXX6	2477
Dmbt1	Q9UGM3	2413	Q60997	2085
Bai3	O60242	1522	Q6ZQ96	612
BEX3	Q00994	111	Q9WTZ9	124
Septin 4	O43236	478	P28661	478

Table 3.1: Pairs of homologous genes used in synthetic data

Method	**Algorithm**	**Program**	**Runtime (seconds)**	**Average runtime (seconds/iterations)**
bl2seq	BLAST algorithm	BioPerl	28.5	0.136
swalign	SW algorithm	Matlab	3.5022	0.0167
pairwiseAlignment	SW algorithm	R	8.45	0.0402

Table 3.2: Average running time for pairwise alignment techniques on 20 genes

would have diagonal 0, but the complete alignment matrix would have alignment scores between each protein and itself depending on the length of the protein (the longer the protein, the higher the score). Nevertheless, the GRF algorithm represents a graph with nodes and edges, so since there is no edge between a node and itself, it makes no sense to calculate a score that represents such an edge, thus making the distance matrix preferable to the complete alignment matrix.

The alignment techniques used were bl2seq in StandAloneBlast module in BioPerl 2.1.8 run on ActivePerl 5.12.4.1205, pairwiseAlignment in Biostrings package 2.20.1 in R, and swalign in Bioinformatics toolbox in Matlab. Appendix A shows the heatmaps of the inverse matrices obtained for the different sequence alignment techniques, and Table 3.2 shows the total runtime and runtime per iteration for each technique comparing 20 proteins with each other (210 iterations). The heatmaps are very similar in terms of alignment scores, meaning that the choice of techniques and substitution matrices has little effect on the overall pattern of scores. However, the runtime is unexpectedly much faster for SW algorithm in both R and Matlab compared with BLAST in BioPerl. Since the alignment patterns are the same (even for different substitution matrices), and the SW algorithm in Matlab runs the fastest, the Matlab alignment technique with BLOSUM62 will be used to construct the weight matrices for synthetic and biological data. The choice of BLOSUM62 as the substitution matrix is motivated by the fact that it is known to be useful for different evolutionary distances[50].

3.2.2 Synthetic data - Algorithm execution

Data initiation

The algorithm was written and run in the R environment. As mentioned before, 20 homologous human and mouse genes were used, and each gene was randomly assigned to either class 0 or class 1. Overall, the genes were equally distributed into classes, such that 5 human genes were classified as 0, and 5 human genes were classified as 1 (same applied for mouse genes). For all genes in class 0, the score expressions were random numbers within the normal distribution $N(4, 1.5)$, and for all genes in class 1, the score expressions were random numbers within the normal distribution $N(6, 1.5)$. The reason for nearby distributions for class 0 and class 1 genes is that overlapping expression scores will test the efficiency of the algorithm at finding the correct classes for each gene. In addition, the message convergence threshold was set to ϵ, which in R is $2.220446 * 10^{-16}$, in order to make sure the convergence is as accurate as possible, and for the overall GRF the convergence threshold was set to 0.0005. After several trial runs for the algorithm on the synthetic data, it was found that the messages converge at a reasonable speed even for threshold $2.220446 * 10^{-16}$, but that the overall GRF underwent many fluctuations in the Euclidean distances and took several hours to run with such threshold $2.220446 * 10^{-16}$, which is why the overall GRF was given a convergence threshold of 0.0005 (sufficiently small without taking too much runtime). From this point on, any reference to ϵ would mean $2.220446 * 10^{-16}$.

Algorithm sensitivity

The main results that reflect the GRF algorithm performance are the node potential functions, the global distribution parameters and the runtime and number of iterations of the GRF. In addition, if the GRF algorithm is input with the synthetic data (expression scores and classes) and the weight matrix as they are, the only parameter that can be changed, and therefore might change the results, is λ. Therefore, the synthetic data was input to the GRF algorithm, while different λ values (100-1000 with steps of 100, and 1000-10000 with steps of 1000) were tested to see how changing it would affect the results. Looking at the global parameters for the different λ values in Table 3.3, it can be seen that the parameters fluctuate up to the $\lambda = 700$, where the parameters stay the same for larger λ values. Figures B.1, B.2 and B.3 show the unique distribution for each species under each class, showing how the parameters change for different λ values.

Table 3.4 shows the runtime and number of iterations for GRF run with the different λ values, as well as the runtime per iteration (approximately equal to the average rate at which messages converge). As can be seen, both runtime and number of iterations decrease quite consistently as λ increases, until runtime starts increasing again at $\lambda > 4000$ (this can also be seen in Figures B.4, B.5, B.6 and B.7). However, the number of iterations increases to a larger extent than the runtime, which means that the average runtime of message convergence is slower as λ increases. Therefore, when $\lambda < 4000$, the messages converge relatively quickly, but this causes the overall GRF to converge slower, while when $\lambda > 4000$, the messages converge slower and slower, but the GRF seems to converge in the same rate as it would for $\lambda = 4000$. Therefore, for the synthetic data, $\lambda = 4000$ generates the fastest converging GRF and messages, which means it will be used for other experiments that need to be executed quickly. The runtime and iteration plots were made in Excel, so they were tested for different trend-lines (linear, exponential, logarithmic and quadratic) and their R^2 values (explained variance of plotted lines) were measured. Table 3.5 shows the R^2 values for the different trends for iteration and runtime in different λ ranges (100-1000 and

λ	Mean non-ageing mouse	Mean ageing mouse	Mean non-ageing human	Mean ageing human	Variance non-ageing mouse	Variance ageing mouse	Variance non-ageing human	Variance ageing human
100	4.437759	5.348929	6.076354	5.491882	0.865905	0.634734	3.023487	1.378011
200	5.028699	4.940149	5.201572	6.219638	1.048593	0.800434	1.324214	1.796387
300	5.028699	4.940149	5.201572	6.219638	1.048593	0.800434	1.324214	1.796387
400	5.028699	4.94015	5.201573	6.21964	1.048594	0.800433	1.324211	1.796393
500	4.595457	5.243736	5.632065	5.55451	0.651986	0.941359	1.438893	2.511147
600	4.595457	5.243736	5.632064	5.554511	0.651986	0.941359	1.438894	2.511142
700	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
800	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
900	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
1000	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
2000	4.595514	5.24373	5.632065	5.554509	0.651983	0.941404	1.438891	2.511153
3000	4.595505	5.24373	5.632065	5.554509	0.651983	0.941397	1.438891	2.511153
4000	4.595479	5.243734	5.632065	5.554509	0.651985	0.941376	1.438891	2.511153
5000	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
6000	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
7000	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
8000	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
9000	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153
10000	4.595457	5.243736	5.632065	5.554509	0.651986	0.941359	1.438891	2.511153

Table 3.3: Global distribution parameters

λ	Runtime	Iterations	Runtime/iteration	Lambda	Runtime	Iterations	Runtime/iteration
100	2140.11	64	33.43921875	1000	2225.29	30	74.17633333
200	3320.53	81	40.99419753	2000	1648.26	17	96.95647059
300	2621.41	55	47.662	3000	1389.04	12	115.7533333
400	2185.72	42	52.04095238	4000	1277.21	10	127.721
500	3534.86	63	56.10888889	5000	1383.25	10	138.325
600	3142.67	52	60.43596154	6000	1280.5	9	142.2777778
700	2891.21	45	64.24911111	7000	1382.89	9	153.6544444
800	2711.25	40	67.78125	8000	1400.17	9	155.5744444
900	2561.71	36	71.15861111	9000	1438.25	9	159.8055556
1000	2225.29	30	74.17633333	10000	1454.65	9	161.6277778

Table 3.4: Runtime for GRF on synthetic data
Runtime/iteration gives the average runtime it takes messages to converge for a certain λ value

	R^2			
	λ=(100-1000)		λ=(1000-10000)	
	Runtime	Iterations	Runtime	Iterations
Linear	0.008	0.692	0.2866	0.5275
Exponential	0.0035	0.7393	0.2676	0.6309
Log	0.0066	0.6052	0.5814	0.8065
Quadratic	0.3164	0.6931	0.7913	0.8571

Table 3.5: R squared values for runtime and iteration trendlines for GRF

1000-10000), showing the highest pattern to be quadratic. It can therefore be concluded that both results and convergence speed of the algorithm are sensitive to λ, but when λ is sufficiently large, the results remain the same (for $\lambda \geq 700$), and the iterations remain approximately the same while the runtime gradually increases (for $\lambda \geq 6000$). The complexity can therefore be rewritten as $O(N_{outer}N_{inner}(\lambda)n^2)$, showing the convergence of the inner loop, and therefore the complexity of the algorithm, depends on λ.

Cross validation and error rates

To determine the extent of error rates generated by the algorithm, it was tested with a cross-validation (CV) leave-one-out (LOO) technique. In it, a gene was left out, as if its class was unknown, and the other genes from the same species were used to construct a Naive Bayes model and predict the class of the LOO gene based on its expression score. This was repeated for all genes, where the predicted class for some genes was the same as their original class (they are denoted here as "Other genes") and the rest of the genes had classes predicted differently from their original class. Using $\lambda = 4000$, which produces fast results consistent for multiple λ values, the CV-LOO was executed to find the error rates for leaving different genes out. Table 3.6 shows the error rates for all LOO genes, and the posterior probabilities for each gene to be ageing-related from the different CV-LOO tests can be seen in Table B.1.

Given those error rates, the average error rate was calculated to be 46.43%, meaning that 53.571% of the genes were correctly classified by the algorithm. Although the error rate is less

LOO gene	**Classification**	**Number misclassified genes**	**%Error**
Other genes	No changes	9	45
76933	$0 \rightarrow 1$	9	45
11792	$0 \rightarrow 1$	9	45
328	$0 \rightarrow 1$	9	45
6464	$0 \rightarrow 1$	9	45
20740	$1 \rightarrow 0$	11	55
6709	$1 \rightarrow 0$	9	45

Table 3.6: Error rates from cross-validation with $\lambda = 4000$
The column "LOO gene" shows the Entrez ID of the gene left out and the second column shows the original class and what it was predicted to be.

LOO gene	**Classification**	**Number misclassified genes**	**%Error**
Other genes	No changes	10	50
76933	$0 \rightarrow 1$	12	60
11792	$0 \rightarrow 1$	9	45
328	$0 \rightarrow 1$	10	50
6464	$0 \rightarrow 1$	9	45
20740	$1 \rightarrow 0$	10	50
6709	$1 \rightarrow 0$	8	40

Table 3.7: Error rates from cross-validation with $\lambda = 100$
The column "LOO gene" shows the Entrez ID of the gene left out and the second column shows the original class and what it was predicted to be.

than half, it is still quite high, which indicates that either the algorithm is not very good at correct classification of genes, or that the synthetic data was not well constructed to retrieve the correct gene classifications. The possible causes for this error rate are further discussed in Section 4.1.
In Section 3.2.2, it was shown that different λ values can return different distribution results, and so the CV-LOO was repeated with a different λ value of 100. Table 3.7 shows the error rates for all LOO genes, and the posterior probabilities for each gene to be ageing-related from the different CV-LOO tests can be seen in Table B.2. These executions show an error rate of 48.57%, which is even higher than before, indicating that in this case a higher λ value gives more accurate results.

Algorithm scalability

As seen in Table 3.4, generally an increase in λ speeds up the convergence of the GRF algorithm. Nevertheless, it does not account for the convergence speed of the messages, which affects the algorithm complexity as well. To study the message convergence, the test which generated the most iterations was chosen, which was $\lambda = 200$ for the synthetic data without LOO genes (81 iterations). Figures B.8 and B.9 show the plots for the runtime and number of iterations of the messages, respectively, and Figure B.10 shows the ratio between them. Table 3.4 shows that the average runtime of message convergence is 40.99, which is quite consistent with Figure B.8. It can be seen that the runtime and iterations are kept relatively constant throughout the algorithm (at a rate of , meaning that given a certain dataset and λ value, the rate of GRF message convergence

Iteration 1	Iteration 2	Iteration 3
5.771021	0.739699	0.703344
2.073092	0.05902	0.057702
0.194114	0.002863	0.002744
0.008771	0.000118	0.000107
0.000271	3.98E-06	3.46E-06
1.46E-05	1.17E-07	1.09E-07
3.33E-07	4.35E-09	4.00E-09
1.85E-08	1.99E-10	1.96E-10
6.25E-10	6.97E-12	5.35E-12
4.08E-11	3.84E-13	3.75E-13
1.21E-12	1.41E-14	1.23E-14
8.47E-14	9.70E-16	9.13E-16
2.77E-15	1.22E-16	2.43E-17
2.88E-16		
4.18E-18		

Table 3.8: Euclidean distances for message convergence
The table shows the Euclidean distances between messages for the first 3 iterations of the GRF on synthetic data with $\lambda = 200$.

would have a relatively constant time complexity regardless of the iteration number of the outer loop. In addition, the minor fluctuations in message convergence rate show that the messages actually change (since their Euclidean distances change), and an additional evidence for that can be seen in Table 3.8, which shows that the Euclidean distances of the converging messages in different iterations are different.

The second consideration for algorithm scalability is how the number of genes affects the algorithm runtime. Using $\lambda = 4000$, which generates the fastest GRF convergence, the synthetic data was run on the GRF with 20, 18, 16, 14, 12 and 10 genes each time (no genes removed up to 10 genes removed). The genes were removed in pairs from the same species every time, making sure that when a mouse gene with highest expression from class 0 was removed, a mouse gene with lowest expression from class 1 was removed. This was done to keep the balance in classes, as well as to remove the overlap in expression scores between classes as much as possible. As can be seen in Table 3.9, there is no clear pattern of increase or decrease in runtime and number of iterations, most likely due to the fact that the GRF algorithm does not converge simply based on number of genes, but also on the expressions of those genes. As such, genes who are clearly separated into classes based on their expression scores will probably converge fast, whereas genes from different classes and overlapping expression scores would probably converge slower. Therefore, it might be possible to predict the runtime of the GRF algorithm primarily based on the number of genes it tests and the distribution of their expressions scores. Given those effects on the algorithm, the complexity can now be rewritten as $O(N_{outer}(n, s), N_{inner}(n, s, \lambda)n^2)$, showing that the outer loop is influenced by both number of genes n and their expression scores s and that the inner loop is affected by n, s and λ. Although it wasn't tested fully, several initial trials showed that when the convergence thresholds for the messages and the GRF overall were modified between ϵ and 0.0005,

Number of genes removed	Runtime	Iterations
No genes removed	1280.12	10
2 genes removed	1186.08	18
4 genes removed	1819.14	37
6 genes removed	219.14	6
8 genes removed	381.58	22
10 genes removed	31.23	2

Table 3.9: Runtime and iterations with different gene numbers

the runtime of the algorithm changed as well. Therefore, the algorithm complexity depends on ϵ_1 (message convergence threshold) and ϵ_2 (overall convergence threshold), giving a final complexity $O(N_{outer}(n, s, \epsilon_2), N_{inner}(n, \lambda, s, \epsilon_1)n^2)$. However, the exact complexity of the convergence elements cannot be calculated theoretically, and must be empirically deduced (in the same way the quadratic pattern was found for different λ values).

3.3 Biological data results

3.3.1 Datasets used

For the biological case, data from complete microarrays is used, which usually consist of tens of thousands of genes. Due to computational and time restrictions, it was preferable to use only a small portion of the microarray, and so the GRF analysis was preceded with a filtration step. In addition, microarray data is usually noisy due to technical variations occurring in the microarray experiment (such as extraction of RNA and its labelling), so the microarray data used must be normalised to remove technical variations[7]. However, there are various normalisation techniques, and no single normalisation is always better than others. Therefore, for the biological data, several normalisation techniques were applied to the data and evaluated using quality control plots (see Section 3.3.3). The weight matrix will be calculated from the bit-score matrix obtained by the swalign algorithm in Matlab used for the synthetic data in Section 3.2.1. The biological data used in this study were derived from human and mouse brains, particularly the frontal cortex and hippocampus, respectively. The choice for this dataset is due to the extensive changes occurring in these particular parts of the brain during ageing, and by studying microarrays from these regions it might be possible to find genes that affect information processing and storage in the ageing brain (see Section 1.5.3). The datasets were made publicly available via GEO accession numbers GDS707 for the human dataset[35] and GDS2082 for the mouse dataset[52]. Table 3.10 shows the tissue type tested, the number of samples and their ages, and the microarray platform with the number of probes they contain.

In total, 18255 unique gene IDs are found in both human and mouse. Those gene IDs were mapped to Uniprot protein IDs, finding that only 16917 genes were mapped to 26961 unique proteins. However, not all sequences appear in NCBI, and only 23462 protein sequences were retrieved. This means that the alignment score matrix has (23462*23461)/2=275,220,991 entries. Nevertheless, since this is a very large number of entries, the data is filtered to reduce the number of genes that are dealt with. It is also important to mention that some gene expressions in the microarray are missing, so the data is imputed (See Section 3.3.3).

Biological system	Tissue	No. of samples	Age	MA platform
Human (18 male and 12 female)	Frontal cortex	30	26-106 years old	Affymetrix Human Genome U95 Version 2 Array (12488 probes)
Mouse	Hippocampus	23	2 months and 15 months	Affymetrix Murine Genome U74 Version 2 Array (12625 probes)

Table 3.10: Biological datasets

Biological system	No. probes	No. annotated probes	No. unique genes
Human	12625	12133	9041
Mouse	12488	11930	9217

Table 3.11: Probe and gene counts for the datasets

3.3.2 Data preprocessing

Missing data

The clustering technique of k-nearest neighbours (knn) is commonly used in pattern recognition to distinguish between samples according to their features, and it can also be used to find missing gene expressions. This technique works by finding k genes with expression profile similar to a gene with missing data, and computing a weighted average of values for the missing data based on the expression in other genes. For example, a microarray was used to measure gene expressions for thousands of genes in N experiments, and gene i has a missing value in experiment 1. The technique finds k genes that have similar gene expressions for $N-2$ of the other experiments, and uses the neighbour gene expressions from experiment 1 to calculate a weighted average value, whereas the weight of each neighbour on the average value depends on how similar its gene expression profile is to the profile of gene i. The similarity of gene profiles is based on log Euclidean distances, since logarithm transformation of Euclidean distances (which are sensitive to outliers) reduces the outlier effect[51].

Data filtration

Microarrays often contain tens of thousands of genes, but when using t-test, only a small percentage of those are deregulated, so testing many genes for their deregulation for a certain biological condition is time consuming and reduces the power of the experiment to detect deregulated genes. Filtering out genes with potential to be non-deregulated therefore saves time and ensures a powerful test. Nevertheless, if filtering is not acknowledged as a statistical test, and search for deregulated genes is done on filtered data as if filtering was not actually done, the result would be optimistic

p-values and a larger false positive (type I error)[4].
To solve the problem of increasing type I error, the filtration is done with a criterion that is independent of the statistic used to find deregulated genes (e.g. t-test). This was shown by Bourgon et al.[4], where the filtration criterion were independent of the labels assigned to the samples (biological conditions), including overall mean and variance, median or inter-quartile range (IQR). Thus, as long as the criterion for filtering does not involve the sample labels (in this case, age), genes can be filtered without creating optimistic p-values and while leaving the false positive unchanged.

Normalisation techniques

In microarray experiments, multiple samples hybridise against separate arrays of probes, which can be used to determine mRNA expression in the samples. However, due to technical variations, the intensities cannot be compared directly and must be calibrated, or normalised. Therefore, the normalisation techniques variance stabilisation and normalisation (VSN) and quantile normalisation were chosen to reduce the technical variation while emphasising the biological variation. There are various normalisation techniques, but those two specific ones are chosen as they are frequently used in microarray preprocessing[2].
When dealing with distributions of probe intensities, the variance of distributions often depends on the mean of those distributions. This variance-mean dependence poses a problem, since one assumption of linear models holds that variances are kept constant throughout a distribution, and due to the variance-mean dependence, the intensity distribution does not follow this assumption in low intensity ranges (see Figure 3.1)[29]. The VSN normalisation technique performs rescaling of between-sample variations, such that the variance of probe intensities for each sample is approximately independent of the mean of the probe intensity. This technique uses an inverse hyperbolic sine transformation, through calibrating and shifting the scales of variations between samples, such that the mean is not linearly dependent on the variance, so transcriptional changes can be detected as significant even for lowly expressed genes[29].

Quantile normalisation is used to adjust all intensities within and between samples by calibrating all samples to have the same intensity cumulative distribution. In this technique, the averages of intensities for each probe are taken, while the probes in each sample are ranked from strongest to weakest intensity. The ranked probe intensities in each sample are then replaced with the ranked probe averages, such that all samples have the same distribution overall, but different probe ranking. This technique normalises probe intensity from multiple arrays such that between-sample variations are minimised[3].
VSN focuses on reducing the within-sample variance by removing the variance-mean dependency in each sample separately, while quantile normalisation reduces both within-sample variance (through reducing the intensities to average intensities) and between-sample variance (by giving all samples the same distribution). This implies that VSN corrects the technical variation without influencing the biological one too much, while quantile normalisation corrects for technical and biological variation by bringing samples with similar genetic profiles closer to each other[2].

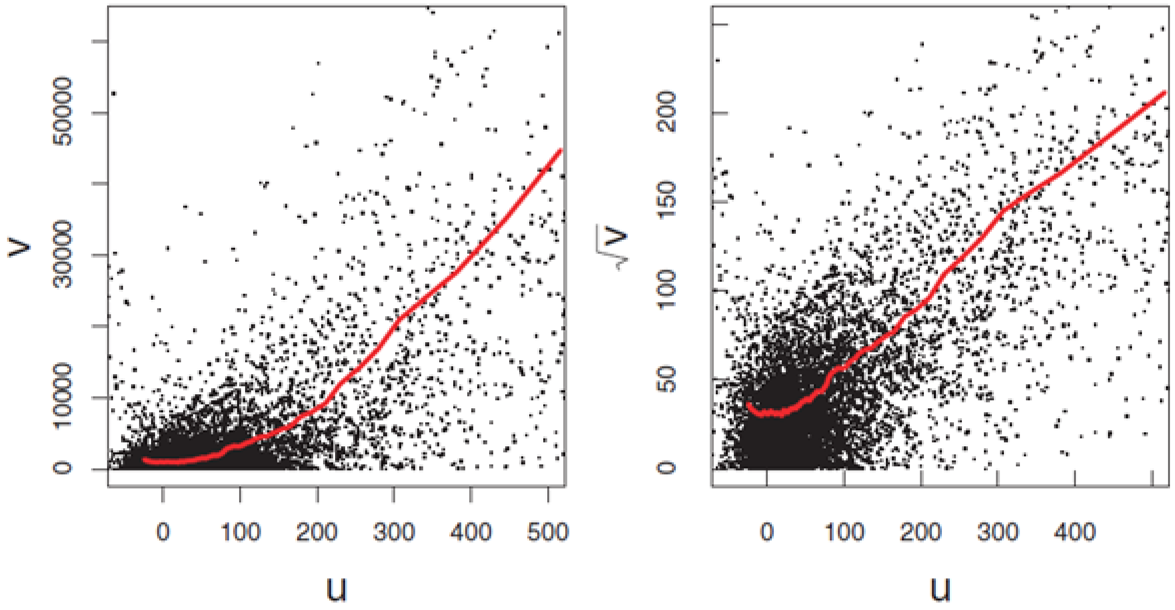


Figure 3.1: Dependence of variance on mean in microarray data distribution
In this figure v is variance, u is the mean. The figure shows how the variance values from microarray data are not linear for lower means[29].

3.3.3 Biological data initialisation

Data imputation and filtration

The imputation of the data was done with default settings ($k = 10$). The filtration of the data was done using the inter-quartile range (IQR) criterion, because it was observed by Bourgon et al.[4] that non-deregulated genes are detected reliably through low variability across samples, and because IQR is robust to outliers. Filtration was done using non-specific filter (nsFilter) and the Bioconductor annotation packages for human (*hgu95av2.db* version 2.5) and mouse (*mgu74av2.db* version 2.5) in R to remove probes that are not annotated, control probes and probes with a low IQR (default threshold at $var.cutoff = 0.5$, meaning that about 50% of the genes with lowest IQR are filtered out). In addition, the IQR criterion was used with the annotation file to filter probes annotated to the same genes, such that only the probe with the highest IQR was kept[4].
After filtering the data with default settings, about 8000 genes were retrieved in total for both datasets. They were merged with the genes mapped to proteins and with the protein sequences retrieved from NCBI, so that each gene had a unique Uniprot ID and unique protein sequence. A total of 7718 protein sequences were found, and retrieving the alignment scores for them ($7717 * 7718/2 = 29779903$ entries) took about 57 hours and 13 minutes.
An attempt was done to input the alignment score matrix into R, but the alignment matrix was too large to be input into R (not enough virtual memory), so it was filtered further using $var.cutoff = 0.9$ for the human and mouse datasets. This returned 1764 genes from the human microarray, and 1762 genes from the mouse microarray, returning an alignment matrix for 3136 proteins.

Data normalisation and quality control

After both normalisation techniques VSN and quantile were applied to the human and mouse datasets separately, it was necessary to determine which normalisation technique is better. As part of this quality control, 3 approaches were employed: (1) MA plots, which show the extent the variance of gene expressions depends on the mean, (2) boxplots, which are used to compare the overall distribution of samples to determine whether they are comparable, and (3) Principal component analysis (PCA), used to determine whether the samples are distinguishable according to their age groups. MA plots are pairwise comparisons of log-intensities between arrays used to identify intensity biases by looking at their ratio and average. The Y-axis represents the log-ratio M between two arrays, and is calculated as $M = log_2 \frac{array1}{array2}$, while the X-axis represents the average intensity A of two arrays, and is calculated as $A = log_2(array1 * array2)/2$. The target MA plot would show the genes under symmetric and even distribution for any two arrays compared. The boxplots are used to compare the overall distribution of the genes in each sample (each box is a sample), and the target plot should be boxes that are similarly distributed[8]. Finally, PCA is a technique that treats the samples as points with n dimensions (where n is the number of probes), and reduces the dimensionality of the samples by finding k new variables (where $k < n$) as linear combinations of the n variables. The new variables are called principal components, and they account for the variation of the n variables while being uncorrelated and orthogonal to each other[44]. The target PCA plot would show the samples clearly separated for different age groups (in mouse case, 2 and 15 months; in human case, below the average age and above the average age of the samples).

As can be seen in Figures C.1, C.2 and C.3, the quantile distribution in both human and mouse improve on the gene distribution compared to their raw forms (VSN normalisation does not improve the distribution as much). Therefore, boxplots show a disposition to quantile normalisation. The MA plots are somewhat difficult to interpret, because VSN generates much smaller expressions for the genes than the quantile normalisation, so MA plots for VSN normalised data are spread in a smaller region and may seem more evenly distributed than the quantile normalised data. Nevertheless, if the scale of the expressions are ignored, Figures C.4, C.5, C.6 and C.7 show human and mouse are more evenly and symmetrically distributed in the quantile normalised data than in the VSN normalised data, which indicates MA plots show quantile normalisation as preferable. Finally, the PCA plots were done separately on mice ages 2 and 15 months, and separately on humans below age of 60 years and above age of 60 years (60 being the mean and median age). Figures C.8, C.9, C.10, C.11, C.12 and C.13 did not show a clear separation of samples from different age groups, thus leaving PCA as inconclusive for this data. The final decision was to use quantile normalisation on the data, since the boxplots show a clear improvement in sample distribution compared to raw data and VSN normalised data, and its MA plots show an even distribution for the quantile normalised data compared with the VSN normalised data.

Expression scores biological data

In the paper by Lu et al.[38], the expression scores were calculated as follows:

$$s_i = \frac{max(expression[i]) - min(expression[i])}{|age(max(expression[i])) - age(min(expression[i]))|}$$

Nevertheless, in the case of Lu et al., the data represented gene expressions of cells attacked by bacteria at different time-points, thus focusing on the same cells at different time-points, so

	Ageing	Non-ageing	Total	Percentage ageing genes
Mouse	96	8714	8810	1.089671
Human	354	8464	8818	4.014516

Table 3.12: Number of ageing and non-ageing genes in unfiltered datasets

that the expressions have gradual changes. However, the ageing biological data involved samples from multiple individuals, so all expressions of every gene had to be considered. Since the filtering guaranteed that every gene is represented by a single vector of probe intensities, the probe intensities were separated into age groups young and old (2 and 15 months for mouse, above or below 60 years for humans). Calculating the difference between young and old samples in both species employed the use of Euclidean distances:

$$s_i = \sqrt{(E(score\_old_i) - E(score\_young_i))^2}$$

Ageing-related genes

The list of ageing genes was retrieved from two main sources: (1) the Gene Ontology (GO) website Amigo[1], used to retrieve all annotations of genes and proteins (Entrez gene ID, MGI IDs, Ensemble IDs and so on) in human and mouse that belong to the GO category of ageing or its children, and (2) the Human Ageing Genomic Resources (HAGR) website[12], used to retrieve all human and mouse genes that were found in studies up to 2010 to be ageing related, but were not necessarily put in ageing gene ontology. From Amigo, 463 entries were obtained for both mouse and human, and from HAGR, 261 Entrez IDs were found for human and 68 gene symbols were found for mouse. With the unfiltered datasets mentioned in Table 3.10, the number of ageing genes in human and mouse was retrieved with the Amigo and HAGR references (see Table 3.12).

With 1% ageing genes in mouse and 4% ageing genes in human, there is a risk that the ageing genes would be under-represented in the Naive Bayes model, and as such the GRF algorithm might tend to give most (if not all) genes a non-ageing classification. Nevertheless, it was attempted to see if this will really happen. To save computation time, both human and mouse datasets were filtered with $var.cutoff = 0.99$, returning 89 genes from each dataset. They were mapped to their equivalent proteins, and their alignment scores were extracted from the alignment matrix with 3136 compared proteins. The final number of genes mapped to proteins and present in the alignment matrix was 154, where 79 genes are from human dataset (4 of which are ageing-related) and 75 genes are from mouse dataset (3 of which are ageing-related).

3.3.4 Biological data - Algorithm execution

Since the number of genes in the biological data is higher than in the synthetic data, it was necessary to reduce the number of neighbours for each gene to allow the algorithm to run faster. As such, the median of the alignment matrix was retrieved (32), and every score below 32 was set to 0. In that way, on average every gene would send messages only to half of its original neighbours (about 77 neighbours on average), so messages would be computed faster and therefore converge faster. In addition, λ was set to 4000 (as it ran fastest for the synthetic data), the message convergence threshold was set to ϵ as before, and the convergence threshold for the GRF overall was set to 0.0005 (allowing faster retrieval of results).

The GRF algorithm did not converge even after 180 hours (648864.25 seconds) and 121 iterations, although it showed signs of slow convergence. The first iteration generated a Euclidean distance of 41089, and within 4 iterations it was reduced to 1579.853. Nevertheless, the Euclidean distances kept fluctuating, as can be seen in Figure 3.2.

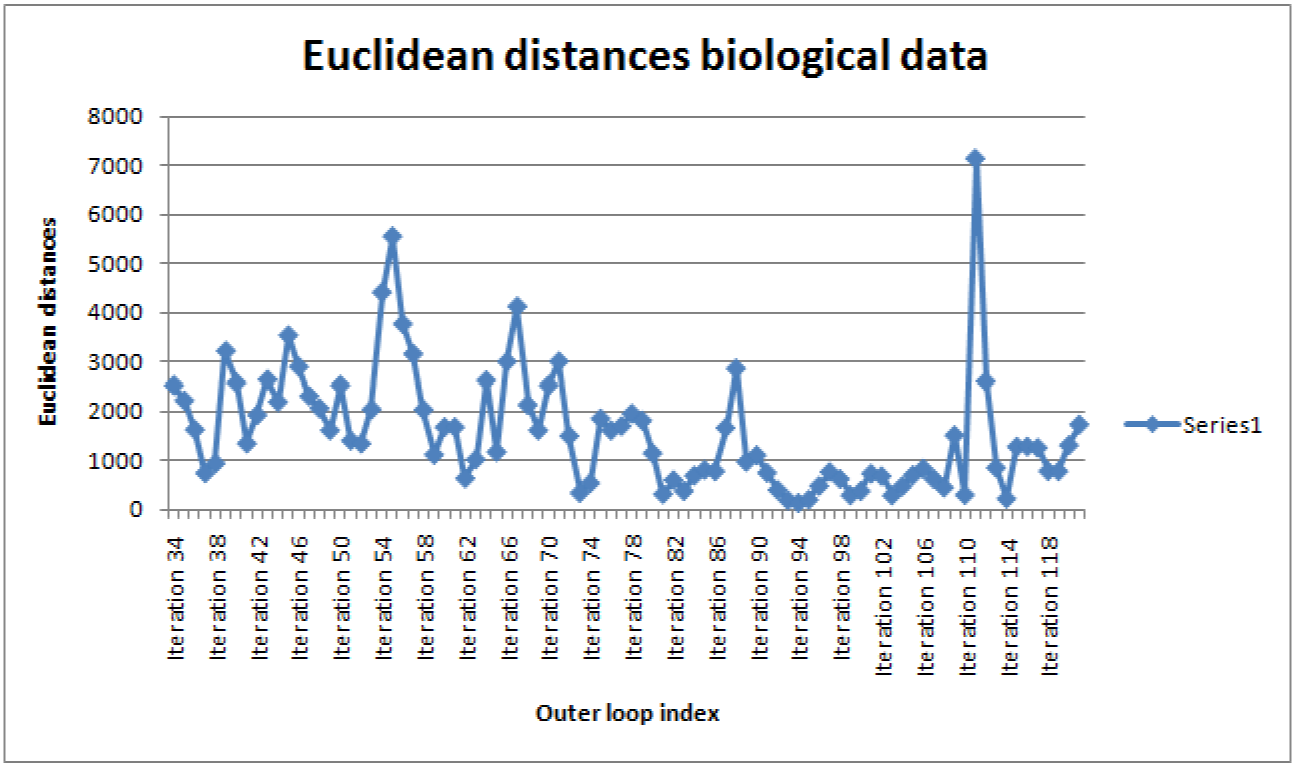


Figure 3.2: Euclidean distances biological data
This plot shows the Euclidean distances of iterations 34 to 121

Another phenomenon observed was that the number of iterations for message convergence remained relatively the same (14-15), and the message convergence runtime also remained in a specific range (5200-5800 seconds), with an average message calculation time of 380 seconds per iteration (so calculating all messages for all genes takes approximately 380 seconds). This has been seen before in the investigation of the inner loop (message) convergence (runtime of message convergence is 37-44 seconds for synthetic data; see Section 3.2.2), and it is possible that the extent to which the message convergence runtime fluctuates depends on the number of genes and their neighbours tested (more neighbours means more calculations and longer convergence runtime).

Since the algorithm did not converge, the iteration with lowest Euclidean distance so far was retrieved (in this case, iteration 94 had Euclidean distance of 150) and all genes with ageing posteriors above 0.5 were retrieved. 82 genes were found to have such posterior probabilities, including ageing-related genes ”57142”, ”7345” and ”1191” in human and gene ”14681” in mouse that had posterior probabilities of 1 (ageing-related genes ”11676” and ”11816” in mouse and gene ”348” in human had very low posterior probabilities). It should be noted that every posterior probability of 1 is actually $1 - \epsilon$ (see Section 2.1.4 for a reminder).

The results show that even with a small number of ageing genes to begin with and with no convergence of the Euclidean distances, multiple genes that are initially non-ageing show high ageing posterior probabilities. Among the 154 genes tested, also several homologous pairs were retrieved, with varying posterior probabilities:

- Aldolase A (ALDOA), an enzyme involved in cytoskeletal protein binding[41], has high posterior probabilities for variants from both species
- Cholecystokinin (Cck), a protein involved in calcium signalling pathways[23], has a mouse variant with high posterior probability, but a human variant with low posterior probability
- Dynamin 1 (Dnm1), a GRF binding protein that functions as parts of signalling complexes used to remodel the actin cytoskelaton[22], has a human variant with high posterior probability, but a mouse variant with low posterior probability
- Metallothionine 3 (Mt3), a metal-binding protein involved in zinc regulation during neural stimulation[18], has a human variant with high posterior probability, but a mouse variant with low posterior probability
- Neurochondrin (Ncdn), a protein that regulates calcium-related processes and may be essential for spatial learning processes[10], has a human variant with high posterior probability, but a mouse variant with low posterior probability
- Neurogranin (Nrgn), a gene that encodes a protein substrate that compensates for calcium absence in related pathways[27], has high posterior probabilities for variants from both species
- Prostaglandin D2 synthase (Ptgds), a neuromodulator (alters nerve impulse transmissions) as well as a factor in the central nervous system[28], has a human variant with high posterior probability, but a mouse variant with low posterior probability
- Proteolipid protein 1 (Plp1), a myelin protein related to axon degradation[20], has low posterior probabilities for both species

Aldoase A and neurogranin are both interesting due to high posterior probabilities for both species, but also due to their functions (involvement in cytoskeletal structure of the cell and involvement in calcium pathways), which have been shown to be ageing related in Section 1.5.3. The other homologous gene pairs, which are partly related to ageing, also show functions that are related to ageing (cell cytoskelaton and structure deterioration, neural stimulations, learning process and calcium pathways), which means that given more time for the algorithm convergence, they might show high posterior probabilities for both variants. Therefore, it can be seen that even with a small initial number of ageing-related genes, the algorithm is able to retrieve multiple homologous genes that are related to the ageing process in the brain, which shows that it can indeed be used to find deregulated genes in cross-species studies other than immune-response microarrays.

The number of genes with posterior probabilities of 1 in iteration 94 is 78, showing that most genes with posterior probabilities bigger than 0.5 are converging to 1. Therefore, it can be expected that if it is allowed to run until the end, the GRF algorithm will probably retrieve all genes with two distinct posterior probabilities of ϵ and $1-\epsilon$. Other genes with posterior probabilities close to 1 are human GRIN1 involved in calcium ion transmembrane transport[1], human creatine kinase involved in brain development[1], mouse neurogranin which may regulate Ca^{2+}-sensitive enzymes[32] and mouse reticulon 3 involved in apoptosis (cell death)[1]. Therefore, it can be seen that some genes appear to be ageing-related even though their homologous pairs are not present in the filtered data. Furthermore, it is possible that if the complete microarray would be run, the homologous genes for those genes may be found to be ageing related as well due to high alignment scores pushing the posterior probabilities of the homologous genes higher.

Chapter 4

Conclusion and discussion

4.1 Algorithm performance

In Section 2.1, the GRF algorithm, originally designed by Lu et al.[38], was fully expressed in terms of a pseudocode and fully explained regarding the use of genes as nodes and alignment scores to establish edges. This explanation will allow future readers of this thesis to implement the algorithm themselves for their own uses.
As mentioned in Section 3.2.2, the GRF algorithm complexity is $O(N_{outer}(n, s, \epsilon_2), N_{inner}(n, \lambda, s, \epsilon_1)n^2)$, showing that it is difficult to theoretically determine how each factor would affect the complexity, and in the same way as in Section 3.2.2, the complexity factors have to be studied empirically. Section 3.2.2 also showed the algorithm has an error rate smaller than 50%, but it is still sufficiently low to consider the algorithm highly accurate at classifying genes. Since the algorithm has been tested before by Lu et al. and was found to work well on biological data both for immune response genes and for brain genes here, it is very likely that the synthetic data was not constructed correctly (it is unclear whether the algorithm can deal with highly overlapping gene expressions for different classes).
It must also be acknowledged that the algorithm implementation here is likely to be different from that mentioned by Lu et al., since certain calculations, such as node potential function initialisation (naive Bayes) and convergence calculation (Euclidean distances), were done differently from the original. Therefore, to test whether the GRF implementation done here was correct, it is necessary to recreate the experiment by Lu et al. to see whether the results using the current GRF algorithm diverge much from the original results.

4.2 Biological conclusions

As mentioned in Section 1.5.3, the hippocampus and frontal cortex are brain regions highly relevant to memory and executive functions, where both diminish in capability as ageing progresses. Particular brain genes deregulated by ageing include immediate early genes (IEGs), which consist of transcriptional factors and effector genes, and genes related to Ca^{2+} pathways or metabolism. As seen in Section 3.3.4, the GRF algorithm discovers genes that may be ageing-related, as well as homologous genes that are ageing-related. Even though there is no indication of the error rate for the biological data, the fact that the GRF algorithm finds several genes to be ageing related, especially those who have functions which suggest association to ageing, indicates that the GRF algorithm definitely has potential for cross-species studies in various biological conditions. Nevertheless, since the algorithm was not run to its full convergence, the results are incomplete, and so it will be necessary to run the algorithm for the same dataset (possibly non-filtered) on a stronger computer that will be able to reach the convergence faster. It should also be emphasised that even after convergence, when the algorithm claims several genes to be ageing-related, those are all statistical findings based on mathematical computation of posterior probabilities, and additional proof of those genes being ageing-related can be found by performing biological validation experiments.

4.3 Outlook

The main problem that was faced during this project was the lack of computational resources, which prevented analysing large biological datasets. Commonly, microarray datasets study tens of thousands of genes, and even a dataset of 12000 brain cell genes (one of the smaller microarrays) was too large for the available computers to deal with. Although the filtering technique used here is very helpful on focusing on genes with high variation (likely to be deregulated), it was used here out of necessity rather than choice, and with the severe filtration of the datasets to about 1% of their original size, it is possible many biologically relevant genes were lost in this step. Therefore, given the ability to use a computer with a stronger processor and more RAM, the algorithm could have been run on the entire biological data to find more ageing-related genes.

The design of the algorithm involves multiple loops and nested loops, as well as slow fluctuating convergence, which can take a very long time for large datasets. It is likely that the algorithm can be modified in order to run more efficiently in terms of space (storage of large biological datasets) and time to calculate the messages faster, and it is also necessary to study the algorithm empirically further to deduce how the convergence can be sped up without creating inaccurate results. The fact that the algorithm is properly explained in Section 2.1 will allow readers to criticise it, to find ways to improve on its performance and to study its complexity further. In addition, it is likely that the complexity of the algorithm can be lowered using any of the algorithmic approaches suggested by Felzenszwalb and Huttenlocher[19], DiMaio and Shavlik[14] or Coughlan and Shen[9].

Felzenszwalb and Huttenlocher suggest how to use belief propagation (BP) for early vision and pixel labelling (based on quantities to estimate the pixels, such as intensity). Their first suggestion involves using negative logarithms for messages to find their message minimums, thus allowing for calculations simpler than message integration (see Section 2.1.5). Another suggestion involves using messages in a bipartite graph, where messages are only established between nodes from different groups, and not from the same group. Finally, they suggest a coarse-to-fine multiscale BP that would calculate a coarse estimate of the messages in the first message iteration and use those as initial messages, while also creating node blocks with the same labels to save computation time on many nodes and their neighbours. Nevertheless, the last suggestion applies to hierarchy structures that can group messages together, and this may not be applied to genes without hierarchy in the same way[19]. DiMaio and Shavlik mention the fact that BP is often used for tree-structured graphs (graphs without cycles), and that in graphs with arbitrary topology there is no guarantee that the optimal results will be found. The topology of the graph was not considered to be an issue in this thesis, but it is possible that certain topological occurrences (such as loops and incomplete node connectedness) might create some computational problems. In addition, DiMaio and Shavlik suggest using an aggregated BP (AggBP), which assumes that if all edge potential functions are equal in a given structure (or substructure), all messages along those edges are the same, and therefore can be easily aggregated and calculated for multiple edges at a time. Nevertheless, this aggregation may not apply in cases of incomplete topology (not all nodes are connected), and so may not apply for the biological data in this thesis (see Section 3.3.3). Another suggestion was to apply Fourier transformations to the messages, which should reduce the complexity of calculating message products[14]. Coughlan and Shen suggest a technique that reduces the complexity of the message calculation by considering sparse neighbourhoods of each gene i, and using constant values for those neighbouring genes who are not in the same state as gene i[9]. The main problem with the techniques here, is that they only apply to Markov Random Fields (MRF), and thus assuming the node values are discrete and can either be grouped or treated as hierarchies, which does not apply for the GRF used in the thesis.

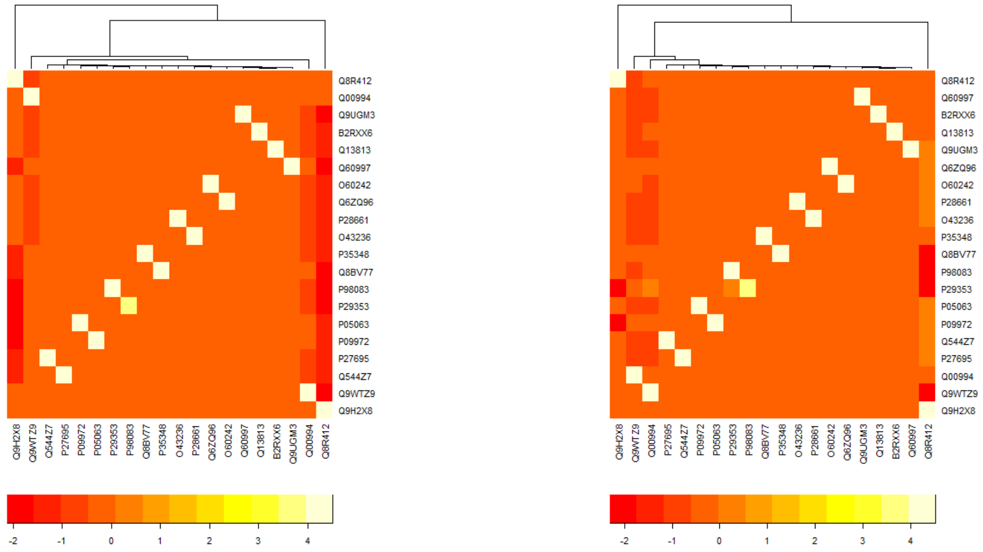
The particular novelty of this study is in the research of the stability and scalability of the GRF and BP algorithm, as well as its ability to classify genes as being deregulated or non-deregulated for ageing. Nevertheless, GRF and BP techniques have been thoroughly used for pattern recognition, graphical models, such as optical flow, 3D biological and chemical imaging (e.g. protein folding structures) and speech recognition, and so the algorithm used here could possibly be adapted for other applications.

4.4 Summary

Cross-species analyses are useful studies to find deregulated genes in multiple microarrays from different species. Nevertheless, there are multiple problems that need to be considered when comparing microarrays from different species, such as different microarray platforms, different ranges of gene expressions and noise due to large environmental variations. The Gaussian Random Field algorithm suggested by Lu et al.[38] was used to perform such cross-species analysis on human and mouse cells to find immunity-associated genes. As the algorithm seemed to perform successfully, and was claimed to be applicable to other organisms and other biological conditions, it was adapted to find ageing-associated genes in brains of mice and humans. First of all, the algorithm was thoroughly explained, and its scalability and its sensitivity were studied. It was also shown that although the biological results found were not conclusive due to use of the small fraction from the studied microarrays, the algorithm has the potential to find ageing-related genes in the data, and might therefore be applicable in further cross-species studies.

Appendix A

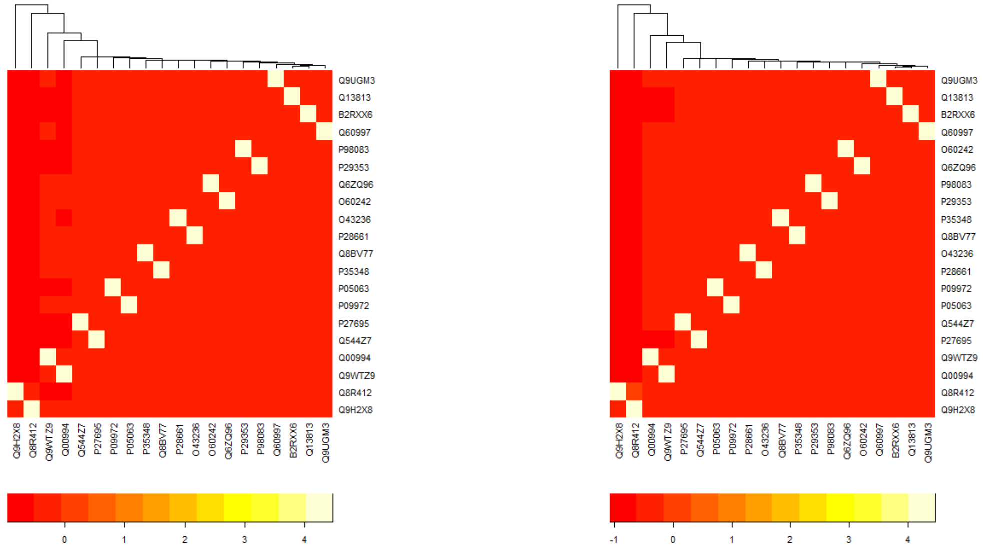
Heatmaps for different sequence alignment techniques



(a) Expect value threshold = 1000

(b) Expect value threshold = 10

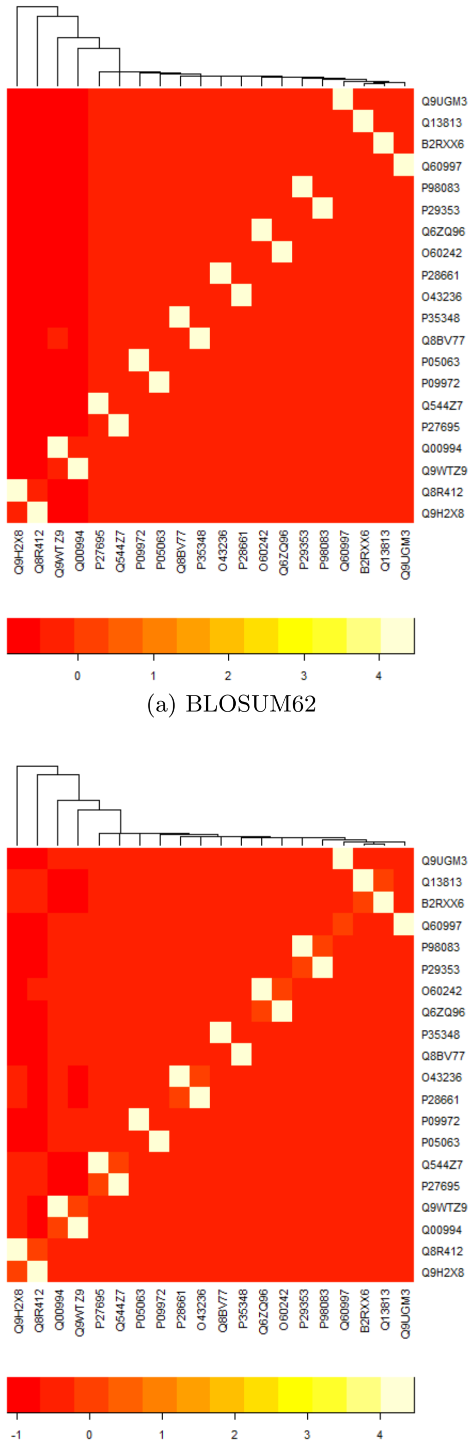
Figure A.1: Heatmap comparison BioPperl



(a) BLOSUM62

(b) PAM250

Figure A.2: Heatmap comparison R



(a) BLOSUM62

(b) PAM250

Figure A.3: Heatmap comparison Matlab

Appendix B

Synthetic data plots and tables

Genes	Other genes	76933	11792	328	6464	20740	6709
20416	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
76933	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
11549	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
11676	1	1	1	1	1	2.22E-16	1
11792	0.999887	1	1	1	1	2.22E-16	1
230	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
328	2.22E-16	2.22E-16	1.23E-14	2.22E-16	2.22E-16	1	2.22E-16
6464	1	1	1	1	1	2.22E-16	1
148	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
83982	1	1	1	1	1	2.22E-16	1
20740	1	1	1	1	1	2.22E-16	1
12945	1	1	1	1	1	2.22E-16	1
210933	1	1	1	1	1	2.22E-16	1
12070	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
18952	1	1	1	1	1	2.22E-16	1
6709	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
1755	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
577	1	1	1	1	1	2.22E-16	1
27018	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
5414	2.22E-16	2.22E-16	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16

Table B.1: Posterior probabilities from cross-validation leave-one-out with $\lambda = 4000$
The first row represents the LOO gene IDs, and the numbers represent the posterior probability for the genes to be ageing-related.

Genes	Other.genes	76933	11792	328	6464	20740	6709
20416	2.22E-16	1	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
76933	3.82E-09	1	2.22E-16	1	1.82E-09	1	2.22E-16
11549	1	2.22E-16	1	1	1	0.999872	2.22E-16
11676	1	1	1	1	1	2.22E-16	2.22E-16
11792	0.999584	3.56E-12	1	0.999989	0.999928	2.22E-16	1
230	2.22E-16	1	2.22E-16	2.22E-16	2.22E-16	1	2.22E-16
328	0.999632	0.999753	0.999999	0.999985	0.99993	5.79E-12	0.00031
6464	1	1	1	1	1	2.22E-16	2.22E-16
148	1	2.22E-16	1	1	1	2.22E-16	0.999668
83982	1	2.22E-16	1	1	1	4.45E-07	1
20740	1.25E-05	2.22E-16	1	1.38E-12	2.05E-08	1	5.49E-06
12945	1	2.22E-16	1	1	1	2.22E-16	1
210933	1	2.22E-16	1	1	1	6.11E-13	1
12070	2.22E-16	1	2.22E-16	1.86E-06	2.22E-16	1	2.22E-16
18952	1	1	1	1	1	2.22E-16	2.22E-16
6709	4.51E-11	1	7.02E-07	2.22E-16	4.77E-14	1	2.22E-16
1755	1	3.11E-05	1	1	1	2.22E-16	0.999998
577	1	2.22E-16	1	1	1	2.22E-16	1
27018	0.999992	1.67E-05	0.00016	1	0.999988	1	2.22E-16
5414	1	2.22E-16	1	1	1	1.83E-05	7.97E-07

Table B.2: Posterior probabilities from cross-validation leave-one-out with $\lambda = 100$
The first row represents the LOO gene IDs, and the numbers represent the posterior probability for the genes to be ageing-related.

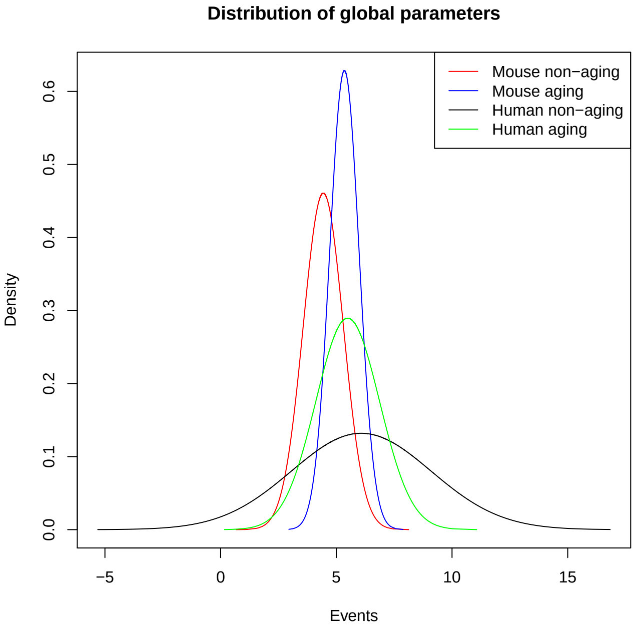


Figure B.1: Distribution of global parameters for synthetic data with λ=100

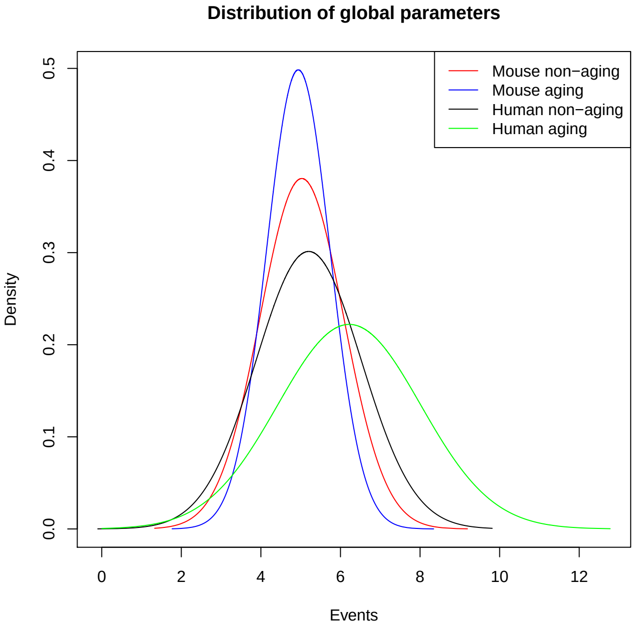


Figure B.2: Distribution of global parameters for synthetic data with λ=200

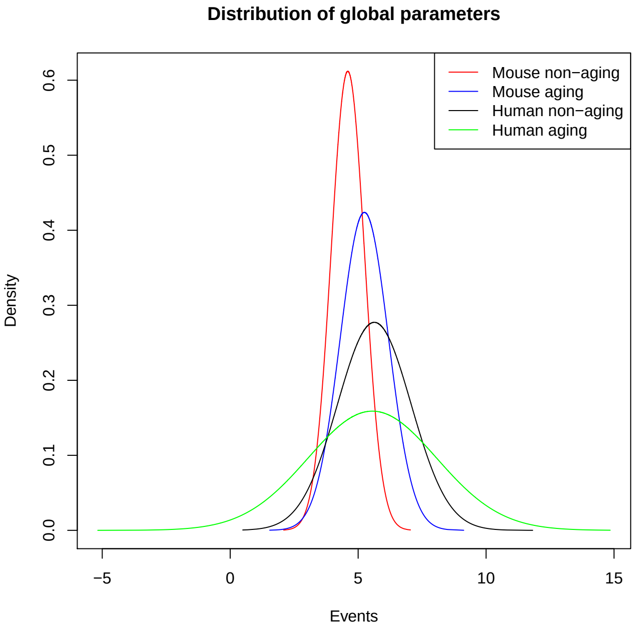


Figure B.3: Distribution of global parameters for synthetic data with λ=800

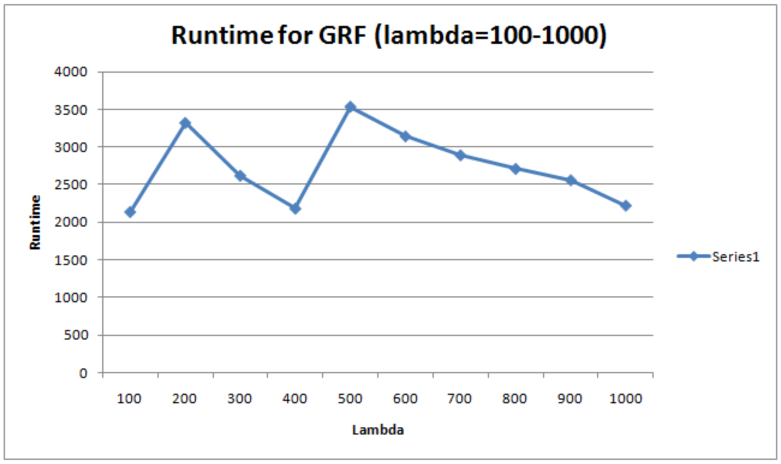


Figure B.4: Runtime of the GRF algorithm for $\lambda = (100, 1000)$

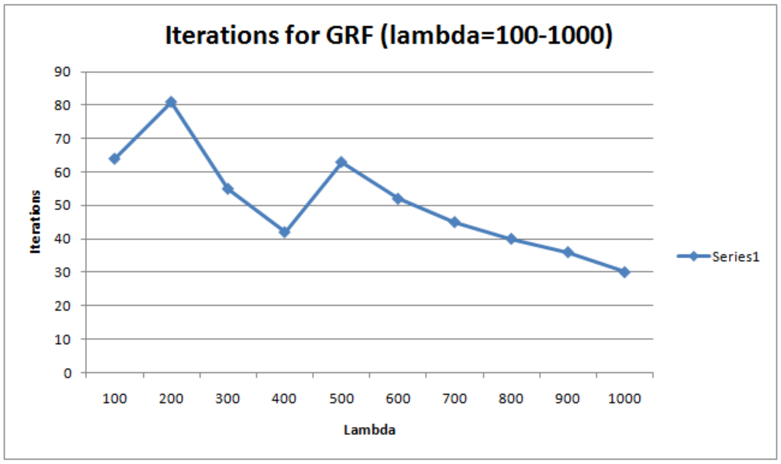


Figure B.5: Iterations of the GRF algorithm for $\lambda = (100, 1000)$

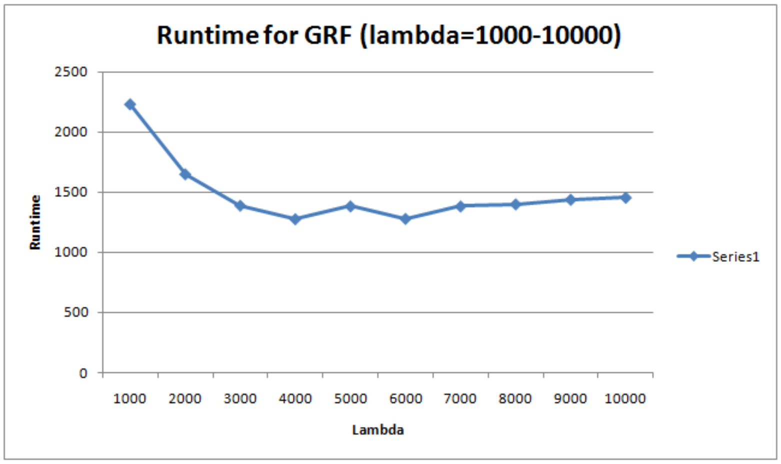


Figure B.6: Runtime of the GRF algorithm for $\lambda = (1000, 10000)$

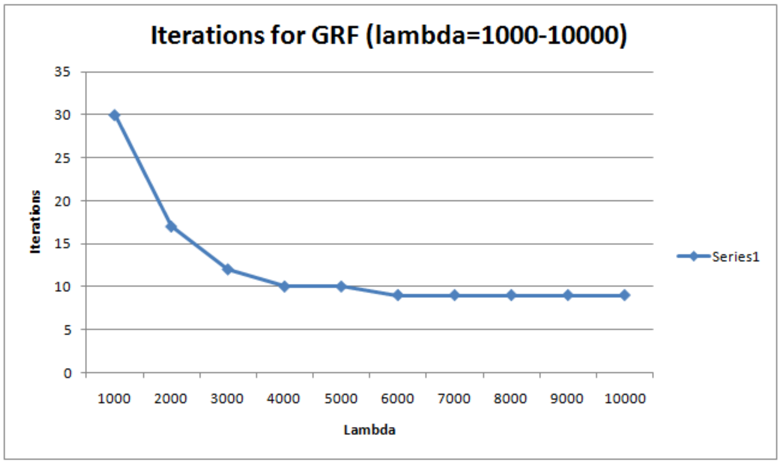


Figure B.7: Iterations of the GRF algorithm for $\lambda = (1000, 10000)$

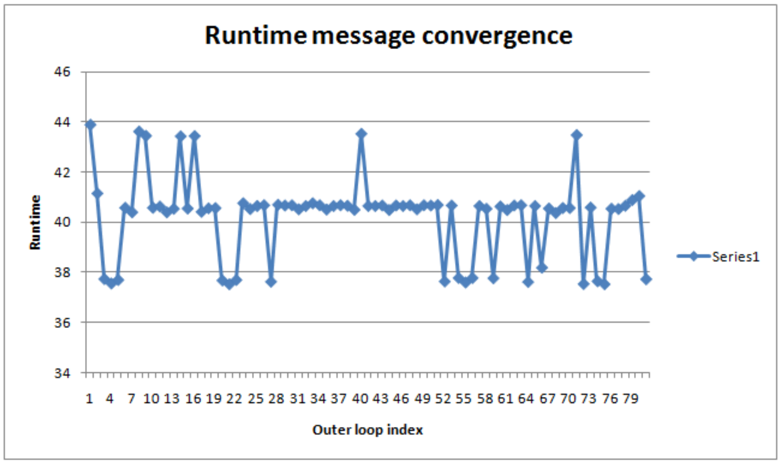


Figure B.8: Runtime of GRF inner loop with $\lambda = 200$

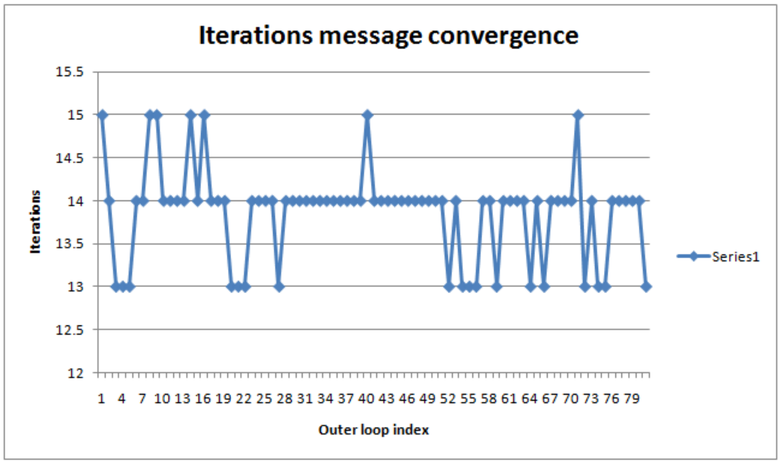


Figure B.9: Iterations of GRF inner loop with $\lambda = 200$

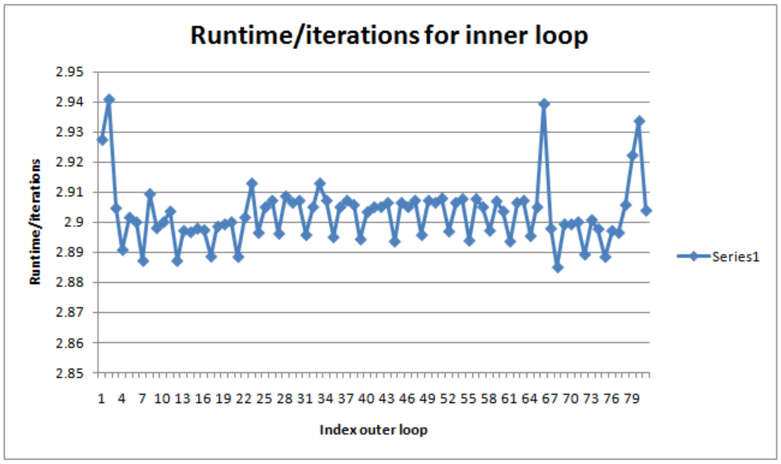


Figure B.10: Runtime/Iteration of GRF inner loop with $\lambda = 200$

Appendix C

Biological data plots and tables

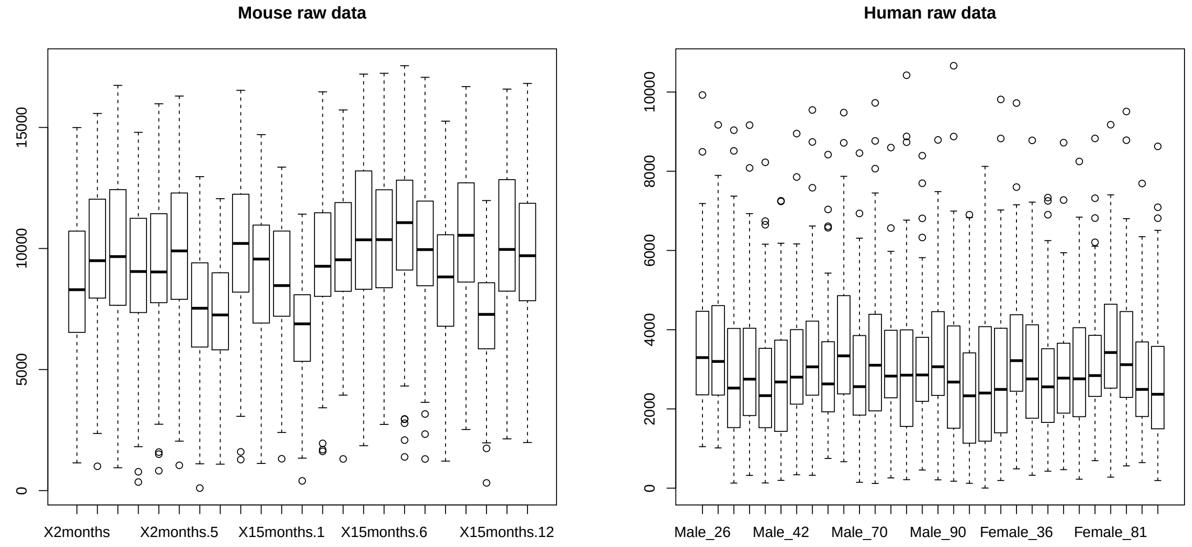


Figure C.1: Boxplots for raw mouse and human data

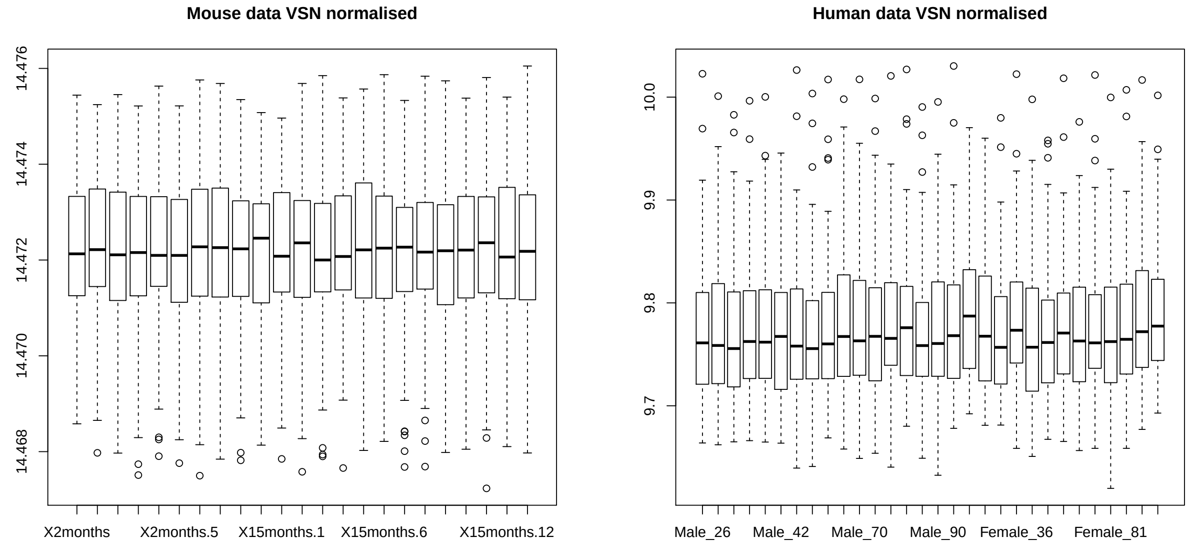


Figure C.2: Boxplots for VSN normalised mouse and human data

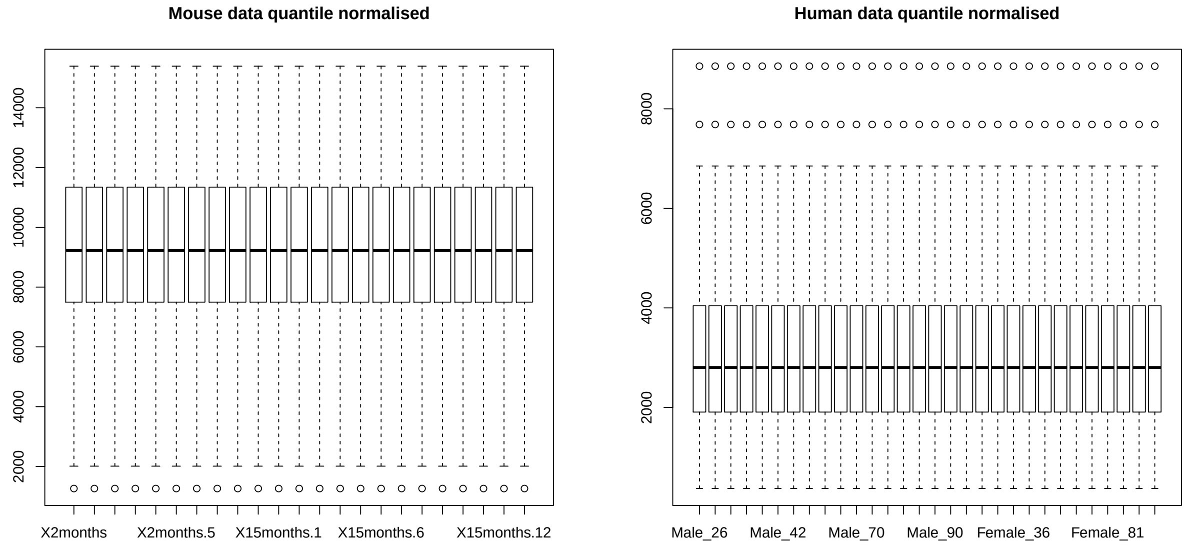


Figure C.3: Boxplots for quantile normalised mouse and human data

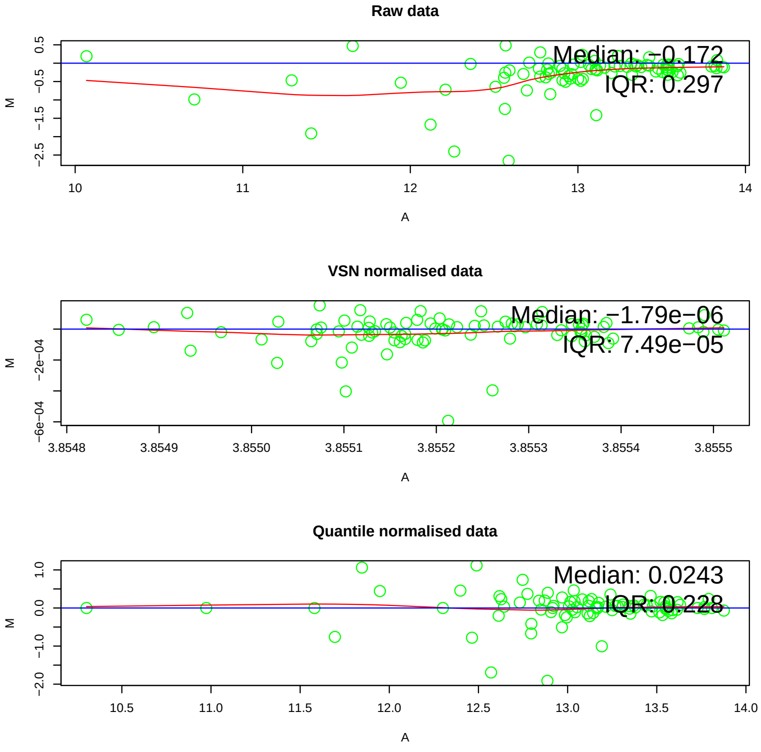
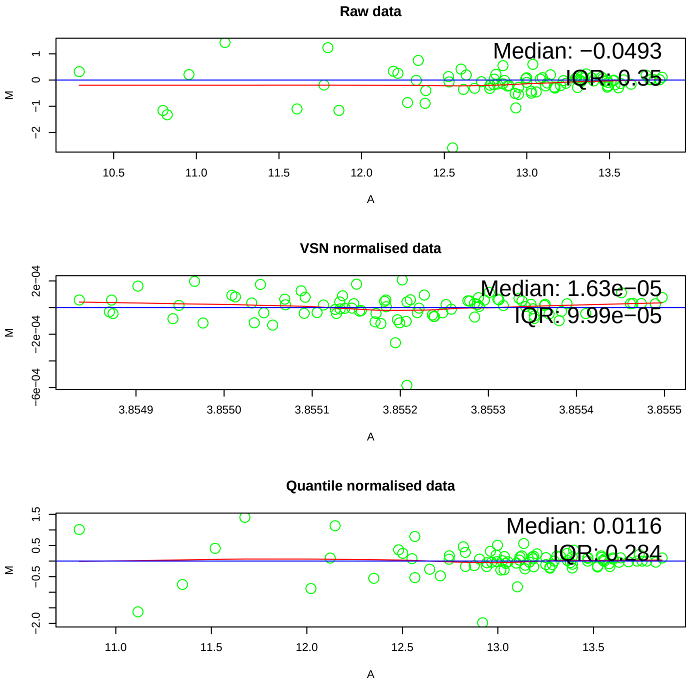


Figure C.4: MA plots for mouse samples of same age
The samples tested are both of age 2 months

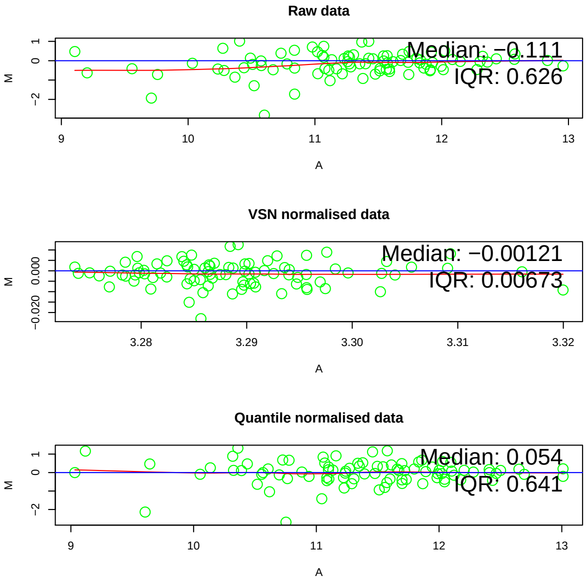


Raw data

VSN normalised data

Quantile normalised data

Figure C.5: MA plots for mouse samples of different ages
The samples tested are 2 months old and 15 months old



Raw data

VSN normalised data

Quantile normalised data

Figure C.6: MA plots for human samples of same age group
The first sample is from an individual age 48 years, and the other is from an individual age 56 years

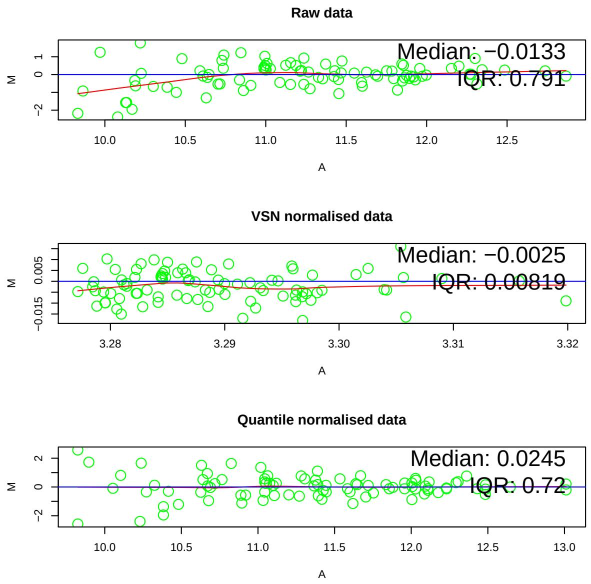


Figure C.7: MA plots for human samples of different age groups
The first sample is from an individual age 48 years, and the other is from an individual age 90 years

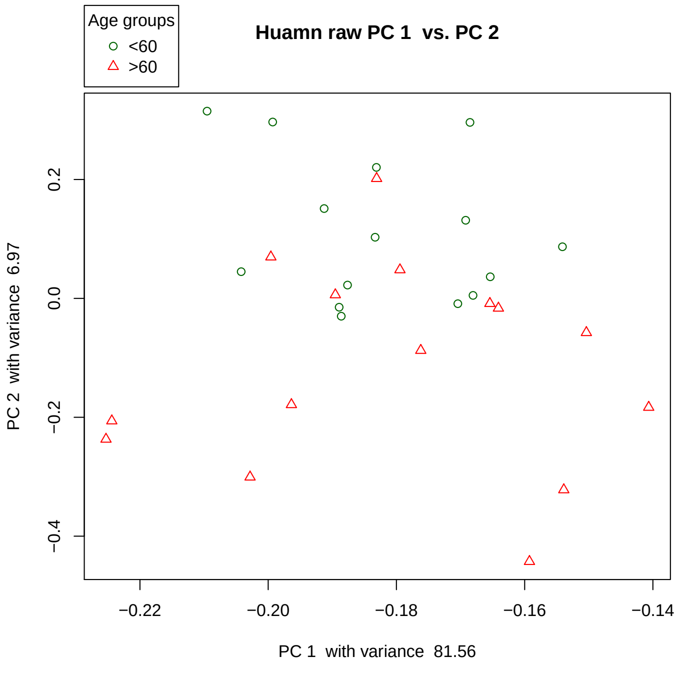


Figure C.8: PCA plot for raw human data

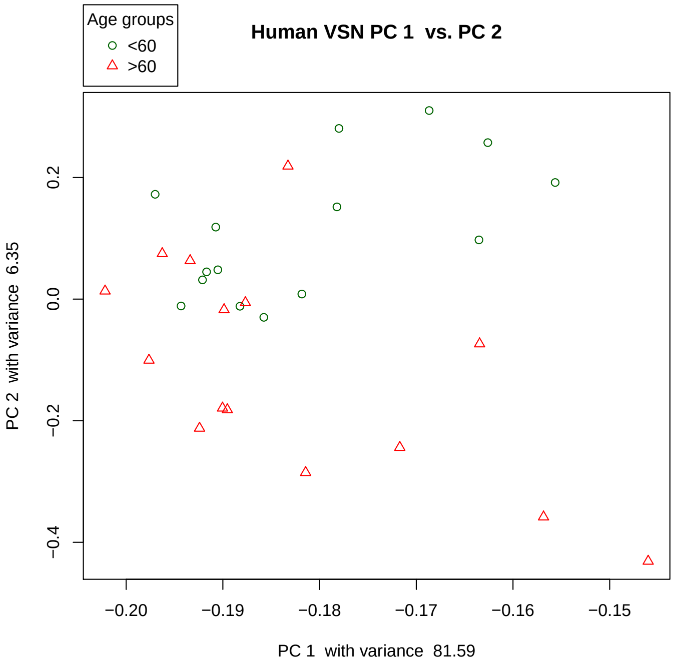


Figure C.9: PCA plot for VSN normalised human data

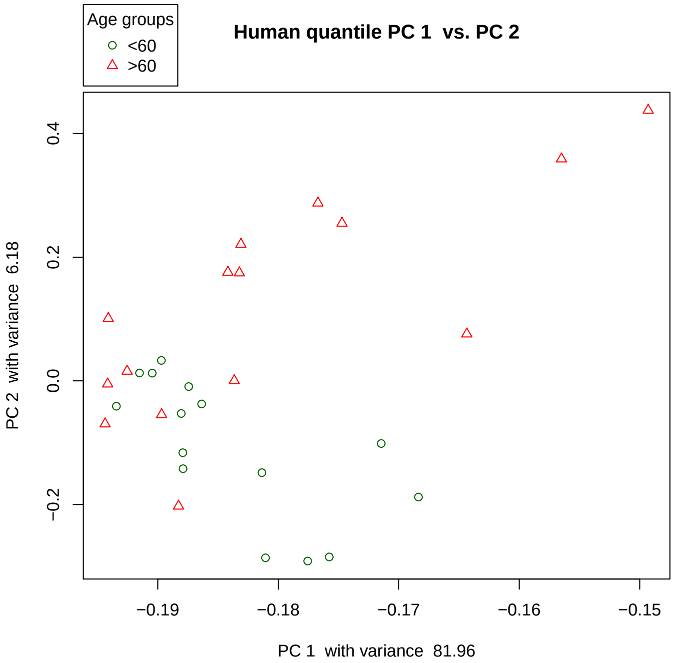


Figure C.10: PCA plot for quantile normalised human data

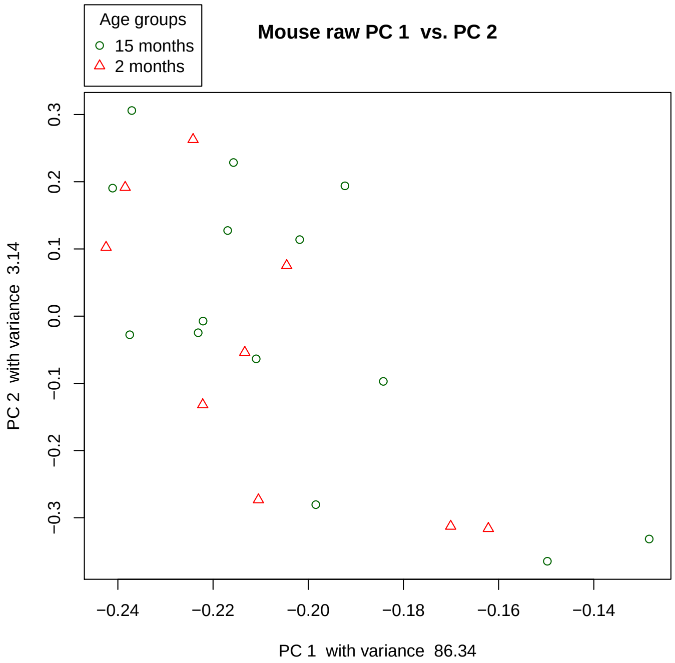


Figure C.11: PCA plot for raw mouse data

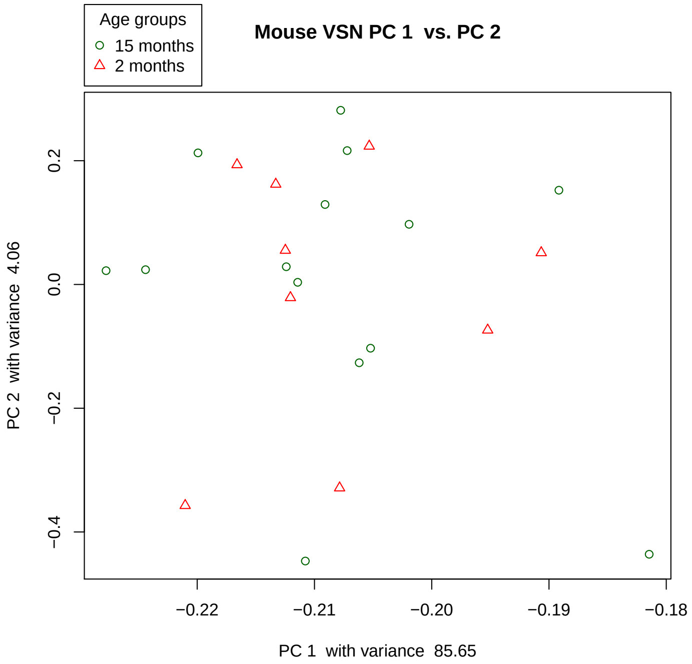


Figure C.12: PCA plot for VSN normalised mouse data

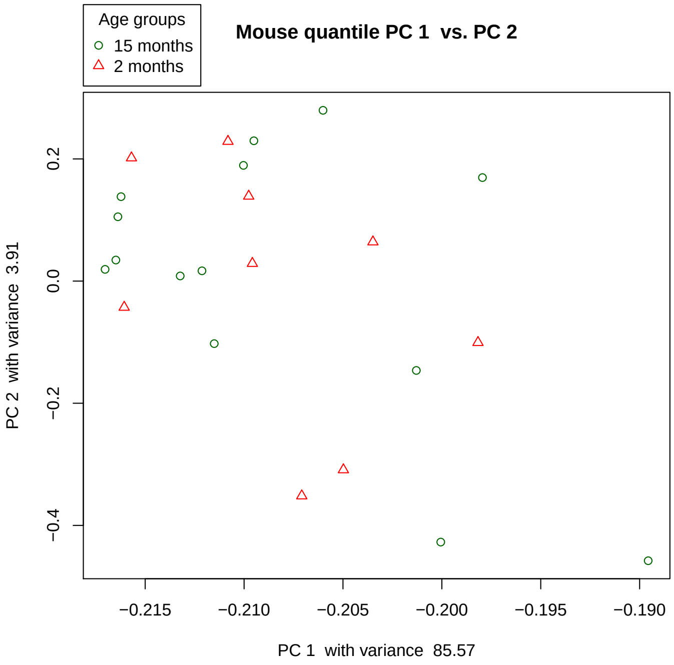


Figure C.13: PCA plot for quantile normalised mouse data

Bibliography

[1] Gene ontology database. Website. Amigo version: 1.8 GO database release 2011-08-13.

[2] Catalin C Barbacioru, Yulei Wang, Roger D Canales, Yongming A Sun, David N Keys, Frances Chan, Karen A Poulter, and Raymond R Samaha. Effect of various normalization methods on applied biosystems expression array system data. *BMC Bioinformatics*, 7:533, 2006.

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

[4] Richard Bourgon, Robert Gentleman, and Wolfgang Huber. Independent filtering increases detection power for high-throughput experiments. *Proc Natl Acad Sci U S A*, 107(21):9546–9551, May 2010.

[5] Warren S Browner, Arnold J Kahn, Elad Ziv, Alexander P Reiner, Junko Oshima, Richard M Cawthon, Wen-Chi Hsueh, and Steven R Cummings. The genetics of human longevity. *Am J Med*, 117(11):851–860, Dec 2004.

[6] Alexander Chan. An analysis of pairwise sequence alignment algorithm complexities: Needleman-wunsch, smith-waterman, fasta, blast and gapped blast.

[7] Gary A. Churchill. Fundamentals of experimental design for cDNA microarrays. *Nature Genetics*, 32 Suppl:490–495, December 2002.

[8] CLC bio, Finlandsgade 10-12, 8200 Aarhus N, Denmark. *Tutorial: Microarray-based expression analysis part II: Quality control*, June 2011.

[9] James M. Coughlan and Huiying Shen. An embarrassingly simple speed-up of belief propagation with robust potentials. *CoRR*, abs/1010.0012, 2010.

[10] M. Dateki, T. Horii, Y. Kasuya, R. Mochizuki, Y. Nagao, J. Ishida, F. Sugiyama, K. Tanimoto, K. Yagami, H. Imai, and A. Fukamizu. Neurochondrin negatively regulates camkii phosphorylation, and nervous system-specific gene disruption results in epileptic seizure. *J Biol Chem*, 280(21):20503–8, 2005.

[11] M. O. Dayhoff and R. M. Schwartz. Chapter 22: A model of evolutionary change in proteins. In *in Atlas of Protein Sequence and Structure*, 1978.

[12] Joao Pedro de Magalhães, Arie Budovsky, Gilad Lehmann, Joana Costa, Yang Li, Vadim Fraifeld, and George M M. Church. The human ageing genomic resources: online databases and tools for biogerontologists. *Aging cell*, November 2008.

[13] Joao Pedro de Magalhães, Joao Curado, and George M Church. Meta-analysis of age-related gene expression profiles identifies common signatures of aging. *Bioinformatics*, 25(7):875–881, Apr 2009.

[14] Frank DiMaio and Jude Shavlik. Improving the efficiency of belief propagation in large, highly connected graphs. 2006.

[15] Sean R Eddy. What is dynamic programming? *Nat Biotechnol*, 22(7):909–910, Jul 2004.

[16] Sean R Eddy. Where did the blosum62 alignment score matrix come from? *Nat Biotechnol*, 22(8):1035–1036, Aug 2004.

[17] A. J. Enright, S. Van Dongen, and C. A. Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res*, 30(7):1575–1584, Apr 2002.

[18] J.C. Erickson, G. Hollopeter, S.A. Thomas, G.J. Froelick, and R.D. Palmiter. Disruption of the metallothionein-iii gene in mice: analysis of brain zinc, behavior, and neuron vulnerability to metals, aging, and seizures. *Journal of Neuroscience*, 17(4):1271–1281, 1997.

[19] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. In *In CVPR*, pages 261–268, 2004.

[20] J.Y. Garbern, D.A. Yool, G.J. Moore, I.B. Wilds, M.W. Faulk, M. Klugmann, K. Nave, E.A. Sistermans, M.S. van der Knaap, T.D. Bird, M.E. Shy, J.A. Kamholz, and I.R. Griffiths. Patients lacking the major cns myelin protein, proteolipid protein 1, develop length-dependent axonal degeneration in the absence of demyelination and inflammation. *Brain*, 125(Pt 3):551–61, 2002.

[21] Marcus J. Grote and Thomas Huckle. Parallel preconditioning with sparse approximate inverses. *SIAM J. Sci. Comput*, 18:838–853, 1996.

[22] Changkyu Gu, Suma Yaddanapudi, Astrid Weins, Teresia Osborn, Jochen Reiser, Martin Pollak, John Hartwig, and Sanja Sever. Direct dynamin-actin interactions regulate the actin cytoskeleton. *EMBO J*, 29(21):3593–606, 2010.

[23] T.V. Hansen and F.C. Nielsen. Regulation of neuronal cholecystokinin gene transcription. *Scand J Clin Lab Invest Suppl*, 234, 2001.

[24] Christopher T Harbison, D. Benjamin Gordon, Tong Ihn Lee, Nicola J Rinaldi, Kenzie D Macisaac, Timothy W Danford, Nancy M Hannett, Jean-Bosco Tagne, David B Reynolds, Jane Yoo, Ezra G Jennings, Julia Zeitlinger, Dmitry K Pokholok, Manolis Kellis, P. Alex Rolfe, Ken T Takusagawa, Eric S Lander, David K Gifford, Ernest Fraenkel, and Richard A Young. Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431(7004):99–104, Sep 2004.

[25] Brian P Head, Jason N Peart, Mathivadhani Panneerselvam, Takaakira Yokoyama, Matthew L Pearn, Ingrid R Niesman, Jacqueline A Bonds, Jan M Schilling, Atsushi Miyanohara, John Headrick, Sameh S Ali, David M Roth, Piyush M Patel, and Hemal H Patel. Loss of caveolin-1 accelerates neurodegeneration and aging. *PLoS One*, 5(12):e15697, 2010.

[26] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A*, 89(22):10915–10919, Nov 1992.

[27] K. Huang and F.L. Huang. Calcium-sensitive translocation of calmodulin and neurogranin between soma and dendrites of mouse hippocampal ca1 neurons. *ACS Chem Neurosci*, 2(4):223–230, 2011.

[28] Z. Huang, Y. Urade, and O. Hayaishi. Prostaglandins and adenosine in the regulation of sleep and wakefulness. *Curr Opin Pharmacol*, 2006.

[29] Wolfgang Huber, Anja von Heydebreck, Holger Sltmann, Annemarie Poustka, and Martin Vingron. Variance stabilization applied to microarray data calibration and to the quantification of differential expression. *Bioinformatics*, 18 Suppl 1:S96–104, 2002.

[30] Alexander W Johnson, Hans S Crombag, Kogo Takamiya, Jay M Baraban, Peter C Holland, Richard L Huganir, and Irving M Reti. A selective role for neuronal activity regulated pentraxin in the processing of sensory-specific incentive value. *Journal of Neuroscience*, 27(49):13430–13435, 2007.

[31] Nevan J Krogan, Gerard Cagney, Haiyuan Yu, Gouqing Zhong, Xinghua Guo, Alexandr Ignatchenko, Joyce Li, Shuye Pu, Nira Datta, Aaron P Tikuisis, Thanuja Punna, Jos M Peregrn-Alvarez, Michael Shales, Xin Zhang, Michael Davey, Mark D Robinson, Alberto Paccanaro, James E Bray, Anthony Sheung, Bryan Beattie, Dawn P Richards, Veronica Canadien, Atanas Lalev, Frank Mena, Peter Wong, Andrei Starostine, Myra M Canete, James Vlasblom, Samuel Wu, Chris Orsi, Sean R Collins, Shamanta Chandran, Robin Haw, Jennifer J Rilstone, Kiran Gandi, Natalie J Thompson, Gabe Musso, Peter St Onge, Shaun Ghanny, Mandy H Y Lam, Gareth Butland, Amin M Altaf-Ul, Shigehiko Kanaya, Ali Shilatifard, Erin O'Shea, Jonathan S Weissman, C. James Ingles, Timothy R Hughes, John Parkinson, Mark Gerstein, Shoshana J Wodak, Andrew Emili, and Jack F Greenblatt. Global landscape of protein complexes in the yeast saccharomyces cerevisiae. *Nature*, 440(7084):637–643, Mar 2006.

[32] Y. Kubota, J.A. Putkey, H.Z. Shouval, and M.N. Waxham. Iq-motif proteins influence intracellular free ca2+ in hippocampal neurons through their interactions with calmodulin. *J Neurophysiol*, 2007.

[33] K. Lin, J. B. Dorman, A. Rodan, and C. Kenyon. daf-16: An hnf-3/forkhead family member that can function to double the life-span of caenorhabditis elegans. *Science*, 278(5341):1319–1322, Nov 1997.

[34] Y. J. Lin, L. Seroude, and S. Benzer. Extended life-span and stress resistance in the drosophila mutant methuselah. *Science*, 282(5390):943–946, Oct 1998.

[35] Tao Lu, Ying Pan, Shyan-Yuan Kao, Cheng Li, Isaac Kohane, Jennifer Chan, and Bruce A Yankner. Gene regulation and dna damage in the ageing human brain. *Nature*, 429(6994):883–891, Jun 2004.

[36] Yong Lu, Peter Huggins, and Ziv Bar-Joseph. Cross species analysis of microarray expression data. *Bioinformatics*, 25(12):1476–1483, Jun 2009.

[37] Yong Lu, Roni Rosenfeld, and Ziv Bar-Joseph. Identifying cycling genes by combining sequence homology and expression data. *Bioinformatics*, 22(14):e314–e322, Jul 2006.

[38] Yong Lu, Roni Rosenfeld, Gerard J Nau, and Ziv Bar-Joseph. Cross species expression analysis of innate immune response. *J Comput Biol*, 17(3):253–268, Mar 2010.

[39] Steven A McCarroll, Coleen T Murphy, Sige Zou, Scott D Pletcher, Chen-Shan Chin, Yuh Nung Jan, Cynthia Kenyon, Cornelia I Bargmann, and Hao Li. Comparing genomic expression patterns across species identifies shared transcriptional profile in aging. *Nat Genet*, 36(2):197–204, Feb 2004.

[40] Joshua J McElwee, Eugene Schuster, Eric Blanc, Matthew D Piper, James H Thomas, Dhaval S Patel, Colin Selman, Dominic J Withers, Janet M Thornton, Linda Partridge, and David Gems. Evolutionary conservation of regulated longevity assurance mechanisms. *Genome Biol*, 8(7):R132, 2007.

[41] M. Merkulova, A. Hurtado-Lorenzo, H. Hosokawa, Z. Zhuang, D. Brown, D.A. Ausiello, and V. Marshansky. Aldolase directly interacts with arno and modulates cell morphology and acidic vesicle distribution. *Am J Physiol Cell Physiol*, 300(6):C1442–55, 2011.

[42] Duncan T Odom, Robin D Dowell, Elizabeth S Jacobsen, William Gordon, Timothy W Danford, Kenzie D MacIsaac, P. Alexander Rolfe, Caitlin M Conboy, David K Gifford, and Ernest Fraenkel. Tissue-specific transcriptional regulation has diverged significantly between human and mouse. *Nat Genet*, 39(6):730–732, Jun 2007.

[43] Todd M Preuss, Mario Cceres, Michael C Oldham, and Daniel H Geschwind. Human brain evolution: insights from microarrays. *Nat Rev Genet*, 5(11):850–860, Nov 2004.

[44] S. Raychaudhuri, J. M. Stuart, and R. B. Altman. Principal components analysis to summarize microarray experiments: application to sporulation time series. *Pac Symp Biocomput*, pages 455–466, 2000.

[45] Tammy M Scott, Inga Peter, Katherine L Tucker, Lisa Arsenault, Peter Bergethon, Rafeeque Bhadelia, Jennifer Buell, Lauren Collins, John F Dashe, John Griffith, Patricia Hibberd, Drew Leins, Timothy Liu, Jose M Ordovas, Samuel Patz, Lori Lyn Price, Wei Qiao Qiu, Mark Sarnak, Jacob Selhub, Lauren Smaldone, Carey Wagner, Lixia Wang, Daniel Weiner, Jacqueline Yee, Irwin Rosenberg, and Marshal Folstein. The nutrition, aging, and memory in elders (name) study: design and methods for a study of micronutrients and cognitive function in a homebound elderly population. *Int J Geriatr Psychiatry*, 21(6):519–528, Jun 2006.

[46] S. K. Shankar. Biology of aging brain. *Indian J Pathol Microbiol*, 53(4):595–604, 2010.

[47] Roded Sharan, Silpa Suthram, Ryan M Kelley, Tanja Kuhn, Scott McCuine, Peter Uetz, Taylor Sittler, Richard M Karp, and Trey Ideker. Conserved patterns of protein interaction in multiple species. *Proc Natl Acad Sci U S A*, 102(6):1974–1979, Feb 2005.

[48] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197, Mar 1981.

[49] Webb Miller Eugene W. Myers Stephen F. Altschul, Warren Gish and David J. Lipmanl. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

[50] T. A. Tatusova and T. L. Madden. Blast 2 sequences, a new tool for comparing protein and nucleotide sequences. *FEMS Microbiol Lett*, 174(2):247–250, May 1999.

[51] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, Jun 2001.

[52] Miguel Verbitsky, Amanda L Yonan, Gal Malleret, Eric R Kandel, T. Conrad Gilliam, and Paul Pavlidis. Altered hippocampal transcript profile accompanies an age-related spatial memory deficit in mice. *Learn Mem*, 11(3):253–260, 2004.

[53] Daniela Wieser, Irene Papatheodorou, Matthias Ziehm, and Janet M Thornton. Computational biology for ageing. *Philos Trans R Soc Lond B Biol Sci*, 366(1561):51–63, Jan 2011.

[54] Adam S. Wilkins. *The Evolution of Developmental Pathways.* Sinauer Associates, Sunderland, MA, USA, 2001.

[55] Xiaojin Zhu, Zoubin Ghahramani, Tommi Jaakkola, and Mit Ii Abstract. Semi-supervised learning with graphs. Technical report, 2005.