ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Transfer Learning of Genome Wide Transcription Dynamics during Malaria Infection

Master Thesis

Venelin Mitov

September 23, 2013

Advisor: Prof. Dr. Manfred Claassen

Department of Computer Science, ETH Zürich

Abstract

Malaria continues to be an endemic disease in vast regions of the world, despite an ongoing active research for its treatment and prevention [WHO, 2012]. One of the major challenges towards understanding the mechanism of the disease in humans at the molecular level is the difficulty to obtain precise post-infection-time series of gene expression profiles in human patients. This thesis project proposes a transfer learning approach to infer post-infection time labels in human patients from controlled time course experiments of malaria infected mice. The proposed approach is supposed to achieve this task on the basis of a gene expression time course dataset of malaria infection in a model organism and a gene expression dataset of malaria infected human individuals with unknown post-infection time. Specifically, we develop and apply our methodology on the basis of an unpublished mouse time series dataset (3 infected mice 1-26 days, 10 uninfected control mice) from our collaborators from the Schneider lab, Stanford University and a published gene expression dataset for a cohort of human individuals [Idaghdour et al., 2012].

The transfer learning approach comprises three steps. First, we build a statistical model of the post-infection time in mice and fit it to the available time-labeled mouse samples. To that end, we explore different classification and regression formulations of supervised post-infection-time inference in malaria infected organisms and evaluate the resulting models with respect to their generalization error and ability to perform automatic variable selection. Next, we train selected model-candidates on infected mouse data, which has been restricted to genes with known homologs in humans. Finally, we apply the resulting models to samples from malaria infected human patients, in order to estimate their post-infection time.

The main contributions of our work are the development of a novel fused elastic net logistic regression model for ordered multi-task classification and the design of a novel ensemble learning method for supervised post-infection time inference, based on the aggregation of simple binary classificaton models. Based on the results, we conclude that the gene-expression profile of an infected host-organism preserves information with respect to the beginning of the infection, and can be used to characterize the disease progression on a fine time-scale.

Acknowledgments

I am greatly indebted to prof. Claassen, the supervisor of this thesis, for his continuous support and guidance. I benefitted very much from his depth of knowledge, professionalism, and scientific intuition. Our numerous meetings provided many insightful ideas and impulses for this work.

Furthermore, I would like to thank my office-mates, Anita, Eirini, Ana and Stefan for interesting discussions as well as for the enjoyable working atmosphere at the institute.

Last but not least, I would like to express my special thanks to our collaborators Brenda and David from the Stanford Microbiology and Immunology Lab, who provided the experimental data for this research.

Contents

List of Figures

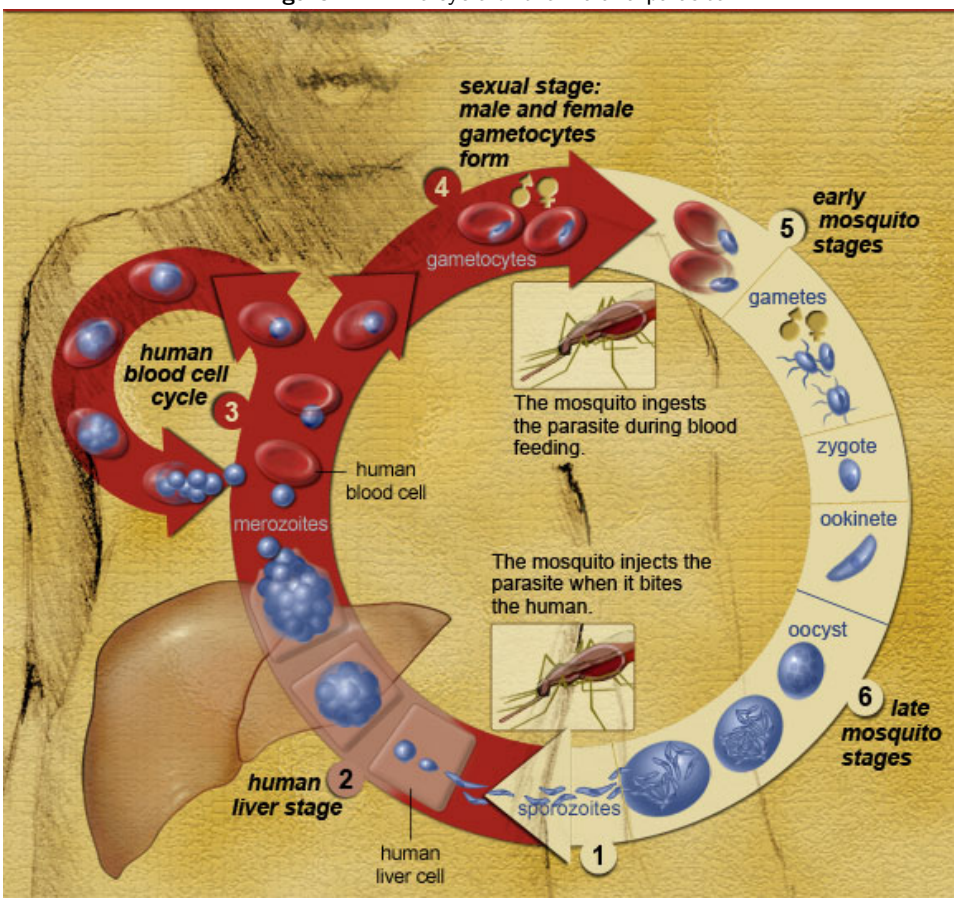
Chapter 1

Introduction

Malaria is a mosquito-borne infectious disease in humans and other vertebrates, which continues to have a tremendous impact on the health and economic situation in vast regions of the world. Due to increasing resistance to the available medications and prevention tools [Parija and Praharaj, 2011, Ranson et al., 2011, WHO, 2012], there is urgent need of development of novel antimalarial drugs to counteract further spread of the disease [WHO, 2012].

Malaria is caused by parasites of genus Plasmodium (*Plasmodium vivax, P. ovale, P. malariae, P. knowlesi and P. falciparum*) which are injected into the human body during the bites/blood-meals of infected female mosquitoes of more than 30 anopheline species WHO [2012]. The life-cycle of Plasmodium parasites takes place in two hosts (Figure 1.1): (i) a vertebrate host, i.e. human, providing the environment for the development of Plasmodium from its immature form, called sporozoite, to its gametocyte producing form, called merozoite; (ii) a vector host, usually a female fertilized mosquito, which takes up Plasmodium gametocytes from the bloodstream of an infected vertebrate, provides these gametocytes with environment for maturing and sexual reproduction and transports newly-born sporozoites to another vertebrate host, where the cycle can begin again Schlagenhauf-Lawlor [2007]. Once injected into the human bloodstream, the unicellular sporozoite travels to a liver cell, where it divides asexually as a schizont, to give thousands of blood-infective merozoites within 1 to 2 weeks Schlagenhauf-Lawlor [2007]. The merozoites quit the liver-cells to enter into the bloodstream, where they penetrate into the erythrocites and continue to reproduce. At regular time-intervals, which's length depends on the Plasmodium species and ranges

Figure 1.1: Life cycle of the malaria parasite



Courtesy: National Institute of Allergy and Infectious Diseases

between 48 and 72 hours, large amounts of erythrocites burst and release newly formed merozoites in the bloodstream Schlagenhauf-Lawlor [2007]. These new parasites can develop into gametocytes, or invade other red blood cells to reiterate the erythrocitic phase. The clinical symptoms of malaria are due to the periodic invasion and destruction of large amounts of red blood cells by the parasites, leading to malarial paroxysm Schlagenhauf-Lawlor [2007]. The malarial paroxysm represents an acute febrile illness, which can rapidly evolve towards a severe complication of the disease, such as serebral malaria or severe anemia Schlagenhauf-Lawlor [2007].

Although much is known about the Plasmodium life cycle and the host-parasite interaction at the cellular level, the underlying molecular mechanisms

remain poorly understood. In a GWA study of Plasmodium falciparum-infected West African children, Idaghdour et al. [2012] confirms a strong effect exerted by malaria infection on the human transcriptome by showing that the gene expression profiles cluster largely, based on the infection status and parasite load of the patients. While Idaghdour et al. [2012] characterizes three major infection states, denoted as "control" (no infection), "low" and "high" parasite load, it is also interesting to understand whether and how the gene expression profile reflects the disease progression on a refined time-scale covering the whole period from the infection through the recovery of the patient. This knowledge might help identifying important gene interactions in the early asymptomatic stages of the disease and unravel potential targets for future drug and vaccine development.

One of the major challenges towards understanding the mechanism of the disease in humans at the molecular level is the difficulty to obtain precise post-infection-time series of gene expression profiles in human patients. Symptom-based post-infection time inference is not possible, because the first symptoms appear after a highly variable incubation period ranging between 1 week and several months [Brasil et al., 2011, Pongsumpun and Mumtong, 2011].

This thesis project proposes a transfer learning approach to infer post-infection time labels in human patients from controlled time course experiments of malaria infected mice. Transfer learning is a method in machine learning that consists in applying the knowledge gained while solving one problem to a different but related problem [West et al., 2007]. For example, a machine learning model for recognition of facial expressions in women can be applied for the recognition of facial expressions in men, because men and women faces share a common set of features. Supposing that a big proportion of the human genes have known homologs in other malaria susceptible vertebrates, such as apes and mice, it should be possible to train a statistical model, like linear regression, on genomic data obtained from a model organism, and apply the fitted model to data comprising homology mapped human genes. Specifically, we developed and applied classification and regression statistical mothods on an unpublished mouse time series dataset (3 infected mice 1-26 days, 10 uninfected control mice) from our collaborators from the Schneider lab at Stanford University and a published gene expression dataset for a cohort of human individuals Idaghdour et al. [2012]. The main contributions of our work can be summarized as follows:

1. Development of the fused elastic net logistic regression (FENLR) model

for ordered multi-task classification - a multi-task classification method that we use to determine the association of a gene expression sample to a given interval in the time-course of the infection (time-window) and to automatically select relevant genes;

2. Development of the Aggregated Time-Window Predictor (ATWINP) - an ensemble machine learning method for gene-expression based post-infection time inference;

Our tests show that, compared to other machine learning methods, ATWINP is significantly better at predicting the post-infection time label of test samples. In a cross-validation test on a dataset comprising 88 samples of 2589 differentially expressed mouse genes the expected difference between predicted and correct post-infection time was estimated at 1.28 days. ATWINP achieved its best predictive performance when using FENLR as underlying time-window classifier. The inherent ability of FENLR to perform automatic feature selection by setting the coefficients of irrelevant features to zero allowed us to identify a gene set whose expression time course dynamics is informative for the inference of post-infection time. Based on these results we conclude that the host's gene-expression profile preserves information regarding the beginning of the infection, and can be used to characterize the disease progression on a fine time-scale.

Thesis Outline

This work is organized as follows.

In Chapter 2, we describe the fused elastic net logistic regression (FENLR) model for ordered multi-task classification. The chapter begins with a theoretical introduction to supervised classification, focusing on linear logistic regression as a method for estimating the conditional class probabilities $\pi(\mathbf{x}) := \mathbb{P}[Y = 1|X = \mathbf{x}]$. We compare the maximum likelihood (ML) and a-posteriori (MAP) estimation approaches and illustrate the combined effect of the Gaussian, Laplacian and fused Laplacian regularizing priors in the case of a single binary classification task. Then we briefly comment on two widely used convex optimization methods, the Newton-Raphson's and the ADMM, which form the algorithmic basis of the FENLR fitting procedure. We introduce multi-task supervised learning through an overview of previous work in the field and give examples of the special case of ordered multi-task binary classification. On that basis, we formulate the FENLR model and develop a novel numerical algorithm for finding its estimate, which adapts very well to

the case of high dimensional data with more predictor variables than training observations ($d \gg n$). To demonstrate the performance of the model in the ordered multi-task setup, we report results from experiments with synthetic data.

In chapter 3, we investigate different supervised learning formulations of the problem of genome based post-infection time inference in malaria-infected organisms. We formulate the aggregated time-window predictictor (ATWINP) and two conventional methods: penalized linear regression and first nearest neighbor. All methods are compared based on their predicting performance, which is measured in terms of expected deviation between the predicted and the true day of infection. By selecting the model with minimal mean error and analyzing its coefficient profile, we provide a list of selected genes, which should be informative for analyzing the disease progression.

The thesis work ends with a discussion of the results and an outlook for future work.

Chapter 2

Multi-Task Learning for Ordered Classification

2.1 Introduction

A decisive cognitive skill in humans is their ability to relate a new concept to a known one. This learning process consists in recognizing the qualitative similarities and differences between the building elements of the new and the known concept. The same idea applies when a human has to learn a set of tightly related new concepts: the ability to build associations between the new concepts in terms of resemblances and contrasts can tremendously improve the quality and the speed of learning. Multi-task learning is the translation of this natural approach to the domain of pattern recognition and machine learning. Usually, this approach is accomplished by building a model that (i) includes the *a-priori* knowledge about task relatedness, and (ii) allows for the tasks to be learned jointly, such that learning one task has a positive effect on the learning of its related tasks. The rest of this section recapitulates several basic machine learning concepts and definitions, establishes the mathematical notation for the rest of this chapter and gives a brief overview of previous work in the field of multi-task learning. section 2.2 describes the fused elastic net logistic regression model for ordered multi-task binary classification. In section 2.3, we report some experiments of the method conducted on synthetically generated data. Chapter 3 will expand the application of this method to a real-world setting consisting of estimating post-infection time in malaria infected organisms. This chapter concludes with a short discussion of the results and findings.

2.1.1 Supervised machine learning and classification

In supervised learning, a learning task consists in inferring a function from a labeled training data, which makes a "prediction" of the label, when evaluated on a new data-point [Mohri et al., 2012]. Depending on the type of the output labels, we distinguish two broad families of supervised learning tasks, namely, regression for numerical labels, such as real numbers, and classification for labels that are taken from some finite set of categories. A common approach to supervised learning consists of building a parametric model of the output-label as a function of the features describing the data, and to fit the parameters of the model to a set of "training" observations, assuming that these training observations are sampled independently and identically (abbr. i.i.d.) from the unknown true distribution of the data.

Classification constitutes a well studied family of supervised learning tasks with a broad range of applications. Given is training data which are realizations from

$$(X_1, Y_1), ..., (X_n, Y_n) \text{ i.i.d. },$$

where the predictor or feature vector $X_i \subset \mathbb{R}^d$, $i = 1, ..., n$, is a random vector and the vector of classes or labels $Y = (Y_1, ..., Y_n) \subset \{0, 1, ..., J-1\}^n$ is a discrete random vector. We denote the training data as the extended matrix $[X|\mathbf{y}]$, where $X = [\mathbf{x}_1, ..., \mathbf{x}_n]^T \in \mathbb{R}^{n \times d}$ is called the design matrix, and the vector $\mathbf{y} = (y_1, ..., y_n) \in \{0, 1, ..., J-1\}^n$ is called the response vector. The rows of $[X|\mathbf{y}]$, $(\mathbf{x}_i, y_i)^T$, are called training observations. The numbers $0, 1, ..., J-1$ denote class-labels with or without ordering between them. The goal is to find a function called classifier $\mathcal{C} : \mathbb{R}^d \to \{0, 1, ..., J-1\}$, assigning to a predictor vector $\mathbf{x} \in \mathbb{R}^d$ an output label, which is a prediction for the corresponding true label y. A common performance measure for a classifier is the expected zero-one test set error [Hastie et al., 2001, Bühlmann and Mächler, 2011]:

$$L_{01}(\mathcal{C}) := \mathbb{P}[\mathcal{C}(X_{new}) \neq Y_{new}]. \tag{2.1}$$

Let

$$\pi_j(\mathbf{x}) := \mathbb{P}[Y = j | X = \mathbf{x}], \ j = 0, 1, ..., J-1 \tag{2.2}$$

be the conditional probability that a given sample $\mathbf{x}$ is labeled by j. The Bayes classifier, defined for each $\mathbf{x}$ individually as

$$\mathcal{C}_{Bayes}(\mathbf{x}) := \arg \max_{0 \leq j \leq J-1} \ \pi_j(\mathbf{x}), \tag{2.3}$$

is the optimal classifier with respect to the zero-one error and the minimum of this error function, known as the Bayes risk is [Bühlmann and Mächler,

2011]:

$$\mathbb{P}[\mathcal{C}_{Bayes}(X_{new}) \neq Y_{new}] \tag{2.4}$$

In reality, the conditional probability distributions $\pi_j(\cdot)$ are not known and it is impossible to construct the Bayes classifier. Therefore, the common approach is to obtain a multivariate function estimate $\hat{\pi}_j(\cdot)$ of $\pi_j(\cdot)$ and to plug it into the definition of the Bayes classifier, to obtain an estimated classifier:

$$\hat{\mathcal{C}}(\mathbf{x}) := \arg \max_{0 \leq j \leq J-1} \hat{\pi}_j(\mathbf{x}), \tag{2.5}$$

2.1.2 Linear logistic regression for binary classification

In this chapter, we concentrate on the case $J = 2$, known as binary classification, and we simplify the notation by denoting$\hat{\pi}_1(\mathbf{x})$ as $\hat{\pi}(\mathbf{x})$. Obtaining the estimate $\hat{\pi}(\mathbf{x})$ is sufficient to define $\hat{\mathcal{C}}(\mathbf{x})$, as $\hat{\pi}_0(\mathbf{x}) = 1 - \hat{\pi}(\mathbf{x})$. We consider linear logistic regression (LLR) as our method of choice for finding an estimator $\hat{\pi}(\mathbf{x})$. While it performs comparably to competing methods, such as support vector machines (SVM) and linear discriminant analysis (LDA), LLR has some notable advantages in that it provides a direct estimate of the probability $\pi(\mathbf{x})$ and tends to be more robust in the case $d \gg n$. Given the multivariate function $\pi : \mathbb{R}^d \to [0,1]$, the strictly monotone logistic transform $\pi \to log(\pi/(1-\pi)) =: logit(\pi)$ maps the interval $(0,1)$ to the real line $\mathbb{R}$ and makes it possible to use any real valued function as a model for the logistic transform of π and then to apply the inverse transform $logit(\pi) \to \frac{\exp(logit(\pi))}{1+\exp(logit(\pi))} = \pi$ to obtain a probability estimate $\hat{\pi} \in (0,1)$. Linear logistic regression (LLR) is defined as the model

$$logit(\pi(\mathbf{x})) \approx \beta_0 + \mathbf{x}^T \boldsymbol{\beta}_{\setminus 0} =: g(\mathbf{x}), \tag{2.6}$$

in which the logistic transform is modeled as the linear function g of the predictor vector $\mathbf{x}$ with coefficients $\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_d)^T$. To simplify the notation, we assume that we have added an offset predictor variable equal to 1 as first component of the predictor vector $\mathbf{x}$, so that we can write g as a vector product:

$$g(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} \tag{2.7}$$

The inverse logit transform, $logit^{-1}$, gives the formula for calculating $\pi(\mathbf{x})$ from $g(\mathbf{x})$:

$$\begin{aligned} \pi(\mathbf{x}) &= \frac{\exp[logit(\pi(\mathbf{x}))]}{1+\exp[logit(\pi(\mathbf{x}))]} \\ &\approx \frac{\exp[\mathbf{x}^T\boldsymbol{\beta}]}{1+\exp[\mathbf{x}^T\boldsymbol{\beta}]} \end{aligned} \tag{2.8}$$

For a fixed design matrix X, the conditional likelihood function of the model coefficients $\boldsymbol{\beta}$ is defined as the conditional probability distribution of the response vector Y, given X and $\boldsymbol{\beta}$:

$$L_{cond}(\boldsymbol{\beta}; [X|\mathbf{y}]) : = \mathbb{P}[Y = \mathbf{y}|X, \boldsymbol{\beta}] \tag{2.9}$$

$$= \prod_{i=1}^{n} Bernoulli(Y_i = y_i; \pi(\mathbf{x_i}; \boldsymbol{\beta})) \tag{2.10}$$

$$= \prod_{i=1}^{n} \pi(\mathbf{x}_i; \boldsymbol{\beta})^{y_i}(1-\pi(\mathbf{x}_i; \boldsymbol{\beta}))^{1-y_i} \tag{2.11}$$

Often it is practical to use the negative natural logarithm of the conditional likelihood, because it converts all products into sums, without shifting the optimum of the likelihood:

$$-\ell_{cond}(\boldsymbol{\beta}; [X|\mathbf{y}]) : = -\log\{L_{cond}(\boldsymbol{\beta}; [X|\mathbf{y}])\} \tag{2.12}$$

$$= -\sum_{i=1}^{n} \left\{ y_i \mathbf{x}_i^T\boldsymbol{\beta} - \log[1+\exp(\mathbf{x}_i^T\boldsymbol{\beta})] \right\} \tag{2.13}$$

A useful trick that allows us to write 2.13 in a more convenient form, is to substitute the responses $y_i \in \{0,1\}$ by $\tilde{y}_i := (2y_i - 1) \in \{-1,1\}$. By plugging $y_i = (\tilde{y}_i + 1)/2$, and representing the left summand $y_i \mathbf{x}_i^T\boldsymbol{\beta}$ in the form $\log[\exp(\frac{1}{2}\tilde{y}_i \mathbf{x}_i^T\boldsymbol{\beta} + \frac{1}{2}\mathbf{x}_i^T\boldsymbol{\beta})]$ the negative conditional log-likelihood can be written as:

$$-\ell_{cond}(\boldsymbol{\beta}; [X|\mathbf{y}]) = \sum_{i=1}^{n} \log\left(1+\exp(-\tilde{y}_i \mathbf{x}_i^T\boldsymbol{\beta})\right) \tag{2.14}$$

$$= \sum \log\left(\mathbf{1}+\exp(-\tilde{\mathbf{y}} \odot X\boldsymbol{\beta})\right), \tag{2.15}$$

where in the last line the symbol $'\sum'$ denotes sum over all elements of the underlying vector, the symbol '$\odot$' denotes the element-wise multiplication between vectors or matrices with the same dimensions and the bold number $\mathbf{1}$

denotes the n-dimensional real vector having all elements equal to 1. Further in this chapter, we will always use the transformed response $\tilde{y} \in \{-1, 1\}$, and, to simplify the notation, we will omit the superscript '~'.

Fitting the coefficients $\boldsymbol{\beta}$ to the training data $[X|\mathbf{y}]$ is done by maximizing the conditional likelihood or, equivalently, by minimizing the negative conditional log-likelihood:

$$\hat{\boldsymbol{\beta}}_{ML} : = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{1+\mathbf{d}}} -\ell_{cond}(\boldsymbol{\beta}; [X|\mathbf{y}]) \tag{2.16}$$

where $\hat{\boldsymbol{\beta}}_{ML}$ is called the maximum conditional likelihood estimate of $\boldsymbol{\beta}$, given training data $[X|\mathbf{y}]$. While no analytic solution is known for this convex nonlinear optimization problem, the double differentiability of the objective function $-\ell_{cond}$ with respect to $\boldsymbol{\beta}$ allows to find it's global minimum efficiently using the Newton-Raphson's gradient descent method, which we briefly describe in section 2.1.4.

Note that to obtain the estimate $\hat{\boldsymbol{\beta}}_{ML}$ via linear logistic regression, we didn't make any assumption about the true distribution of the predictors X. In contrast, another widely used model for classification, linear discriminant analysis (LDA), assumes normal conditional distributions of the predictor variables given their assigned class labels, $(X|Y = j) \sim \mathcal{N}_d(\boldsymbol{\mu}_j, \Sigma)$, $j = 0, 1$, and maximizes the full likelihood

$$L(\boldsymbol{\beta}; [X|\mathbf{y}]) : = \mathbb{P}[Y = \mathbf{y}, X|\boldsymbol{\alpha}] \tag{2.17}$$

$$= \mathbb{P}[X; \boldsymbol{\alpha}]\mathbb{P}[Y = \mathbf{y}|X, \boldsymbol{\alpha}], \tag{2.18}$$

where $\boldsymbol{\alpha}$ are the linear coefficients of the LDA model. As explained in Hastie et al. [2001], p.127-128, the exact functional form of $\mathbb{P}[Y = \mathbf{y}|X, \boldsymbol{\beta}]$ in eq. (2.9) and $\mathbb{P}[Y = \mathbf{y}|X, \boldsymbol{\alpha}]$ in eq. (2.18) is the same, but with the inclusion of the marginal density, $\mathbb{P}[X; \boldsymbol{\alpha}]$, in the full likelihood, LDA incorporates more information about the coefficients $\boldsymbol{\alpha}$, so they can be estimated more efficiently, i.e. based on fewer training observations.[1]On the other hand, calculating $\mathbb{P}[X; \boldsymbol{\alpha}]$ in (2.18) requires the empirical estimation of the Gaussian moments $\boldsymbol{\mu}_j$ and Σ. While estimating the common covariance matrix Σ empirically might be affordable, because it doesn't need labeled observations, the empirical calculation of the first moments $\boldsymbol{\mu}_j$ can be misleading in the cases of violations of the normality assumption, few training observations or

[1]According to Hastie et al. [2001], in the worst case, maximizing the conditional instead of the full likelihood might result in about one third more training observations needed to achieve the same predictive performance.

the presence of outliers in the training data and, particularly in the case of our interest, $d \gg n$. Unless otherwise stated, through the end of this chapter, we will always work with conditional log-likelihoods, and in order to avoid cumbersome terminology, we will skip the subscript "*cond*".

Another competing model, the Support Vector Machine (SVM) [Hastie et al., 2001], is known to have nearly equivalent performance on predicting the correct labels, as LLR, but doesn't provide a direct estimate of the conditional probabilities π_j . Since our real world application (see chapter 3) requires the specification of these conditional probabilities, we abstained from also exploring SVM classification.

2.1.3 Regularization and variable selection through penalization

Two important criteria in help of evaluating the quality of a model areZou and Hastie [2005]:

1. accuracy of predicted outcome on unseen data - a model that predicts poorly is hard to defend;
2. interpretation of the model - models that reveal important relationships between the outcome and the covariates are often helpful for researchers, who struggle to understand the underlying mechanisms of some studied process.

A toy example We simulate training data $[X|\mathbf{y}]$, $X \in \mathbb{R}^{8\times 4}$, $\mathbf{y} \in \{0,1\}^8$. The covariate vectors $X_{\cdot 1}$, $X_{\cdot 3}$, $X_{\cdot 4}$ are sampled from the standard normal distribution, $\mathcal{N}(0,1)$, and $X_{i,2}$, $i = 1,...,8$, are sampled from $\mathcal{N}(X_{i,1}, 0.5)$ resulting in $cov(X_{\cdot 1}, X_{\cdot 2}) = 0.86$. This setting simulates an often encountered practical case, when one of the covariates is a noisy copy of another one, resulting in very high correlation between their predictor vectors. The responses y_i, $i = 1,...,8$ are sampled from $Bernoulli[\pi(X_{i\cdot}; \boldsymbol{\beta})]$, where the probabilities π are defined as in eq. (2.8), i.e. $\pi(\mathbf{x}; \boldsymbol{\beta}) := \frac{\exp[\mathbf{x}^T\boldsymbol{\beta}]}{1+\exp[\mathbf{x}^T\boldsymbol{\beta}]}$ and the coefficients vector is $\boldsymbol{\beta} := (3, 3, -0.5, 0)$[2]. In the same way, we simulate test data $[X^{test}|\mathbf{y}^{test}]$, $X^{test} \in \mathbb{R}^{500\times 4}$, $\mathbf{y}^{test} \in \{0,1\}^{500}$. In this simulation, we know that the data originated from an LLR model with known coefficients $\boldsymbol{\beta}$ and, therefore, we know the Bayes classifier and can use $[X^{test}|\mathbf{y}^{test}]$ to estimate the Bayes risk (see eq. 2.4):

$$\mathbb{P}[\mathcal{C}_{Bayes}(X_{new}) \neq Y_{new}] = 0.075. \tag{2.19}$$

[2]We assume a 0 intercept and don't include it in the notation

Throughout this section, we examine different ways to fit the LLR coefficients $\boldsymbol{\beta}$ from the training data $[X|\mathbf{y}]$, and compare the obtained estimates, $\hat{\boldsymbol{\beta}}$, with respect to criteria 1 and 2. A good fit would recover the irrelevance of the predictors $X_{\cdot 3}$ and $X_{\cdot 4}$ and would pinpoint $X_{\cdot 1}$and $X_{\cdot 2}$ as relevant features, by setting their corresponding coefficients, $\hat{\beta}_1$ and $\hat{\beta}_2$ to close non-zero values.

The maximum likelihood estimate is obtained by minimizing the negative log-likelihood function. Figure 2.1 a) shows the contours of the negative log-likelihood over the plane (β_1, β_2) for the fixed estimated values of β_3 and β_4. Due to the small number of training data-points, the negative log-likelihood surface is almost flat in a large area surrounding its minimum. This causes high variance of the estimates $\hat{\boldsymbol{\beta}}$ as a function of the training data and poor generalization performance on new data.

One way to deal with this problem is to introduce additional information in the model fitting procedure, in order to prevent the "overfitting" effect, by reducing the dependence of the estimate $\hat{\boldsymbol{\beta}}$ on the training data. This approach, called regularization, is usually accomplished by incorporating a prior distribution of the coefficients $\boldsymbol{\beta}$. Assuming a known prior distribution $\mathbb{P}[\boldsymbol{\beta}]$, the Bayes theorem gives a formula for the posterior distribution of $\boldsymbol{\beta}$:

$$\underbrace{\mathbb{P}\left[\boldsymbol{\beta}|\, X, \mathbf{y}\right]}_{Posterior} = \frac{\overbrace{\mathbb{P}\left[X, \mathbf{y}|\boldsymbol{\beta}\right]}^{Likelihood}\overbrace{\mathbb{P}[\boldsymbol{\beta}]}^{Prior}}{\underbrace{\mathbb{P}\left[X, \mathbf{y}\right]}_{Evidence}} \tag{2.20}$$

The maximum a-posteriori (MAP) estimate of $\boldsymbol{\beta}$, defined as

$$\hat{\boldsymbol{\beta}}_{MAP}: = \arg\max_{\boldsymbol{\beta}} \mathbb{P}\left[\boldsymbol{\beta}|\, X, \mathbf{y}\right], \tag{2.21}$$

$$= \arg\min_{\boldsymbol{\beta}} \left(-\ell_{cond}(\boldsymbol{\beta}; [X|\mathbf{y}]) + \underbrace{(-\log(\mathbb{P}[\boldsymbol{\beta}]))}_{penalty} \right) \tag{2.22}$$

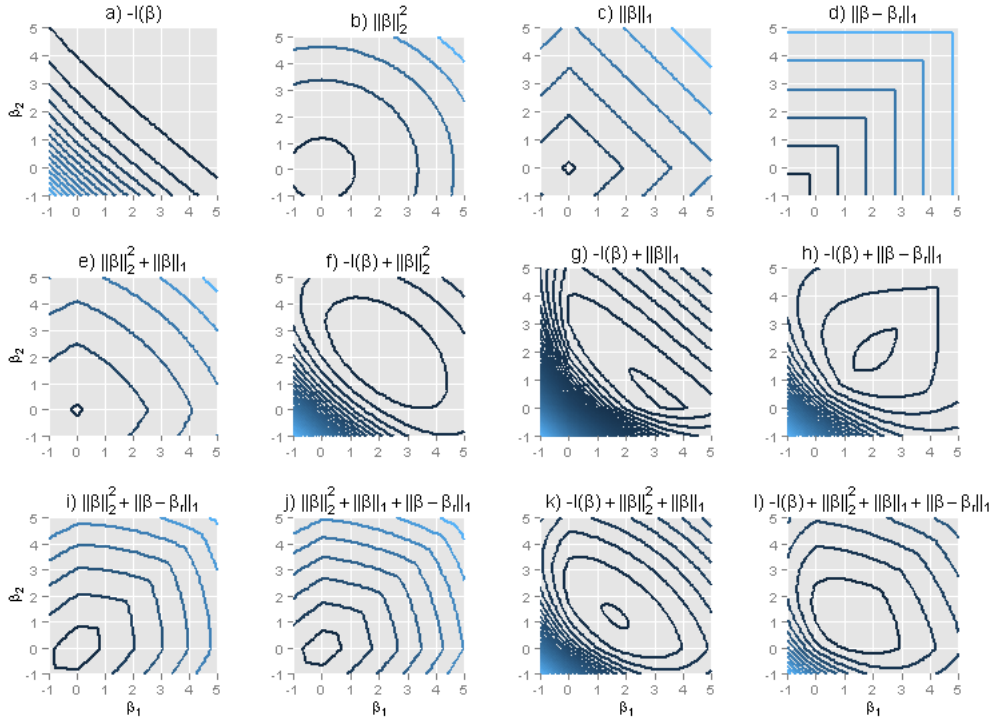
can be estimated from the data, by neglecting the unknown constant term $\mathbb{P}\left[X, \mathbf{y}\right]$.

Table 2.1 summarizes some commonly used regularizing priors and Figure 2.1 illustrates their effect on the resulting negative log-posterior in the toy-example.

Table 2.1: Common regularizing priors and their effect on the estimated coefficients $\hat{\beta}$

Prior ($\mathbb{P}[\boldsymbol{\beta}]$)	Penalty ($-\log(\mathbb{P}[\boldsymbol{\beta}])$)	Properties
Gaussian: $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$; $p(\boldsymbol{\beta}) \propto \exp\left(-\frac{\boldsymbol{\beta}^T \boldsymbol{\beta}}{2\sigma^2}\right)$ parameter: $\lambda_2 := \frac{1}{\sigma^2}$, $\lambda_2 > 0$.	Ridge (L2): $\frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$ Figure 2.1b,f.	Continuous shrinkage of $\boldsymbol{\beta}$ towards $\mathbf{0}$ [Zou and Hastie, 2005]; Doesn't select variables (keeps all coefficients)
Laplace: $\boldsymbol{\beta} \sim Lap(\mathbf{0}, \tau I)$; $p(\boldsymbol{\beta}) \propto \exp\left(-\frac{\|\boldsymbol{\beta}\|_1}{\tau}\right)$ parameter: $\lambda_1 := \frac{1}{\tau}$, $\lambda_1 > 0$	Lasso (L1): $\lambda_1 \|\boldsymbol{\beta}\|_1$ Figure 2.1c,g.	Continuous shrinkage of $\boldsymbol{\beta}$; Selects variables, by setting coefficients to 0; In the case $d > n$, selects at most n variables [Zou and Hastie, 2005]; Tends to select only one arbitrary variable from a group of highly correlated variables [Zou and Hastie, 2005] Performance dominated by Ridge in the case $d > n$ [Zou and Hastie, 2005].
Gaussian×Laplace: $\boldsymbol{\beta} \sim \frac{1}{Z}\mathcal{N}(\mathbf{0}, \sigma^2 I) Lap(\mathbf{0}, \tau I)$; $p(\boldsymbol{\beta}) \propto \exp\left(-\frac{\|\boldsymbol{\beta}\|_1}{\tau}\right)$ parameters: $\lambda_1 := \frac{1}{\tau}$, $\lambda_2 := \frac{1}{\sigma^2}$, $\lambda_1, \lambda_2 > 0$.	Elastic net (L1+L2): $\lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$ Figure 2.1e,k.	Continuous shrinkage of $\boldsymbol{\beta}$; Selects groups of correlated variables [Zou and Hastie, 2005]; Similar performance as Ridge in all cases [Zou and Hastie, 2005];
Fusing Laplace: $(\boldsymbol{\beta} - \boldsymbol{\beta}_r) \sim Lap(\mathbf{0}, \tau I)$, where $\boldsymbol{\beta}_r := (\beta_2, \beta_3, ..., \beta_n, \beta_1)$ $p(\boldsymbol{\beta} - \boldsymbol{\beta}_r) \propto \exp\left(-\frac{\|\boldsymbol{\beta} - \boldsymbol{\beta}_r\|_1}{\tau}\right)$ parameter: $\nu := \frac{1}{\tau}$, $\nu > 0$.	Fused lasso: $\nu \|\boldsymbol{\beta} - \boldsymbol{\beta}_r\|_1$ Figure 2.1d,h,i,j,l.	Usefull when the variables are ordered with expected proximity between coefficients for neighboring variables Tibshirani et al. [2005]; "Pulls" $\boldsymbol{\beta}$ towards the identity line, so that the coefficients become close to each other and often equal, without being shrinked to $\mathbf{0}$.

Figure 2.1: Equi-height contours of the (penalized) negative log-likelihood over the plane (β_1, β_2) for fixed values of β_3 and β_4.



All models have been trained on the same set of 8 training points. The used penalizing parameters λ_1, λ_2 and ν have been chosen with the aim to make the effect of the penalties clearly visible, without aiming at the optimal prediction error-values. The coefficients β_3 and β_4 are fixed to their optimal estimates, based on evaluation of the models on a discretized subset of the hypercube $[-1, 5]^4$. The height-difference between two contours is fixed, so that denser positioned contours denote a steeper slope. Brighter blue color denotes a higher value. a) unpenalized negative log-likelihood; b) ridge c) lasso; d) fusing L1 penalty; e) elastic net (ridge+lasso); f) negative log-likelihood with a ridge penalty; g) negative log-likelihood with a lasso penalty; h) negative log-likelihood with a fusing L1-penalty (note the pointed contour lines at identity line); i) fused ridge penalty; j) fused elastic net penalty; k) elastic net penalized negative log-likelihood; l) fused elastic net penalized negative log-likelihood;

For the toy-example, the minimal expected zero-one error on the test-set $[X_{test}|Y_{test}]$ has been reached using the fused elastic net penalty with penalizing parameters $\lambda_1 = 0.3$, $\lambda_2 = 0.1$, $\nu = 0.2$ and was equal to 0.9 (fig. Figure 2.1 l), compared to 0.14 for the unpenalized log-likelihood estimate (fig. Figure 2.1 a), 0.12 for the ridge-and lasso-penalized estimates (fig.

Figure 2.1 f,g) and 0.11 for the elastic-net penalized estimate (fig. Figure 2.1 k). The optimal estimate of the coefficients for the fused elastic-net penalty was $\hat{\boldsymbol{\beta}} = (1.25, 1.25, 0, -0.25)$. Apart from the power of regularization in preventing overfitting, this example demonstrates the benefit from imposing a fused penalty in the case when there is a prior belief about proximity of coefficients associated with neighboring features. In section 2.2 we show how we adapt this regularizing prior to the case of ordered multi-task logistic regression.

2.1.4 Convex optimization for Classification

A major challenge in statistical modeling is to define a model that, on the one hand, is well adaptable to the phenomenon of study, and on the other hand, can be fit to the training data in an efficient way. In the case of model-fitting via likelihood or posterior maximization, the fitting procedure reduces to an optimization problem. One reason why linear models like linear regression, LLR, LDA and their L2- and L1- regularized variants have gained popularity is the fact that fitting these models to the training data results in having to optimize a convex function. This section briefly describes two methods for convex optimization that we will use for fitting models considered in this thesis. Specifically, these comprise the Newton-Raphson's method for optimization of twice-differentiable functions, and the Alternating Direction Method of Multipliers (ADMM) [Boyd et al., 2011], which is suitable for decomposable objectives of the form $f(x) + g(x)$. In section 2.2, we show how we use these two methods for fitting the FENLR model.

The Newton-Raphson's method for differentiable functions

The ridge-penalized negative log-likelihood from eq. (2.16) is convex and twice differentiable. Therefore, finding its global minimum can be done iteratively with quadratic rate of convergence using the Newton-Raphson's method. Starting from an initial guess $\mathbf{a}$, this method makes successive approximations to the root of a differentiable function $f(\mathbf{x})$ by the use of its first order Taylor approximation

$$f(\mathbf{x}) \approx f(\mathbf{a}) + \nabla f(\mathbf{a})(\mathbf{x} - \mathbf{a}).$$

By setting the left hand side to 0 and solving for $\mathbf{x}$, one obtains an approximation $\hat{\mathbf{x}}_0$ for the root $\mathbf{x}_0$ of f. Iterating over this step, by replacing $\mathbf{a}$ with the current estimate $\hat{\mathbf{x}}_0$, quickly approaches a root of f. Minimizing a convex twice-differentiable function reduces to finding a root of its gradient.

Algorithm 2.1 is a pseudo-code of the Newton-Raphson's method in this case:

Algorithm 2.1 Newton-Raphson's method for optimizing a twice-differentiable function

Input: twice-differentiable function $f : \mathbb{R}^d \to \mathbb{R}$, $\mathbf{x}_0 \in \mathbb{R}^d$, $k = 0$
do {
  $\mathbf{x}_{k+1} := \mathbf{x}_k - (\nabla\nabla f(\mathbf{x}_k))^{-1} \nabla f(\mathbf{x}_k)$
  $k := k + 1$
} until convergence

In practice, the Newton-Raphson's method can achieve convergence to very high precision within 10 iterations. However, in the case $d \gg 0$, the inversion of the $d \times d$-dimensional Hessian matrix in each iteration can turn into a serious performance bottleneck. Using the conventional method "solve" in R, on a computer with 64 bit-3.1GHz Intel™(Core™) i7-processor, the inversion of the Hessian matrix for $d = 5000$ takes $\sim 160s$, compared to $\sim 1s$ for $d = 1000$, and $\sim 0.15s$ for $d = 500$. If we ignore the costs for calculating the gradient and the hessian, with $d = 5000$, a full 10 iteration Newton-Raphson's execution takes approximately 26 minutes. As we will see in the next sections, we are about to use the Newton-Raphson's procedure as a localized optimization step, which makes part of a larger optimization algorithm, therefore, accounting to numerous (possibly tens of thousands) Newton-Raphson's executions. An elegant approach to avoid the computational cost in this case is shown in section 2.2.

The ADMM method

When there is a penalty-term on the L1 norm of its argument, the objective function is no more differentiable and it is impossible to use the Newton-Raphson's method. Yet, if the objective function of the optimization problem is still convex, there exist other efficient methods to solve it.

The Alternating Direction Method of Multipliers (ADMM) [Boyd et al., 2011] is an iterative optimization algorithm, which provides a framework for solving constrained optimization problems of the form:

$$\begin{aligned} \min \quad & f(\boldsymbol{\chi}) + g(\boldsymbol{\zeta}) \\ \text{subject to } & P\boldsymbol{\chi} + Q\boldsymbol{\zeta} = \mathbf{s} \end{aligned} \tag{2.23}$$

with variables $\chi \in \mathbb{R}^n$ and $\zeta \in \mathbb{R}^m$, where $P \in \mathbb{R}^{p \times n}$, $Q \in \mathbb{R}^{p \times m}$ and $\mathbf{s} \in \mathbb{R}^p$.[3]One important property of ADMM, accounting for its broad applicability, is that in order to guarantee convergence, it makes relatively loose assumptions on the functions f and g. For instance, f and g need not to be differentiable, and are allowed to accept the value $+\infty$. In essence, ADMM can be applied to any equality constrained convex optimization problem of the general form

$$\begin{aligned} &\min\ F(\xi) \\ &\text{subject to } A\xi = \mathbf{c} \end{aligned} \tag{2.24}$$

provided that it can be presented in the decomposed form (2.23) and the following two assumptions hold [Boyd et al., 2011]:

Assumption 1. The (extended-real-valued) functions $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ are closed, proper and convex.

Assumption 2. The unaugmented Lagrangian, defined as the function

$$L_0(\chi, \zeta, \omega) := f(\chi) + g(\zeta) + \omega^T(P\chi + Q\zeta - \mathbf{s}),$$

where $\omega \in \mathbb{R}^p$ are the Lagrange multipliers, has a saddle point.

Without going into details, the first assumption ensures that we are dealing with well behaved convex functions in the objective (see The ADMM method) Boyd et al. [2011]. The second assumption sheds light on the underlying principle of ADMM and its precursors, dual ascent and the method of multipliers [Boyd et al., 2011, p.5-10]. Denoting $\theta := (\chi, \zeta)$, the dual function for the unaugmented Lagrangian L_0 is defined as

$$h(\omega) := \inf_{\theta} L_0(\theta, \omega),$$

and the dual problem to problem (2.23) is defined as

$$\max\ h(\omega). \tag{2.25}$$

A thorough study of the relations between the primal problem (2.23) and the dual problem (2.25), and (2.25), has brought up conditions under which their optimal values would match, and it would be possible to obtain the optimum θ^* from an optimal dual point ω^*, by solving

$$\theta^* := \arg\min_{\theta} L_0(\theta, \omega^*). \tag{2.26}$$

[3]Here we use a slightly modified notation from the original paper Boyd et al. [2011] with the Greek analogs of the the letters 'x' and 'z' in order to avoid the conflict with the name 'x' for predictor variables.

The joint optimum $(\boldsymbol{\theta}^*, \boldsymbol{\omega}^*)$ would then be a saddle point for the unaugmented Lagrangian, thus, justifying the requirement stated in Assumption 2. In many cases, it is possible to find $\boldsymbol{\omega}^*$ in a "dual ascent" procedure, which iteratively solves a localized version $\boldsymbol{\theta}^{k+1} := \arg\min_{\boldsymbol{\theta}} L_0(\boldsymbol{\theta}, \boldsymbol{\omega}^k)$ of problem (2.26) and finds a new candidate $\boldsymbol{\omega}^{k+1}$ closer to the optimal $\boldsymbol{\omega}^*$, until satisfying a convergence criterion. Solving (2.26) is done via another known optimization technique. A further improvement to the idea of iterative convergence to the optimal dual variable $\boldsymbol{\omega}^*$ is accomplished by the addition of a penalty term, $\frac{1}{2}\rho||P\boldsymbol{\chi} + Q\boldsymbol{\zeta} - \mathbf{s}||_2^2$ to L_0, where $\rho > 0$ is a penalizing parameter. This results in the direct precursor of ADMM, the method of multipliers, which replaces L_0 by the augmented Lagrangian:

$$L_\rho(\boldsymbol{\chi}, \boldsymbol{\zeta}, \mathbf{w}) := f(\boldsymbol{\chi}) + g(\boldsymbol{\zeta}) + \boldsymbol{\omega}^T(P\boldsymbol{\chi} + Q\boldsymbol{\zeta} - \mathbf{s}) + \frac{1}{2}\rho||P\boldsymbol{\chi} + Q\boldsymbol{\zeta} - \mathbf{s}||_2^2.$$

In essence, using the augmented Lagrangian, L_ρ instead of L_0, improves the robustness and ensures the convergence under less stringent assumptions for the objective $f + g$. For example, the dual ascent method necessitates that there is exactly one minimizer to $L_0(\boldsymbol{\theta}, \boldsymbol{\omega}^*)$, which can be ensured by strict convexity of $f + g$. The method of multipliers, and consecutively, ADMM doesn't require strict convexity of its objective. This property makes it a good candidate for solving problems with L1-norm terms on the coefficients, which are not strictly convex in the case $d \gg n$ [Tibshirani, 2013].

With the scaled dual variable $\boldsymbol{\xi} := \boldsymbol{\omega}/\rho$, Algorithm 2.2 lists the general scaled form of ADMM [Boyd et al., 2011].

Algorithm 2.2 ADMM (general scaled form)

Initialization: $\boldsymbol{\chi}^0 = \boldsymbol{\zeta}^0 = \boldsymbol{\xi}^0 = \mathbf{0}$; $k = 0$
do {
$\boldsymbol{\chi}$-update:$\boldsymbol{\chi}^{k+1} := \arg\min_{\boldsymbol{\chi}} \left(f(\boldsymbol{\chi}) + \frac{1}{2}\rho||P\boldsymbol{\chi} + Q\boldsymbol{\zeta}^k - \mathbf{s} + \boldsymbol{\xi}^k||_2^2 \right)$
$\boldsymbol{\zeta}$-update: $\boldsymbol{\zeta}^{k+1} := \arg\min_{\boldsymbol{\zeta}} \left(g(\boldsymbol{\zeta}) + \frac{1}{2}\rho||P\boldsymbol{\chi}^{k+1} + Q\boldsymbol{\zeta} - \mathbf{s} + \boldsymbol{\xi}^k||_2^2 \right)$
$\boldsymbol{\xi}$-update: $\boldsymbol{\xi}^{k+1} := \boldsymbol{\xi}^k + P\boldsymbol{\chi}^{k+1} + Q\boldsymbol{\zeta}^{k+1} - \mathbf{s}$
$k = k + 1$
} while($k < MAXITER$ and not converged)

ADMM owes its name to its direct precursor, the method of multipliers, and the alternating way in which it optimizes the localized augmented Lagrangian. As we saw, ADMM works at a higher level of abstraction, compared to other

optimization algorithms like Newton-Raphson. While the Newton-Raphson's gradient descent algorithm iterates over basic calculus operations, such as the calculation of a gradient vector and a hessian matrix, the iterative operations in ADMM are small localized optimizations tasks, which are orchestrated in a way, that, under some assumptions, leads to the optimum of a large global problem. The localized optimization tasks can usually be solved by lower-level optimization procedures, or even analytically.

2.1.5 A brief overview of multi-task learning

Any multi-task learning (MTL) procedure aims at improving the performance of some set of individual learning tasks by using additional information, encoded in the relatedness between these tasks. Examples of tasks that could be approached from the MTL point of view are (i) inference of clinical scores at different time-points in modeling disease progression [Zhou et al., 2013], (ii) recognition of spam e-mails in different demographic groups [Attenberg et al., 2009], (iii) identification of host-pathogen protein interactions in different infectious diseases [Kshirsagar et al., 2013], (iv) modeling of marketing preferences of similar social groups [Evgeniou and Micchelli, 2005] and many others. A more detailed view of the examples above suggests that most of the MTL problems fall in one of the two categories:

1. Multiple similar learning problems share the same data. This is the case for example (i), where the training data for all tasks has the same design matrix corresponding to an initial state of the diseased patient, while the response vectors for every task corresponds to different time-points in the progression of the disease.

2. A single learning problem has to be solved in multiple similar data-domains. This is the case for examples (ii), (iii) and (iv). In each of them, the training examples come from multiple data-sources with similar or identical sets of features.

A challenge that is present for problems in both categories is the incorporation of the task relatedness topology in the MTL model. A straightforward way to express task relatedness is to produce a graph with weighted edges, in which every node is associated with an individual task and the weight on each edge quantifies the estimated pairwise relatedness between its corresponding task-nodes. Let $R \in (\{0\} \cup \mathbb{R}_+)^{t \times t}$ be the adjacency matrix corresponding to the task relatedness topology of t individual tasks. Let $B := [\boldsymbol{\beta}^{(1)}, ..., \boldsymbol{\beta}^{(t)}] \in \mathbb{R}^{(1+d) \times t}$ be the coefficient matrix for these tasks and let $\mathcal{L}_k(\boldsymbol{\beta}^{(k)})$ be the

loss function associated with each task, for example $\mathcal{L}_k$ can be taken to be the negative log-likelihood. One way to learn the tasks simultaneously by encouraging similarity between related tasks is to solve [Zhou et al., 2011]:

$$\hat{B} := \arg\min_{B} \sum_{k=1}^{t} \mathcal{L}_k(\boldsymbol{\beta}^{(k)}) + ||BR||_F^2, \tag{2.27}$$

where $||\cdot||_F$ is the L2 (Frobenius) matrix norm. As the added penalty in the above optimization task is twice differentiable, the computational complexity would in most cases be dominated by the sum of the individual task's loss functions.

In subsection 2.1.3, we saw that a fused lasso penalty can "pull" consecutive coefficients in a single-task model towards the identity line, so that they become close to each other and often equal. This variable "fusion" approach, which has been introduced by Land and Friedman [1996] and later popularized by Tibshirani et al. [2005], can be transferred to the domain of multi-task learning in order to induce sparsity in the difference of coefficient vectors associated with pairwise related tasks. Specifically, one can replace the Frobenius norm with the L1 norm in eq. (2.27) to obtain a fused lasso penalized MTL fit in the form

$$\hat{B} := \arg\min_{B} \sum_{k=1}^{t} \mathcal{L}_k(\boldsymbol{\beta}^{(k)}) + ||BR||_1. \tag{2.28}$$

The optimization problem (2.28) is hard to solve in general, due to the non-differentiability and possible not strict convexity of the L1 penalty term. However, efficient solutions exist in the more specific case of acyclic hierarchical topology of the task relatedness, like linear ordering in the case of modeling consecutive time-points in a disease-progression [Zhou et al., 2013] or phylogenetic tree of evolutionary close species [Widmer, 2012]. Another example, in which problem (2.28) can be solved efficiently is the FENLR model described in (section 2.2) where the tasks are linearly and cyclically ordered with L1-penalized absolute difference between neighboring tasks.

An additional challenge, that occurs in MTL problems of the second category, is to unify the feature-sets between the data-domains associated with different tasks. For instance, in a genomic study of temporal disease dynamics, where different tasks are associated with microarray data from different species, it might be challenging to produce a homology mapping between the genes of the different species.

In any case an MTL approach to solve a set of individual learning tasks would introduce an additional level of complexity to the model and, therefore, it is important to predict in advance to what extent would an MTL solution improve the individual-task's generalization performance. Widmer [2012] provides guide lines on deciding whether an MTL approach would be beneficial, depending on the observed task similarity and the saturation of the learning curve as function of the amount of training examples for each individual task.

2.2 The fused elastic net logistic regression (FENLR) method for ordered binary classification

We consider a set of t ordered binary classification tasks, $1, ..., t$, on a set of n d-dimensional labeled training observations $\{\mathbf{x}_1, ..., \mathbf{x}_n\} \subset \mathbb{R}^{1+d}$, with $x_{i1} = 1,\ i = 1, .., n$. The order of the tasks reflects their similarity. For instance, neighboring tasks should be more likely to assign the same label to a test observation, compared to tasks that are ordered far from each other. The training data for all tasks is encoded in the matrix $[X|Y]$, where $X = [\mathbf{x}_1, ..., \mathbf{x}_n]^T \in \mathbb{R}^{n\times(1+d)}$ is the common design matrix shared by all tasks[4], and the response vector for task j is written as the column-vector $\mathbf{y}_j = Y_{\cdot j}$ of the matrix $Y = [\mathbf{y}_1, ..., \mathbf{y}_t] \in \{-1, 1\}^{n\times t}$, $j = 1, ..., t$. We define a single-task LLR model for task $j = 1, ..., t$ with training data $[X|\mathbf{y}_j]$ as:

$$logit(\pi^{(j)}(\mathbf{x})) \approx \mathbf{x}^T\boldsymbol{\beta}^{(j)} =: g^{(j)}(\mathbf{x}), \tag{2.29}$$

where $\pi^{(j)}(\mathbf{x}_i) := \mathbb{P}[Y_{ij} = 1|\mathbf{x}]$ is the probability that the true label of the observation $\mathbf{x}$ is 1, and $\boldsymbol{\beta}^{(j)} \in \mathbb{R}^{1+d}$ are the LLR coefficients. The negative log-likelihood is defined in the same way as in eq. (2.15):

$$-\ell^{(j)}(\boldsymbol{\beta}^{(j)}; [X|\mathbf{y}_j]) = \sum \log\left(\mathbf{1} + \exp(-\mathbf{y}_j \odot X\boldsymbol{\beta}^{(j)})\right), \quad j = 1, ..., t. \tag{2.30}$$

As we saw in the introduction section, minimizing an L1-L2-penalized version of the negative log-likelihood leads to sparse solutions keeping non-zero coefficients for the relevant sets of correlated feature-vectors. This idea reduces to a single-task fitting procedure, in which we find the L1-L2 penalized estimate of the coefficients by consecutively solving the optimization problems

$$\boldsymbol{\beta}^{(j)*} := \arg\min_{\boldsymbol{\beta}^{(j)}\in\mathbb{R}^{(1+d)}} \left\{-\ell^{(j)}(\boldsymbol{\beta}^{(j)}; [X|\mathbf{y}_j]) + ||\boldsymbol{\lambda}_1 \odot \boldsymbol{\beta}^{(j)}||_1 + \frac{1}{2}||\boldsymbol{\lambda}_2 \odot \boldsymbol{\beta}^{(j)}||_2^2\right\} \tag{2.31}$$

[4] Assume that the first column of the design matrix X is the constant vector $\mathbf{1}$.

for $j = 1, .., t$. A small detail of this formulation is that we have presented the regularizing parameters $\lambda_1 > 0$ and $\lambda_2 > 0$ as real vectors of the form $\boldsymbol{\lambda}_1 = (0, \lambda_1, ..., \lambda_1) \in \mathbb{R}^{1+d}$ and $\boldsymbol{\lambda}_2 = (0, \lambda_2, ..., \lambda_2) \in \mathbb{R}^{1+d}$, in order to account for the usually unpenalized intercept $\beta_0^{(j)}$.

Now we wish to incorporate the prior knowledge about the similarity between neighboring tasks into the model-fitting procedure. An important observation, which directly follows from the continuity of the modeling function in 2.8, is that two LLR-models operating on the same data would produce similar output if their coefficients were similar. Therefore, similarity between neighboring LLR models for neighboring tasks can be encoded by penalizing the difference between their coefficients. Let $B := [\boldsymbol{\beta}^{(1)}, ..., \boldsymbol{\beta}^{(t)}] \in \mathbb{R}^{(1+d)\times t}$ be the coefficient matrix for all tasks and let $R \in \mathbb{R}^{t\times t}$ be a matrix defined in the following way:

$$R_{ij} := \begin{cases} 1 & \text{if } j = i - 1 \text{ or } (i, j) = (1, t) \\ 0 & \text{otherwise} \end{cases}, \ i, j = 1, ..., t.$$

We call R the column-rotating matrix for B, because the columns of the $(1 + d) \times t$-matrix BR are the same as the columns of B, but rotated by 1 column to the left, in other words, $BR = [\boldsymbol{\beta}^{(2)}, \boldsymbol{\beta}^{(3)} ..., \boldsymbol{\beta}^{(1)}] \in \mathbb{R}^{(1+d)\times t}$. Let $\nu \geq 0$ be a penalizing parameter. Denote by $[\cdot]$ the $(1 + d) \times t$-matrix with all columns equal to a vector $\cdot$, by $[\nu]$ the $(1 + d) \times t$-matrix, each element of which is equal to ν, and by I the $t \times t$-dimensional identity matrix. We define the multi-task fused L1-L2-penalized negative log-likelihood as the function:

$$\begin{aligned} -\ell^{MT}(B; [X|Y]) \ &:= \ -\sum_{j=1}^{t} \ell^{(j)}(\boldsymbol{\beta}^{(j)}; [X|\mathbf{y}_j]) && (2.32) \\ &\quad + \sum_{j=1}^{t} \left(||\boldsymbol{\lambda}_1 \odot \boldsymbol{\beta}^{(j)}||_1 + \frac{1}{2}||\boldsymbol{\lambda}_2 \odot \boldsymbol{\beta}^{(j)}||_2^2 \right) && (2.33) \\ &\quad + ||[\nu] \odot B(I - R)||_1 && (2.34) \\ &= \ \sum \log\Big([1] + \exp(-Y \odot XB) \Big) && (2.35) \\ &\quad + ||[\boldsymbol{\lambda}_1] \odot B||_1 + \frac{1}{2}||[\boldsymbol{\lambda}_2] \odot B||_2^2 && (2.36) \\ &\quad + ||[\nu] \odot B(I - R)||_1 && (2.37) \end{aligned}$$

The first formulation shows that if the penalizing parameter ν is set to 0, the optimizing ℓ^{MT} with respect to B can be split across the columns of B, and

is equivalent to the single-task optimization with elastic net penalty (2.31). The fusing L1 penalty (2.37) represents a scaled sum of absolute differences between each pair of consecutive columns of B and cannot be decomposed column-wise. The maximum likelihood fit of the parameters B to the training data $[X|Y]$ is found by solving the optimization problem

$$B^* = \arg\min_{B \in \mathbb{R}^{(1+d)\times t}} -\ell^{MT}(B; [X|Y]). \tag{2.38}$$

As a sum of convex functions, the function ℓ^{MT} is also convex but, due to the presence of L1-terms, it is not differentiable and, therefore, not solvable by gradient descent methods. Through the rest of this section, we show one way to solve this problem numerically by adapting it to the modular framework of the ADMM algorithm described in section section 2.1.4.

To begin, we convert problem (2.38) to the canonical ADMM-form (2.23) by introducing the variable matrices $\chi \in \mathbb{R}^{(1+d)\times t}$ and $\zeta \in \mathbb{R}^{(1+d)\times t}$, and separating the differentiable from the non-differentiable terms:

$$\begin{aligned} \min \quad & \underbrace{\sum \log\Big([1] + \exp(-Y \odot X\chi)\Big) + \frac{1}{2}|||[\boldsymbol{\lambda}_2] \odot \chi||_2^2}_{=: f(\chi)} \\ & + \\ & \underbrace{|||[\boldsymbol{\lambda}_1] \odot \zeta||_1 + |||[\nu] \odot \zeta(I - R)||_1}_{=: g(\zeta)} \\ \text{subject to} \quad & \chi - \zeta = [0] \end{aligned} \tag{2.39}$$

Alternatively, it is possible to decompose problem (2.38) into a constrained sum of three functions, $\tilde{f}(\underset{\circ}{\chi}) + \tilde{g}(Y) + \tilde{h}(\overset{\circ}{\zeta})$ of the augmented variable matri-

ces $\underset{\circ}{\chi} := \begin{bmatrix} \chi^T \\ 0_{t\times(1+d)} \end{bmatrix}_{2t\times(1+d)}$, $\Upsilon := \begin{bmatrix} \zeta^T \\ \zeta^T \end{bmatrix}_{2t\times(1+d)}$ and $\overset{\circ}{\zeta} := \begin{bmatrix} 0 \\ (I - R^T)\zeta^T \end{bmatrix}_{2t\times(1+d)}$:

$$\begin{aligned} \min \quad & \underbrace{\sum \log\Big([1] + \exp(-Y \odot X\underset{\circ}{\chi}^T)\Big) + \left\| \frac{1}{2} \begin{bmatrix} [\lambda_2] \\ 0_{(1+d)\times t} \end{bmatrix} \odot \underset{\circ}{\chi} \right\|_2^2}_{=:\tilde{f}\left(\underset{\circ}{\chi}\right)} \\ & + \\ & \underbrace{\left\| \begin{bmatrix} [\lambda_1]^T \\ [\lambda_1]^T \end{bmatrix} \odot \Upsilon \right\|_1}_{=:\tilde{g}(\Upsilon)} + \underbrace{\left\| \begin{bmatrix} 0_{t\times(1+d)} \\ [\nu]^T \end{bmatrix} \odot \overset{\circ}{\zeta} \right\|_1}_{=:\tilde{h}\left(\overset{\circ}{\zeta}\right)} \end{aligned} \tag{2.40}$$

$$\text{subject to} \quad \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \underset{\circ}{\chi} + \begin{bmatrix} -I & 0 \\ 0 & -I \end{bmatrix} \Upsilon + \begin{bmatrix} 0 & 0 \\ 0 & (I_{t\times t} - R^T)^{-1} \end{bmatrix} \overset{\circ}{\zeta} = \begin{bmatrix} 0_{t\times(1+d)} \\ 0_{t\times(1+d)} \end{bmatrix}.$$

Due to time constraints, an ADMM implementation, based on the formulation in (2.40) has not been considered. However, it might be interesting to implement it in the future, because it would reduce the recursive nesting of iterative optimization procedures (see Second level ADMM for the ζ-update).

The scaled form of the ADMM algorithm for problem (2.39) is given in (2.2):

Algorithm 2.3 ADMM for ℓ^{MT}

Initialization: $\chi^0 = \zeta^0 = \xi^0 = [0]_{(1+d)\times t}$;$k := 0$
do {
χ-update:$\chi^{k+1} := \arg\min_\chi \left(f(\chi) + \frac{1}{2}\rho||\chi - \zeta^k + \xi^k||_2^2\right)$
ζ-update: $\zeta^{k+1} := \arg\min_\zeta \left(g(\zeta) + \frac{1}{2}\rho||\chi^{k+1} - \zeta + \xi^k||_2^2\right)$
ξ-update: $\xi^{k+1} := \xi^k + \chi^{k+1} - \zeta^{k+1}$
$k := k + 1$
} while($k < MAXITER$ and not converged)

The convergence criterion is straightforward to implement, following the instructions in Boyd et al. [2011, p. 16-17].

In the next two subsections, we describe the χ-update and the ζ-update.

Newton-Raphson gradient descent procedure for the χ-update

For the χ-update, we notice that there is no coupling between the columns of the variable matrix χ. Therefore, it is computationally more convenient to obtain χ^{k+1} by solving separately for $j = 1, ..., t$:

$$\chi_{\cdot j}^{k+1} := \arg\min_{\chi_{\cdot j}} \underbrace{\left\{ \log\Big(\mathbf{1} + \exp(-Y_{\cdot j} \odot X\chi_{\cdot j})\Big) + \frac{1}{2}||\boldsymbol{\lambda}_2 \odot \chi_{\cdot j}||_2^2 + \frac{1}{2}\rho||\chi_{\cdot j} - \Omega_{\cdot j}^k||_2^2 \right\}}_{\tilde{f}^{k(j)}(\chi_{\cdot j})}. \tag{2.41}$$

To shorten the expressions, in the above equation, we substitute the constant $\zeta_{\cdot j}^k - \xi_{\cdot j}^k$, by the symbol $\Omega_{\cdot j}^k$.

The function $\tilde{f}^{k(j)}(\chi_{\cdot j})$ is twice differentiable and convex and, therefore, can be optimized efficiently using the Newton-Raphson's method. By means of vector calculus, we find analytical expressions for the gradient and hessian of $\tilde{f}^{k(j)}$:

$$\mathbb{R}^{1+d} \ni \nabla \tilde{f}^{k(j)}(\chi_{\cdot j}) \quad = \quad X^T \delta(\chi_{\cdot j}) + \eta(\chi_{\cdot j}) \tag{2.42}$$

$$\mathbb{R}^{(1+d)\times(1+d)} \ni \nabla\nabla \tilde{f}^{(j)}(\chi_{\cdot j}) = ([\mathbf{w}] \odot X)^T ([\mathbf{w}] \odot X) + (\boldsymbol{\lambda}_2 + \boldsymbol{\rho})^T I, \tag{2.43}$$

where

$$\mathbb{R}^n \ni \delta(\chi_{\cdot j}) := \left[-Y_{\cdot j} \odot \exp(-Y_{\cdot j} \odot X\chi_{\cdot j})\right] \div \left[\mathbf{1} + \exp(-Y_{\cdot j} \odot X\chi_{\cdot j})\right], \tag{2.44}$$

$$\mathbb{R}^{1+d} \ni \eta(\chi_{\cdot j}) := (\boldsymbol{\lambda}_2 + \boldsymbol{\rho}) \odot \chi_{\cdot j} - \rho\Omega_{\cdot j}^k, \tag{2.45}$$

$$\mathbb{R}^n \ni \mathbf{w} := \sqrt{\exp(-Y_{\cdot j} \odot X\chi_{\cdot j})} \div [\exp(-Y_{\cdot j} \odot X\chi_{\cdot j})]. \tag{2.46}$$

The symbol $'\div'$ denotes element-wise division between its vector operands and I denotes the identity matrix. Now, all that remains is to insert the expressions for the gradient and the hessian into Algorithm 2.1.

We noticed in section 2.1.4 that for large number of covariates, d, the inversion of the hessian matrix in the Newton-step risks to become computationally challenging. It turns out, that we can solve this optimization problem by only

considering tractable inversions of $n-$dimensional matrices [van Houwelingen et al., 2006, Goeman, 2010]. We know that at the global minimum $\chi^*_{\cdot j}$ of $\tilde{f}^{k(j)}$ the gradient (2.42) should vanish. Setting (2.42) to the vector $\mathbf{0}$ reveals that there exists an n-dimensional real vector $\gamma^*_j := -\delta(\chi^*_{\cdot j})$, such that

$$X^T\gamma^*_j \quad = \quad \eta(\chi^*_{\cdot j}) = (\boldsymbol{\lambda}_2 + \boldsymbol{\rho}) \odot \chi^*_{\cdot j} - \rho\Omega^k_{\cdot j}. \tag{2.47}$$

The two equations below follow directly from (2.47):

$$\chi^*_{\cdot j} = (X^T\gamma^* + \rho\Omega^k_{\cdot j}) \div (\boldsymbol{\lambda}_2 + \boldsymbol{\rho}) \tag{2.48}$$

$$\gamma^*_j = (XX^T)^{-1}X\left((\boldsymbol{\lambda}_2 + \boldsymbol{\rho}) \odot \chi^*_{\cdot j} - \rho\Omega^k_{\cdot j}\right) \tag{2.49}$$

Equation (2.48) shows that $\chi^*_{\cdot j}$ lays in an n-dimensional space. Let $h : \mathbb{R}^n \to \mathbb{R}^d$ and $h^{-1} : \mathbb{R}^d \to \mathbb{R}^n$ be the following two (mutually inverse) functions:

$$h(\gamma) : \quad = \quad (X^T\gamma + \rho\Omega^k_{\cdot j}) \div (\boldsymbol{\lambda}_2 + \boldsymbol{\rho}) \tag{2.50}$$

$$h^{-1}(\chi) : \quad = \quad (XX^T)^{-1}X\left((\boldsymbol{\lambda}_2 + \boldsymbol{\rho}) \odot \chi - \rho\Omega^k_{\cdot j}\right) \tag{2.51}$$

The following theorem will form the basis of defining an optimization problem over an $n-$dimensional variable whose optimum can be used to unambiguously reconstruct the $d-$dimensional solution the initial problem.

Theorem 2.1 *Let the function $\phi^{k(j)} : \mathbb{R}^n \to \mathbb{R}$ be defined as:*

$$\phi^{k(j)}(\gamma) := \tilde{f}^{k(j)}(h(\gamma)).$$

χ^* is the global minimum of $\tilde{f}^{k(j)}$ if and only if $\gamma^* := h^{-1}(\chi^*)$ is the global minimum of $\phi^{k(j)}$.

It follows from Theorem 2.1 that the minimization problem (2.41) can be solved by minimizing the n -dimensional function $\phi^{k(j)}$, and setting

$$\chi^{k+1}_{\cdot j} := h\left(\arg\min_{\gamma\in\mathbb{R}^n} \phi^{k(j)}(\gamma)\right). \tag{2.52}$$

Minimizing the function $\phi^{k(j)}$ is done again by the Newton-Raphson's method using the following analytical expressions for the gradient and hessian:

$$\tilde{X} := X \div \left[\sqrt{\boldsymbol{\lambda}_2 + \boldsymbol{\rho}}\right]^T,$$

$$\tilde{\Omega}^k := \Omega^k \div \left[\sqrt{\boldsymbol{\lambda}_2 + \boldsymbol{\rho}}\right],$$

$$\boldsymbol{\Psi}^{k(j)}(\boldsymbol{\gamma}) := \exp\left(-Y_{\cdot j} \odot \tilde{X}\tilde{X}^T\boldsymbol{\gamma} - \rho Y_{\cdot j} \odot \tilde{X}\tilde{\Omega}^k_{\cdot j}\right),$$

$$\mathbf{w}_j(\boldsymbol{\gamma}) := \left(Y_{\cdot j} \odot \sqrt{\boldsymbol{\Psi}^{k(j)}(\boldsymbol{\gamma})}\right) \div \left(\mathbf{1} + \boldsymbol{\Psi}^{k(j)}(\boldsymbol{\gamma})\right),$$

$$\nabla\phi^{k(j)}(\boldsymbol{\gamma}) = \tilde{X}\tilde{X}^T\left(\left(-Y_{\cdot j} \odot \boldsymbol{\Psi}^{k(j)}(\boldsymbol{\gamma})\right) \div \left(\mathbf{1} + \boldsymbol{\Psi}^{k(j)}(\boldsymbol{\gamma})\right)\right) + \tilde{X}\tilde{X}^T\boldsymbol{\gamma},$$

$$\nabla\nabla\phi^{k(j)}(\boldsymbol{\gamma}) := \left(\mathbf{w}_j(\boldsymbol{\gamma}) \odot \tilde{X}\tilde{X}^T\right)^T\left(\mathbf{w}_j(\boldsymbol{\gamma}) \odot \tilde{X}\tilde{X}^T\right) + \tilde{X}\tilde{X}^T\boldsymbol{\gamma}.$$

Second level ADMM for the ζ-update

In the sequel of this section we will rely on a fact, known from subdifferential calculus [Boyd et al. [2011, p. 30],Rockafellar [1997, §23]]. For $\kappa \in \mathbb{R}$, $\kappa \geq 0$ and any real number a, the *soft thresholding operator, S_κ*, is defined as:

$$S_\kappa(a) := \begin{cases} a - \kappa & a > \kappa \\ 0 & |a| \leq \kappa \\ a + \kappa & a < -\kappa, \end{cases}$$

or equivalently,

$$S_\kappa(a) := (a - \kappa)_+ - (-a - \kappa)_+.$$

Theorem 2.2 *Let $\lambda, \rho > 0$, x is a real variable and v is some real constant. The optimiaztion problem*

$$x^* := \arg\min_x \left(\lambda|x| + (\rho/2)(x - v)^2\right)$$

has the closed-form solution

$$x = S_{\lambda/\rho}(v).$$

The ζ-update is:

$$\zeta^{k+1} := \arg\min_{\zeta} \left(|||[\boldsymbol{\lambda}_1] \odot \zeta||_1 + |||[\nu] \odot \zeta(I-R)||_1 + \frac{1}{2}\rho||\zeta - \Omega||_2^2 \right), \quad (2.53)$$

where $\boldsymbol{\lambda}_1 \in \mathbb{R}^{1+d}$, $\boldsymbol{\nu} \in \mathbb{R}^{1+d}$, ζ, χ^{k+1}, $\xi^k \in \mathbb{R}^{(1+d)\times t}$, $(I-R) \in \mathbb{R}^{t\times t}$, $[\cdot] \in \mathbb{R}^{(1+d)\times t}$ denotes the matrix with t columns, equal to the $(1+d)$-dimensional vector $\cdot$, and $\Omega := \chi^{k+1} + \xi^k$. Due to the two L1-norms, the objective function is not differentiable and, unlike the case in the χ-update, it is not column-wise decomposable. Again, we use ADMM, to solve problem (2.53) numerically. Because the objective function remains invariant with respect to transposition, we can write problem (2.53) as well:

$$(\zeta^{k+1})^T := \arg\min_{\zeta^T} \left(|||[\boldsymbol{\lambda}_1]^T \odot \zeta^T||_1 + |||[\nu]^T \odot (I-R)^T\zeta^T||_1 + \frac{1}{2}\rho||\zeta^T - \Omega^T||_2^2 \right) \quad (2.54)$$

Defining the two variables $\overset{\circ}{\chi} := \zeta^T$ and $\overset{\circ}{\zeta} := (I-R)^T\zeta^T$, we present (2.54) in the canonical ADMM form 2.23 as:

$$\begin{aligned} \min \quad & \overset{\circ}{f}(\overset{\circ}{\chi}) + \overset{\circ}{g}(\overset{\circ}{\zeta}) \\ \text{subject to } & (I-R)^T\overset{\circ}{\chi} - \overset{\circ}{\zeta} = [0]_{t\times(1+d)}, \end{aligned} \quad (2.55)$$

where $\overset{\circ}{f}(\overset{\circ}{\chi}) := |||[\boldsymbol{\lambda}_1]^T \odot \overset{\circ}{\chi}||_1 + \frac{1}{2}\rho||\overset{\circ}{\chi} - \Omega^T||_2^2$, and $\overset{\circ}{g}(\overset{\circ}{\zeta}) := |||[\nu]^T \odot \overset{\circ}{\zeta}||_1$. The scaled-form ADMM for is given in Algorithm 2.4

Iterative smooth thresholding for the $\overset{\circ}{\chi}$-update

The $\overset{\circ}{\chi}$-update is:

$$\overset{\circ}{\chi}^{k+1} := \arg\min_{\overset{\circ}{\chi}} \left(|||[\boldsymbol{\lambda}_1]^T \odot \overset{\circ}{\chi}||_1 + \frac{1}{2}\rho||\overset{\circ}{\chi} - \Omega^T||_2^2 + \frac{1}{2}\overset{\circ}{\rho}||(I-R)^T\overset{\circ}{\chi} - \overset{\circ}{\zeta}^k + \overset{\circ}{\xi}^k||_2^2 \right) \quad (2.56)$$

We notice that the problem (2.56) is column-wise decomposable, meaning that we can split it into subproblems of the form

$$\overset{\circ}{\chi}_{\cdot l}^{k+1} := \arg\min_{\overset{\circ}{\chi}_{\cdot l}} \left(|||[\boldsymbol{\lambda}_1]_{\cdot l}^T \odot \overset{\circ}{\chi}_{\cdot l}||_1 + \frac{1}{2}\rho||\overset{\circ}{\chi}_{\cdot l} - \Omega_{\cdot l}^T||_2^2 + \frac{1}{2}\overset{\circ}{\rho}||(I-R)^T\overset{\circ}{\chi}_{\cdot l} - \overset{\circ}{\zeta}_{\cdot l}^k + \overset{\circ}{\xi}_{\cdot l}^k||_2^2 \right) \quad (2.57)$$

Algorithm 2.4 ADMM (general scaled form)

Initialization: $\overset{\circ}{\chi}^0 = \overset{\circ}{\zeta}^0 = \overset{\circ}{\xi}^0 = [0]_{t\times(1+d)}$; $k = 0$
do {
$\overset{\circ}{\chi}$-update: $\overset{\circ}{\chi}^{k+1} := \arg\min_{\overset{\circ}{\chi}} \left(\overset{\circ}{f}(\overset{\circ}{\chi}) + \frac{1}{2}\overset{\circ}{\rho}||(I-R)^T\overset{\circ}{\chi} - \overset{\circ}{\zeta}^k + \overset{\circ}{\xi}^k||_2^2 \right)$
$\overset{\circ}{\zeta}$-update: $\overset{\circ}{\zeta}^{k+1} := \arg\min_{\overset{\circ}{\zeta}} \left(\overset{\circ}{g}(\overset{\circ}{\zeta}) + \frac{1}{2}\overset{\circ}{\rho}||(I-R)^T\overset{\circ}{\chi}^{k+1} - \overset{\circ}{\zeta} + \overset{\circ}{\xi}^k||_2^2 \right)$
$\overset{\circ}{\xi}$-update: $\overset{\circ}{\xi}^{k+1} := \overset{\circ}{\xi}^k + (I-R)^T\overset{\circ}{\chi}^{k+1} - \overset{\circ}{\zeta}^{k+1}$
$k = k+1$
} while($k < MAXITER$ and not converged)

The term $(I-R)^T\overset{\circ}{\chi}$ represents the $t\times(1+d)$-dimensional matrix, the j^{th} row of which represents the row-vector difference[5] $\left(\overset{\circ}{\chi}_{j\cdot} - \overset{\circ}{\chi}_{(j+1)\cdot}\right)$. Because of this coupling between consecutive rows of $\overset{\circ}{\chi}$, problem 2.56 cannot be row-decomposed. In addition, gradient descent methods like Newton-Raphson are not applicable in the presence of L1-norm terms. Therefore, we use a coordinate descent approach for solving problem (2.57) for $l = 1, ..., 1+d$. To find a minimum of a function using coordinate descent, in every iteration, one searches a minimum along one coordinate direction from the current point, considering all other coordinates as constants. Every next iteration uses a different coordinate direction cyclically throughout the procedure. In terms of convergence rate, completing one cycle along all directions is equivalent to one gradient descent iteration.

Let $\overset{\circ}{\chi}_{\cdot l}$ be the current estimate of $\overset{\circ}{\chi}_{\cdot l}^{k+1}$ from (2.57), and let $j \in \{1, ..., t\}$. After eliminating constant terms from rows far away from j, a coordinate descent

[5] In the case $j = t$, $j+1$ should be thought of as 1.

step for the j^{th}element of $\overset{\circ}{\chi}_{\cdot l}$ consists in solving

$$\begin{aligned}
\overset{\circ}{\chi}_{jl}^{+} &:= \arg\min_{\overset{\circ}{\chi}_{jl}} \quad ||\lambda_{1l}\overset{\circ}{\chi}_{jl}||_1 + \frac{1}{2}\rho||\overset{\circ}{\chi}_{jl} - \Omega_{lj}||_2^2 \\
&\quad + \frac{1}{2}\overset{\circ}{\rho}||-\overset{\circ}{\chi}_{jl} + \underbrace{(\overset{\circ}{\chi}_{(j-1)l} - \overset{\circ}{\zeta}{}^{k}_{(j-1)l} + \overset{\circ}{\xi}{}^{k}_{(j-1)l})}_{\overset{\circ}{\Omega}_{(j-1)l}}||_2^2 \\
&\quad + \frac{1}{2}\overset{\circ}{\rho}||\overset{\circ}{\chi}_{jl} - \underbrace{(\overset{\circ}{\chi}_{(j+1)l} + \overset{\circ}{\zeta}{}^{k}_{jl} - \overset{\circ}{\xi}{}^{k}_{jl})}_{\overset{\circ}{\Omega}_{jl}}||_2^2 \\
&= \arg\min_{\overset{\circ}{\chi}_{jl}} \quad \lambda_{1l}|\overset{\circ}{\chi}_{jl}| + \frac{1}{2}\rho\left(\overset{\circ}{\chi}_{jl} - \Omega_{lj}\right)^2 \\
&\quad + \frac{1}{2}\overset{\circ}{\rho}\left(\overset{\circ}{\Omega}_{(j-1)l} - \overset{\circ}{\chi}_{jl}\right)^2 \\
&\quad + \frac{1}{2}\overset{\circ}{\rho}\left(\overset{\circ}{\chi}_{jl} - \overset{\circ}{\Omega}_{jl}\right)^2 \\
&= \arg\min_{\overset{\circ}{\chi}_{jl}} \quad \lambda_{1l}|\overset{\circ}{\chi}_{jl}| + \frac{\rho + 2\overset{\circ}{\rho}}{2}\left(\overset{\circ}{\chi}_{jl} - \frac{\rho\Omega_{lj} + \overset{\circ}{\rho}\overset{\circ}{\Omega}_{(j-1)l} + \overset{\circ}{\rho}\overset{\circ}{\Omega}_{jl}}{\rho + 2\overset{\circ}{\rho}}\right)^2 \qquad (2.58)
\end{aligned}$$

By denoting $\kappa_{jl} := \frac{\rho+2\overset{\circ}{\rho}}{2}$ and $a_{jl} := \frac{\rho\Omega_{lj}+\overset{\circ}{\rho}\overset{\circ}{\Omega}_{(j-1)l}+\overset{\circ}{\rho}\overset{\circ}{\Omega}_{jl}}{\rho+2\overset{\circ}{\rho}}$ and using Theorem 2.2, we find:

$$\overset{\circ}{\chi}_{jl}^{+} = S_{\kappa_{jl}}(a_{jl}).$$

To find $\overset{\circ}{\chi}_{\cdot l}^{k+1}$, we repeat the same step, letting the index j to iterate cyclically over the $\{1, ..., t\}$ until satisfying a convergence criterion for the difference in the objective function between two complete cycles.

Smooth thresholding for the $\overset{\circ}{\zeta}$-update

The $\overset{\circ}{\zeta}$-update is:

$$\overset{\circ}{\zeta}{}^{k+1} := \arg\min_{\overset{\circ}{\zeta}} \left(||[\nu]^T \odot \overset{\circ}{\zeta}||_1 + \frac{1}{2}\overset{\circ}{\rho}||(I-R)^T\overset{\circ}{\chi}{}^{k+1} - \overset{\circ}{\zeta} + \overset{\circ}{\xi}{}^{k}||_2^2 \right). \qquad (2.59)$$

This problem is column- and row-decomposable and can easily be solved for each element of $\overset{\circ}{\zeta}^{k+1}$ by smooth thresholding (Theorem 2.2):

$$\overset{\circ}{\zeta}^{k+1}_{jl} = S_{\nu/\overset{\circ}{\rho}}\left(\overset{\circ}{\chi}^{k+1}_{jl} - \overset{\circ}{\chi}^{k+1}_{(j+1)l} + \overset{\circ}{\xi}^{k}_{jl}\right),\ j = \{1,...,t\},\ l = \{1,...,1+d\}. \quad (2.60)$$

2.3 Experiments with synthetic data-sets

We tested the FENLR model on a number of synthetic data-sets, which we produced in the following way:

1. Fix values for the regularizing parameters λ_1, λ_2 and ν and sample a coefficient matrix $B \in \mathbb{R}^{100\times 8}$, where each row has been sampled from the corresponding prior distribution:

$$\begin{aligned} B_{i,1} &\sim \mathcal{N}\left(0, \frac{1}{\lambda_2}\right) \times Lap\left(\mathbf{0}, \frac{1}{\lambda_1}\right);\ k = 2,...,8 \\ (B_{i,k} - B_{i,k-1}) &\sim \mathcal{N}\left(0, \frac{1}{\lambda_2}\right) \times Lap\left(0, \frac{1}{\lambda_1} I\right) \times Lap\left(0, \frac{1}{\nu}\right), \end{aligned}$$

   for $i = 1,...,100$.

2. Then, generate a design matrix $X \in \mathbb{R}^{3000\times 100}$ from a uniform distribution and calculate the probabilities π for each observation.

3. Finally, draw the corresponding response vector from a Bernoulli distribution (coin-tossing) with the corresponding probability π for every observation in X.

Using the above procedure, we generated families of 20 data-sets for 18 different combinations of regularizing parameters simulating the Cartesian product of the following scenarios for each regularizer:

- λ_1 : non-sparse (1.5); intermediate (3.5); sparse (6.5);
- λ_2 : small (0); big (6);
- ν : non-sparse (0); intermediate (4); sparse (8).

With each of the 20 training data-sets for a given combination of meta-parameters, we evaluated the performance of the following five models:

- FENLR
- ENLR (single task elastic net logistic regression)

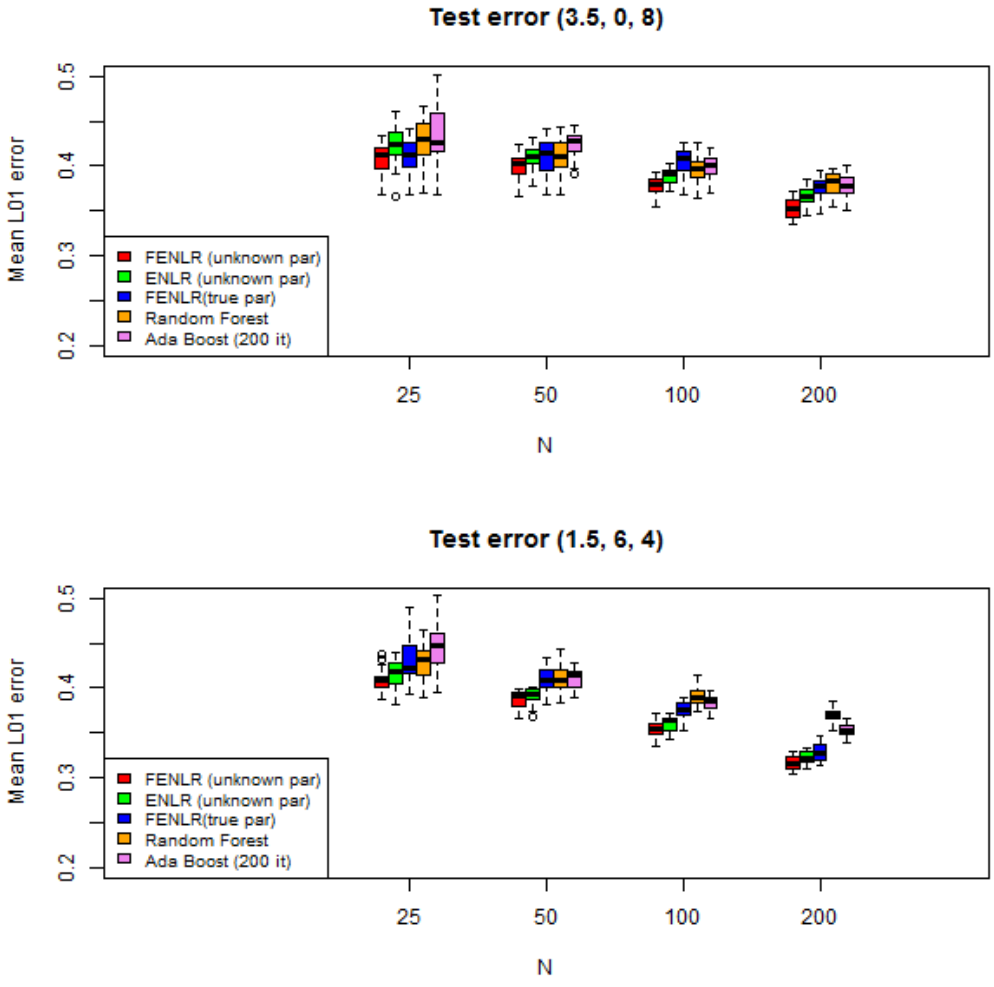
- FENLR with the predefined regularizing parameter values;
- Random Forest (used R package "RandomForest");
- Ada Boost with 200 iterations (used R package "ada").

Instead of performing a costly cross-validation for optimal meta-parameter selection, we reserved a single validation set containing 1400 observations and a single test set containing other 1400 observations.

Figure 2.2 represents comparative results on two meta-parameter combinations with different number of training observations (25, 50, 100 and 200) and Figure 2.3 shows the histogram of the selected meta-parameter values.

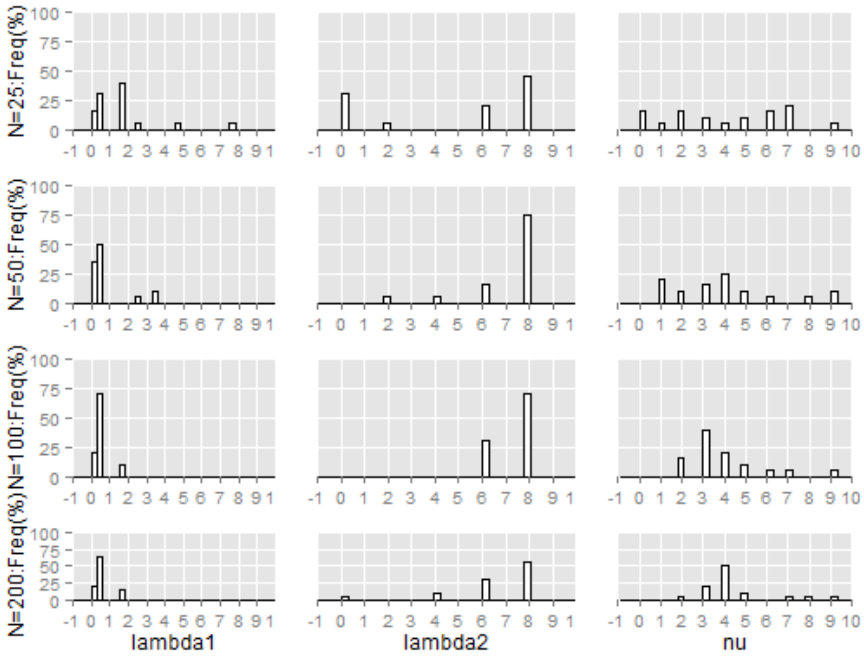
It can be seen from these results, that the FENLR model tends to outperform single-task learning models, particularly when there is high similarity of the neighboring tasks (bigger values of the parameter ν). In most cases, the the parameter ν is estimated at values, that are lower than the original ones, particularly when the number of training data-points, N, is small. It is also noticeable that most models achieve optimal validation error for large values of λ_2 and small values of the other regularizing parameter λ_1. This can be explained with the fact that the original model (true values of the coefficients B) contains many small but non-zero values, while the L1 penalty imposes sparsity on the coefficient estimates.

Figure 2.2: Comparison of estimated expected prediction L01 error for different models trained on synthetic data-sets.

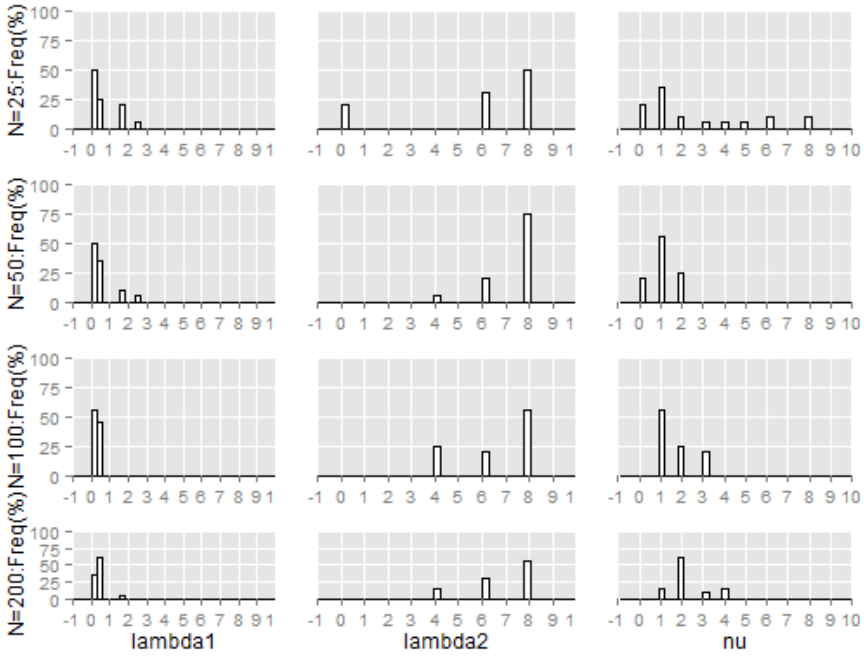


The values of the regularizing parameters λ_1, λ_2 and ν are given in parentheses. Every box-whisker represents the average test-set L01 error on a test-set of 1400 observations.

Figure 2.3: Histograms of estimated optimal regularizing parameters
a) original regularizing meta-parameters: $\lambda_1 = 3.5$, $\lambda_2 = 0$, $\nu = 8$.



a) originial regularizing meta-parameters: $\lambda_1 = 1.5$, $\lambda_2 = 6$, $\nu = 4$.



Chapter 3

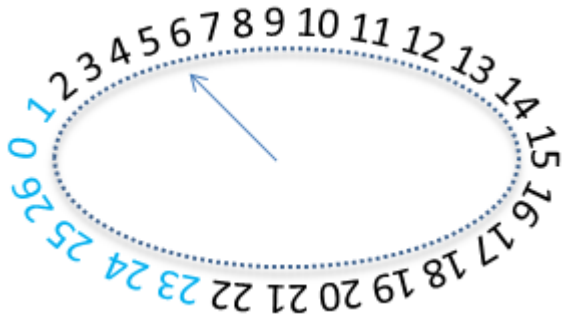
Inference of post-infection time from infected murine gene-expression data

In this chapter we consider different supervised learning formulations of the problem of genome based post-infection time inference in malaria-infected organisms. For each formulation, we define a model, which we test and evaluate with respect to its accuracy of prediction on new data and interpretation capacity. All described models have been trained on data obtained from an Illumina Inc. bead-chip microarray experiment, including 78 samples[1] from three malaria-infected mice, collected over a period of 26 days after the infection and ten control samples taken from healthy mice.

It is important to note that all three infected mice recovered from the disease and that there is a well pronounced similarity between gene-expression profiles for the initial days of infection (i.e. days $1, 2, 3, ...$), and profiles for the final days (i.e. $..., 24, 25, 26$) and controls (day 0). This observation justifies a representation of the post-infection time as a circular axis and we denote this axis by $\mathbb{T}$ (Figure 3.1).

[1] Unless specified otherwise, throughout this chapter, by the word "sample" we will mean a $d-$dimensional numerical vector representing gene-expression intensities measured in one Illumina-bead chip microarray.

Figure 3.1: The circular time-axis $\mathbb{T}$, representing the post-infection time of an organism, which recovered fully from the disease.



In blue color: the time window $w^{(23,6)}$.

3.1 Classification formulation

In a classification formulation, we consider the training data as realizations from [2]

$$(X_1, Y_1), ..., (X_n, Y_n) \text{ i.i.d. },$$

where the predictor or feature vector $X_i \subset \mathbb{R}^d$, $i = 1, ..., n$, are random gene-expression profiles and the response vector $Y = (Y_1, ..., Y_n) \subset \{0, 1, ..., 26\}^n$ denotes the corresponding post-infection times [3]. A predictor is defined as a function of the form:

$$\mathcal{P} : \mathbb{R}^d \to \mathbb{T}.$$

We call the value $\mathcal{P}(\mathbf{x})$ for an expression profile $\mathbf{x}$ the predicted post-infection time, and define the prediction loss function of $\mathcal{P}$ as the expected difference on $\mathbb{T}$ between the predicted and true post-infection time:

$$\mathcal{L}(\mathcal{P}) := E_X[Y \ominus \mathcal{P}(X) \,|X], \tag{3.1}$$

where the operation $\ominus$ denotes the difference operation on $\mathbb{T}$, defined as the minimal number of days separating two points on $\mathbb{T}$, for example, $23 \ominus 2 = 2 \ominus 23 = 6$, $12 \ominus 23 = 23 \ominus 12 = 11$ (Figure 3.1).

In the two subsections below we denote the training data by $[X|\mathbf{y}]$, where $X \in \mathbb{R}^{n \times (1+d)}$ is the design matrix, and $\mathbf{y} \in \mathbb{T}^n$ is the corresponding observed

[2] Note however, that the independence assumption is violated in the case of training observations taken from the same mouse case, which is inevitable in the case of 3 infected mice.

[3] The label 0 denotes a control sample

response vector. The goal is to find a predictor $\mathcal{P}$, which minimizes the prediction loss function.

3.1.1 The k-Nearest Neighbor Predictor

The k-nearest neighbor algorithm (k-NN) is a non-parametric method for classifying objects based on closest training examples in the feature space. Let $\delta(\cdot,\cdot)$ be an arbitrary distance function defined on $\mathbb{R}^{d+1}$. Considering the design-matrix X, the k-Neighborhood of a training feature vector $\mathbf{x}$ is defined as the set of the closest k training feature vectors to $\mathbf{x}$ with respect to the distance δ and is denoted as $N_k(\mathbf{x};\, X,\, \delta)$. The k-nearest neighbor predictor is defined as

$$\mathcal{P}_{kNN}(\mathbf{x};\, [X|\mathbf{y}], \delta) := \arg\max_{y\in\mathbb{T}} \sum_{i\in N_k} 1[y = y_i] \tag{3.2}$$

with the remark that in the case of non-unique maximum in the above formula, the response is selected by randomly selecting one of the optimal candidate classes.

In the case of very limited number of infected training samples, our choice of the parameter k was limited to $k = 1$ and, after comparison with some other distance metrics, we selected the Euclidean norm of a vector difference as our preferred distance metric.

3.1.2 The aggregated time-window predictor (ATWINP)

Let $t := |\mathbb{T}|$ be the total number of days. A time-window $w^{(jl)}$ of length $l < t$ and starting from day j is defined as the ordered sequence of l consecutive days on $\mathbb{T}$, starting from day j. For example $w^{(1,5)} := \{1, 2, 3, 4, 5\}$, and $w^{(22,6)} := \{22, 23, 24, 25, 26, 0\}$. Consider all t overlapping time-windows of a fixed number of days $l < t$. For instance with $l = 3$, these would be:

$w^{(0,3)}$	$w^{(1,3)}$	.....	$w^{(24,3)}$	$w^{(25,3)}$	$w^{(26,3)}$
0	1	.....	24	25	26
1	2	.....	25	26	0
2	3	.....	26	0	1

Let $\mathbf{x}$ be a gene-expression sample. For a fixed window length $l \in \{1, ..., t-1\}$ and for each time window $w^{(jl)}$, $j \in \mathbb{T}$, denote by $W^{(jl)}(\mathbf{x})$ the probability that the post-infection time of $\mathbf{x}$ is in the window $w^{(jl)}$:

$$W^{(jl)}(\mathbf{x}) := \mathbb{P}[Y \in w^{(jl)} | X = \mathbf{x}].$$

Denote by $\mathbf{y}^{(jl)} \in \{0,1\}^n$ the appurtenance of each element of the known response vector$\mathbf{y}$ in the window $w^{(jl)}$:

$$\mathbf{y}^{(jl)} := \mathbf{1}(\mathbf{y} \in w^{(jl)}).$$

Let for $j \in \mathbb{T}$, $\hat{W}^{(jl)}(\mathbf{x})$, be an estimator of the above probability $W^{(jl)}(\mathbf{x})$, obtained by training on the data binary classification data $[X|\mathbf{y}^{(jl)}]$.

The ATWINP predictor for the estimators $\hat{W}^{(jl)}(\mathbf{x})$ is defined as follows:

$$\mathcal{P}_{ATWINP}\left(\mathbf{x}; \hat{W}^{(\cdot l)}\right) := \arg\max_{y \in \mathbb{T}} \sum_{j=0}^{t-1} \left\{ \frac{\hat{W}^{(jl)}(\mathbf{x})}{l} 1[y \in w^{(jl)}] + \frac{1 - \hat{W}^{(jl)}(\mathbf{x})}{t-1-l} 1[y \notin w^{(jl)}] \right\}. \tag{3.3}$$

3.2 Regression formulation

In a regression approach, we assume that the post-infection time can be modeled as a continuous function of the gene-expression profile:

$$f : \mathbb{R}^d \to [0,26] \subset \mathbb{R}$$

Again, we consider the training data as realizations from

$$(X_1, Y_1), ..., (X_n, Y_n) \text{ i.i.d. },$$

where the predictor or feature vector $X_i \subset \mathbb{R}^d$, $i = 1, ..., n$ are random gene-expression profiles, but unlike classification, the response vector $Y = (Y_1, ..., Y_n) \subset [0,26]^n$ is a real vector with the property:

$$Y_i = f(X_i) + \epsilon_i,\ i = 1, .., n,$$

where ϵ_i are i.i.d. realizations from $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Because $E[\epsilon] = 0$, the value $f(\mathbf{x})$ equals the expected value of Y given a sample $\mathbf{x}$, i.e. $f(\mathbf{x}) = E[Y|\mathbf{x}]$. If $\hat{f}$ is a given estimate of f, we define a predictor of the post-infection time from a sample $\mathbf{x}$ as:

$$\mathcal{P}_{\hat{f}}(\mathbf{x}) := \lfloor \hat{f}(\mathbf{x}) \rceil_{\mathbb{T}},$$

where $\lfloor \cdot \rceil_{\mathbb{T}}$ denotes the nearest integer to the real number $\cdot$, found in $\mathbb{T}$.

There are different parametric and non-parametric approaches to find an estimate of f. In the case $d \gg n$, we consider regularized linear regression as

the most promising approach. In its simplest form, linear regression assumes that f is a linear function of the predictor variables:

$$f(\mathbf{x}) = \beta_0 + \mathbf{x}^T\boldsymbol{\beta},$$

where $\beta_0 \in \mathbb{R}$ and $\boldsymbol{\beta} \in \mathbb{R}^d$. Further, we will omit the intercept β_0 from the notation, assuming that $\boldsymbol{\beta}$, $\mathbf{x} \in \mathbb{R}^{1+d}$ with $x_1 = 1$.

To find a fit of the model coefficients, $\boldsymbol{\beta}$ to the data $[X|\mathbf{y}]$, we used the R-package "penalized" [Goeman, 2010] which provides a MAP estimator with elastic net regularizing prior.

3.3 Comparative model evaluation based on mouse and human data

All described models have been trained on an Illumina Inc. bead-chip microarray data-set including 78 samples from three malaria-infected mice, collected over a period of 26 days after the infection and ten control samples taken from healthy mice. Specifically, we designed two data-sets:

- "Mouse" containing measurements of 5757 gene-expression levels in 88 mouse-samples (78 infected and 10 control samples; see subsection A.1.1);
- "Mouse-Human" containing measurements of 2589 gene-expression levels in 243 samples (88 mouse, 94 infected human, 59 control human), which was obtained after homology mapping between the mouse data and a data-set of Illumina microarray samples from malaria-infected human patients (see section A.1 and Idaghdour et al. [2012] for further details).

To estimate the prediction error (3.1), we performed 3-fold "leave-one-mouse-out" cross validation. In every cross-validation fold, we trained one of the models on all samples from two of the infected mice and 6 of the control mouse-samples, leaving the infected samples from the other mouse and the remaining control-samples as a validation set. Experiments with other cross-validation scenarios, allowing for the samples from one infected mouse to be split over the training and validation set in a fold, resulted in overoptimistic estimates of the prediction error, due to non-disease-associated correlations between train- and test-samples belonging to the same organism. Tuning the regularizing meta-parameters λ_1, λ_2 and ν for the penalized linear models

has been done by parallel estimation of the 3-fold cross validation error on a finite grid of meta-parameter values[4].

3.3.1 Model evaluation based on the post-infection-time prediction error

Table 3.1 and (Figure 3.2) compare the estimated average post-infection-time prediction error for the following six predictor models:

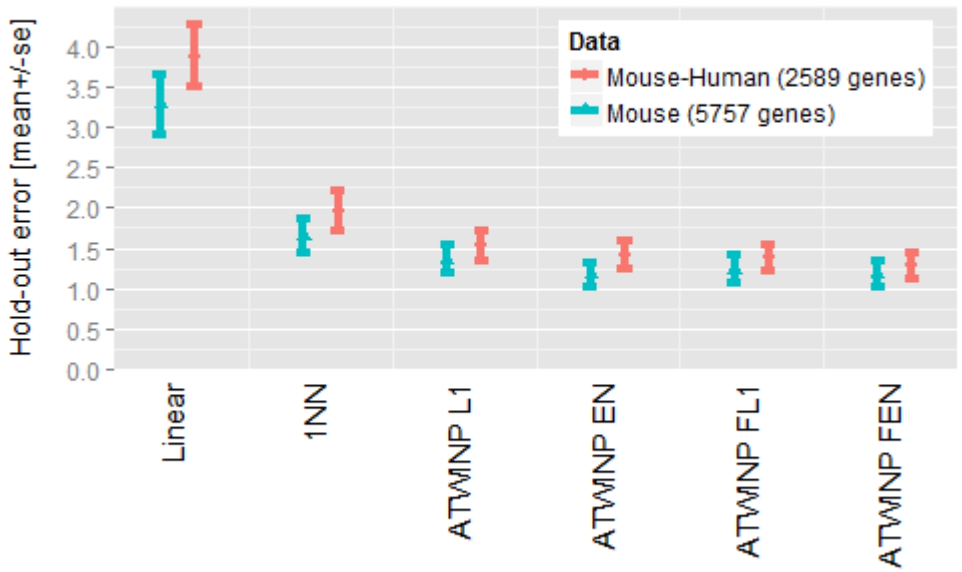
- Linear: an elastic net regularized linear regression model ($\lambda_1 > 0$, $\lambda_2 > 0$, $\nu = 0$);
- 1NN: first-nearest-neighbor based on Euclidean distance between test and training samples;
- ATWINP L1: ATWINP-model using lasso-penalized LLR as underlying time-window-predictor ($\lambda_1 > 0$, $\lambda_2 = \nu = 0$);
- ATWINP EN: ATWINP-model using elastic net-penalized LLR ($\lambda_1 > 0$, $\lambda_2 > 0$, $\nu = 0$);
- ATWINP FL1: ATWINP-model using fused lasso-penalized LLR ($\lambda_1 > 0$, $\lambda_2 = 0$, $\nu > 0$);
- ATWINP FEN: ATWINP-model using FENLR (fused elastic net-penalized LLR) ($\lambda_1 > 0$, $\lambda_2 = 0$, $\nu > 0$).

Table 3.1: Summary of estimated expected prediction error for the tested models

		Mouse (5757 genes)					Mouse-Human (2589 genes)				
$\mathcal{P}$	l	λ_1	λ_2	ν	E	SE	λ_1	λ_2	ν	E	SE
Linear	n.a.	0.6	7	n.a.	3.28	0.38	0	6.5	n.a.	3.88	0.39
1NN	n.a.	n.a.	n.a.	n.a.	1.64	0.21	n.a.	n.a.	n.a.	1.97	0.25
ATWINP L1	12	1.4	0	0	1.36	0.18	1	0	0	1.53	0.18
ATWINP EN	12	0.7	1.2	0	1.17	0.15	0.9	1	0	1.42	0.18
ATWINP FL1	12	0.9	0	0.2	1.24	0.18	1.3	0	1.1	1.38	0.16
ATWINP FEN	12	0.4	0.8	0.9	1.18	0.16	1.4	1.4	0.8	1.28	0.16

[4]Computation has been performed on the ETH high-performance cluster "Brutus" (http://en.wikipedia.org/wiki/Brutus_cluster)

Figure 3.2: Comparison of the tested predictors with respect to mean prediction error

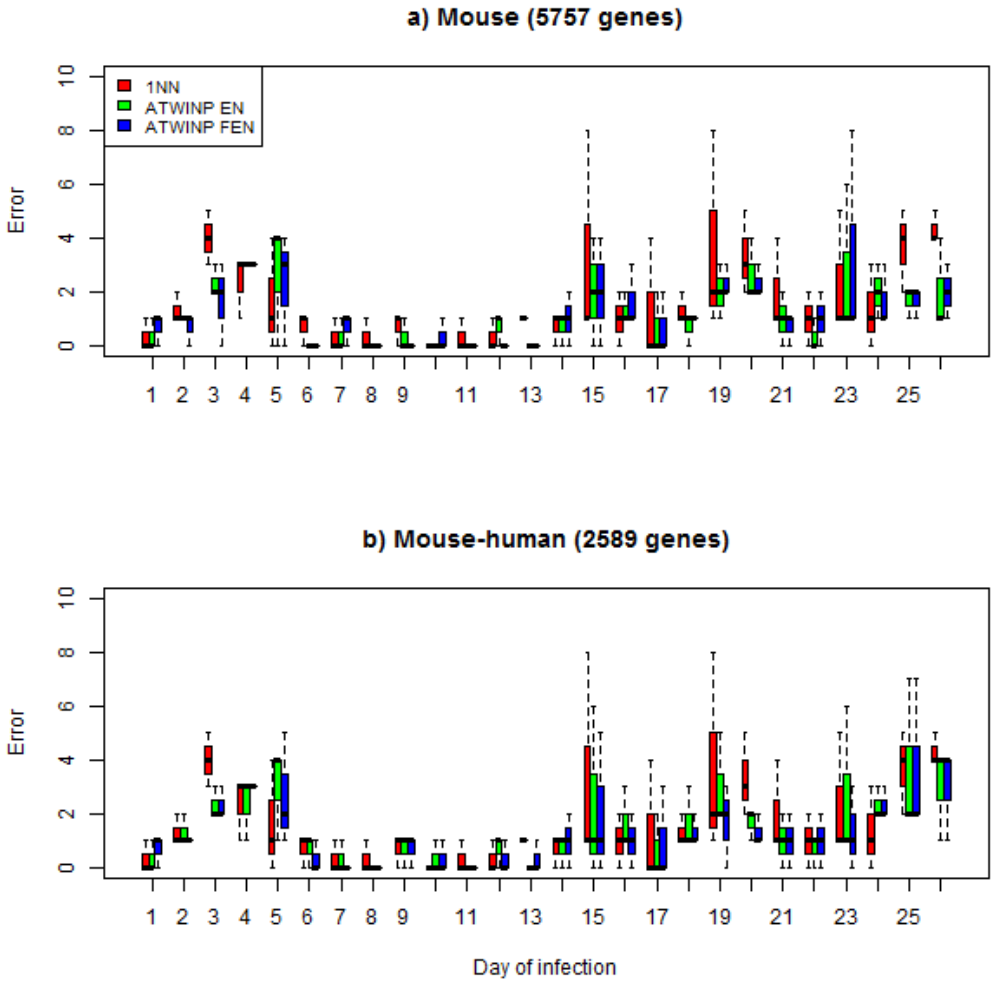


The increase in prediction error in the Mouse-Human data-set, caused by loss of information from filtering-out non-mapped genes, remains within the range of one standard error. The elastic net penalized linear regression model is dominated by all other predictors, implying a lack of linear dependency of the post-infection-time from the gene-expression levels. The nearest-neighbor predictor is dominated by all ATWINP predictors by more than one standard error, suggesting that the Euclidean distance between complete samples includes the effect of many gene-levels, which are uninformative for the malaria post-infection time. The ATWINP EN and FEN predictors reached the lowest cross-validation prediction errors, while demonstrating two different classification approaches to solve the inference problem, i.e. single-task and multi-task classification.

Figure 3.3 gives a box-plot representation of the prediction error associated with every post-infection day for the 1NN, ATWINP EN and ATWINP FEN predictors. The prediction error patterns are very similar in the two tested data-sets, suggesting low prediction errors in the interval of days $[6, 13]$ and higher prediction errors in the interval of days $[14, 5]$[5], peaking at days 5, 15, 19, 23.

[5]Intervals are to be interpreted as sequences of consecutive days on the circular axis $\mathbb{T}$.

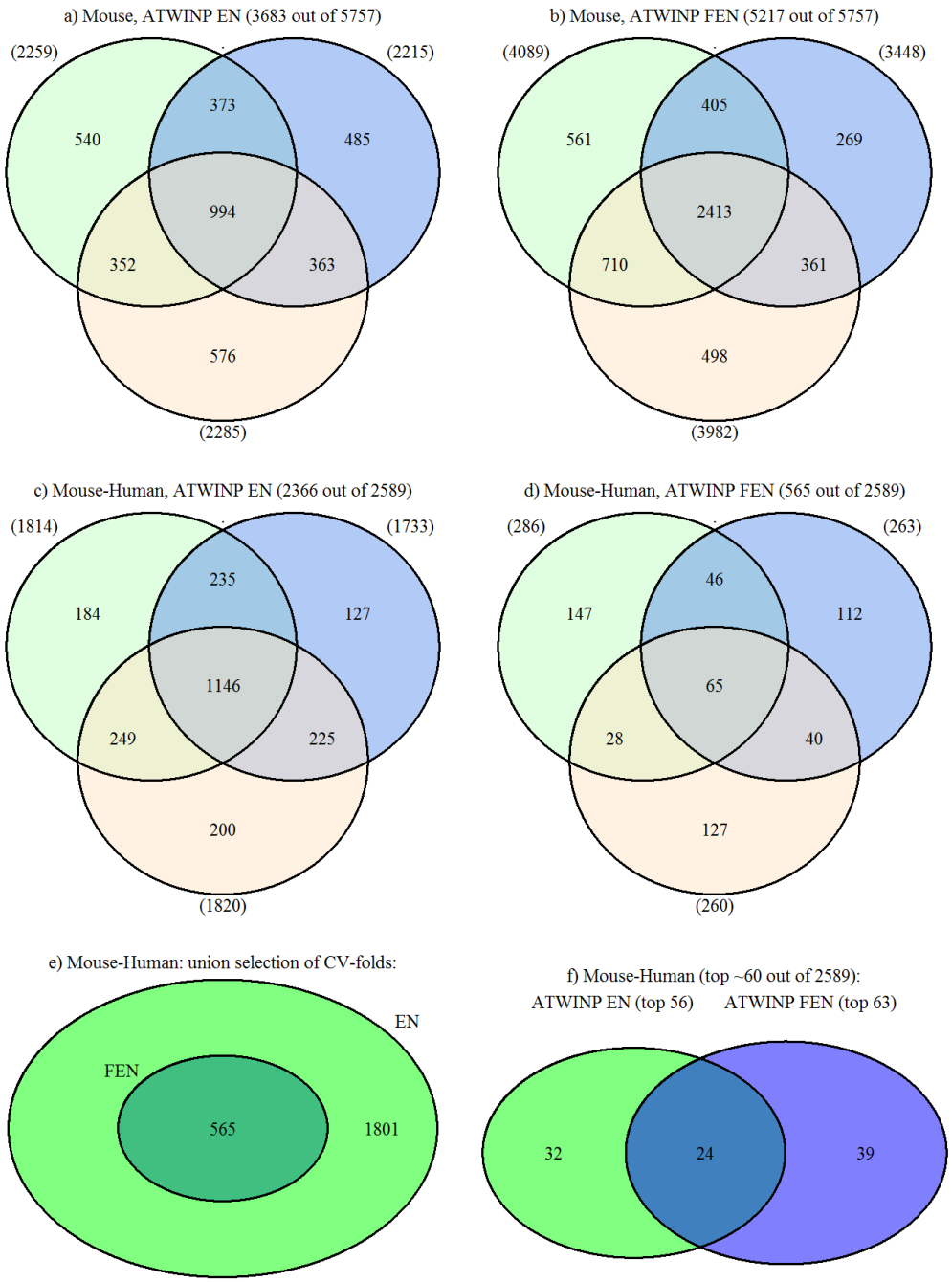
Figure 3.3: Comparison of the tested predictors with respect to prediction-error for each day of infection
Each box-whisker represents the measured prediction error from the three cross-validation folds.



3.3.2 Model evaluation based on automatic variable selection

An important property of the models with an elastic net penalty is that they do both continuous shrinkage and automatic variable selection simultaneously [Zou and Hastie, 2005]. By automatic variable selection, we mean that those models can eliminate non-informative genes by setting their cor-

Figure 3.4: Venn diagram of selected genes by each CV-fold



a,b,c,d: The cardinality of the sets is given in parentheses next to the corresponding circles. The cardinality of the union of all folds is given in parantheses next to the model name.

responding model coefficients to zero, and include whole groups of highly correlated relevant genes into the model, once one of these genes has been selected [Zou and Hastie, 2005].

For both data-sets, the linear regression model reached its optimal prediction error at low values of the lasso regularizing parameter λ_1, favoring higher values for the ridge penalty λ_2. This accounts for a non-sparse estimate of the model coefficients assigning small non-zero weights to all genes. Therefore, the linear regression failed to perform automatic variable selection.

While the nearest neighbor predictor has a very intuitive interpretation, because it can provide a similarity-based ranking of the training samples with respect to a test-sample, it is unable to perform automatic variable selection and, like other non-selective models, can be sensitive to noise and misleading non-disease-associated correlations between the training and test-samples.

Figure 3.4 represents a Venn diagram of the selected sets of genes by the ATWINP EN and FEN models, in which each circle corresponds to a set of genes selected in one cross-validation fold. In the case of the Mouse data-set (a,b), the two models reached minimal cross-validation error at small values of λ_1 resulting in non-sparse coefficient profiles. There is no consensus in the selected genes sets between different cross-validation folds, as for both models the three-fold intersection contains less than 50% of the overall selected genes (number given in parentheses next to the model name). For the Mouse-Human data-set (c,d,e,f), we observe a non-sparse gene selection for the ATWINP EN model and a sparse selection for the ATWINP FEN model. Again, no consensus in the gene-selection could be observed between the three cross-validation folds. In (e) we see that all 565 genes, that have been selected in the sparse FEN model have been included as well in the EN selection. By applying a threshold to the absolute coefficient values, we produced a list of the top 56 genes selected by the ATWINP EN model and the top 63 genes, selected by the ATWINP FEN model (Figure 3.4 f). In these two sets, we retain genes that exceed the threshold value in at least on cross-validation fold for at least one day. The two sets agree on 24 genes. Figure 3.5 and Figure 3.6 represent the estimated coefficients of these sets of genes in the form of a heat-map, produced after hierarchical clustering with respect to the pairwise correlation between the coefficient vectors.

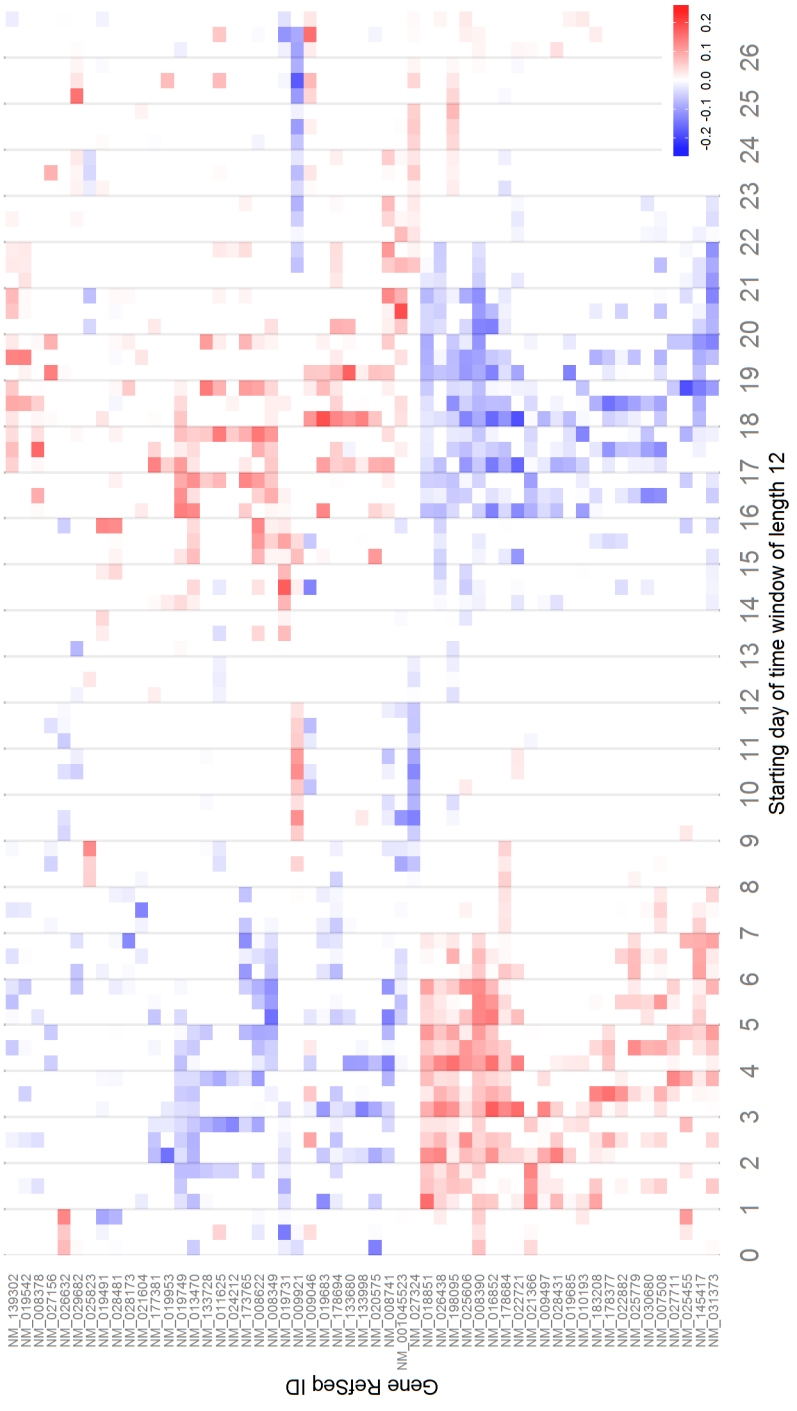


Figure 3.5: Heat-map representation of selected genes (Mouse-Human, ATWINP EN (threshold=0.185)

Each colored square represents the coefficient value estimated in one cross-validation fold for one time-window.

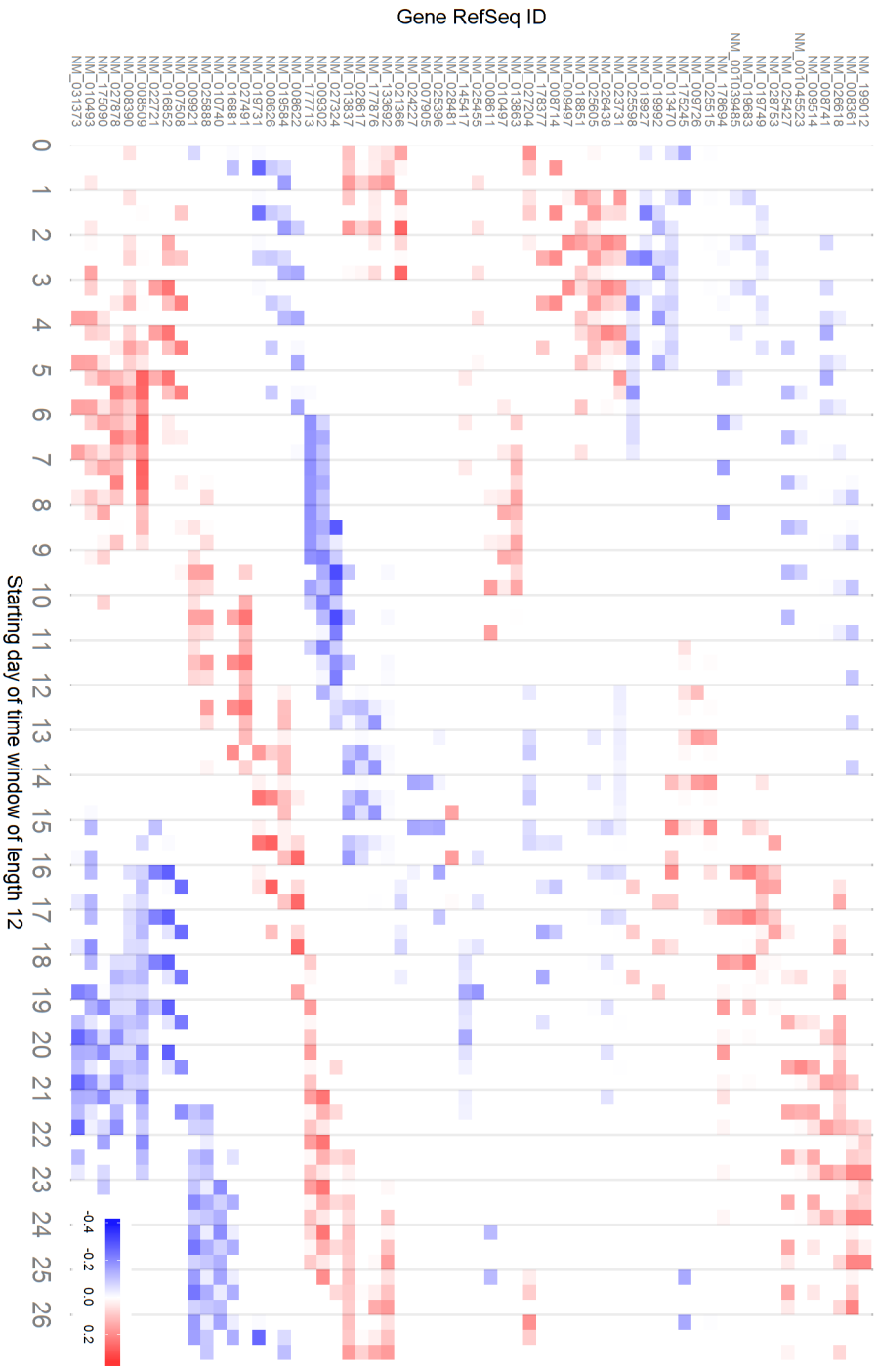
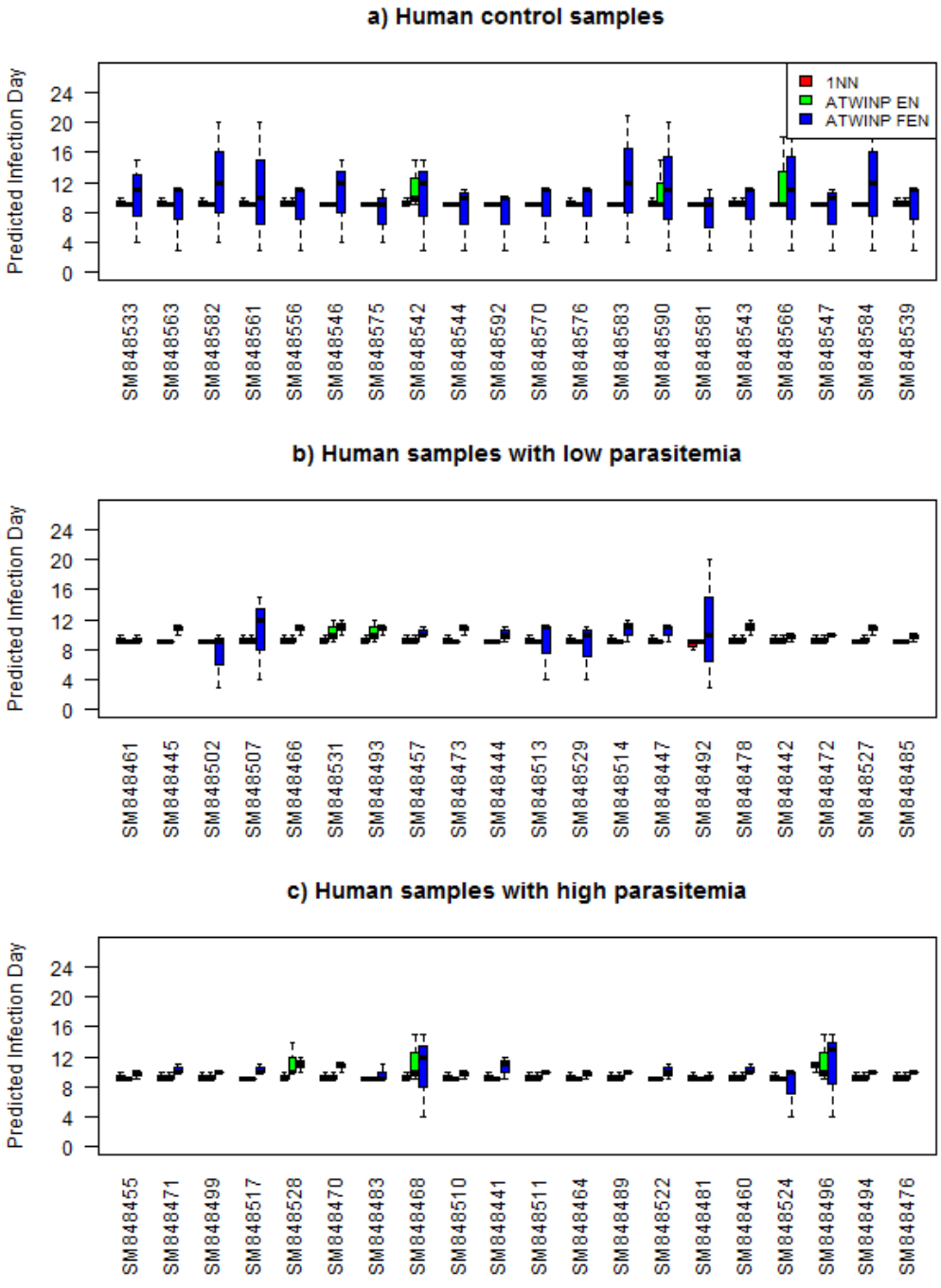


Figure 3.6: Heat-map representation of selected genes (Mouse-Human, ATWINP FEN (threshold=0.2)

3.3.3 Estimation of post-infection time in humans

To test the transfer-learning approach, we applied the nearest neighbor, ATWINP EN and FEN predictors to the human samples from the Mouse-Human data-set. Specifically, we selected random sub-groups of 20 control human samples, 20 human samples with low parasitemia and 20 human samples with high parasitemia and we invoked on them the predictor models from each cross-validation fold, resulting in three predictions per sample per model. The resulting predictions are given in Figure 3.7. For the control-samples the only valid prediction would be day 0, corresponding to a non-infected mouse sample. However, we see that for 17 out of 20 control human samples, the non-sparse 1NN and ATWINP EN models agree on predictions in the range $[9, 11]$. The ATWINP FEN predictions vary considerably between different cross-validation folds. The model estimated in the first fold predicts days 3 or 4 for all 20 control-samples, the model from the second fold predicts days between 10 and 12 and the model from the third fold predicted days between 9 and 20. The predictions for the low-parasitemia and high-parasitemia groups were predominantly in the range $[9, 11]$ for all samples.

Figure 3.7: Post-infection time prediction in humans



Chapter 4

Discussion

In this thesis, we developed a novel method for inferring Malaria post-infection time as a function of the gene-expression profile in a model organism. Our tests on a homology-mapped Mouse-Human data-set show that the model can predict the post-infection time of a new infected mouse-sample with expected deviation of 1.28 days from the true post-infection time. Based on this result, we can conclude that the gene-expression profile of an infected host-organism preserves information with respect to the beginning of the infection, and can be used to characterize the disease progression on a fine time-scale. Furthermore, we were able to identify a set of genes that are informative for the disease progression in mice and we could quantify the effect of each selected gene at all points in the time-course of the infection.

While these results are very satisfying, we have to admit that the knowledge transfer from mouse to human patients did not provide a valuable estimation of the post-infection time in humans. A further analysis of the reasons leading to this transfer learning failure would go beyond the scope of this thesis. However, it might be useful to give some ideas for future research. To begin with, it might be interesting to analyze the reasons why most post-infection time predictions on human samples are close to days in the range $[9, 11]$. Could this be an artifact of the data pre-processing procedure or, are there biological reasons that would explain the observed proximity between arbitrary human samples and samples of infected mice in this range? Another direction would be to revise the homology mapping procedure. While at the present time the homology mapping has been done in a fully automated way, it might be beneficial to make selection of relevant genes in the human context based on prior biological knowledge from previous genome-wide

studies of malaria in humans [Timmann et al., 2012, Bahcall, 2009]. Further, apart from transfer learning, it would be desirable to extend the ATWINP FENLR predictor to a non-cyclic case, because this would allow to analyze cases in which the infected organism does not recover from the disease.

Finally, it would be desirable to explore the modeling of disease progression in model organisms from the point of view of probabilistic graphical models [Segal et al., 2003] in order to describe the progression of the transcriptome states in mice and thereby possibly identify novel gene relationships that play a key role through the development of the disease.

Appendix A

Appendix

A.1 Preprocessing and homology mapping of murine and human microarray data

A.1.1 Murine Illumina Beadchip microarrays

All described models have been fitted to an Illumina Beadchip microarray data-set including 78 samples from 3 malaria-infected mice, collected once per day over a period of 26 days after the infection and 10 control samples taken from healthy mice.

Each sample in the data-set contains 26579 ILMN probes corresponding to 18744 unique RefSeqs. Preprocessing of the data was made using the R-package "lumi" [Du et al., 2008, Barbosa-Morais et al., 2010] and included the following steps:

1. Background correction based on Illumina Bead-chip control probes;
2. Log-transformation;
3. Detection call filtering with detection-threshold, set to 0.01, keeping genes that were detected in at least 9 of the 10 control-samples or in at least one day for all infected mice;
4. Summarizing multiple probes for a single RefSeq gene ID, by taking the mediane.
5. Only for dataset "Mouse": For all genes, perform a student t-test of the hypothesis of equal mean-values between the distribution of control probes for a given gene and the distribution for infected mice for the

same gene for every individual infection day (1,...,26). Only genes with p-value ¡0.01 for at least one day have been retained, resulting in 5757 retained genes.

The dataset "Mouse" was created following steps 1 to 5 and was stored in the file 'bgaffy.quantile.01.medianacc.01.RData'.

A.1.2 Human Illumina Beadchip microarrays

A detailed description of the data can be found in the supplementary information to Idaghdour et al. [2012]. Gene expression data from human patients:

- 155 human patients, from which 96 were infected with malaria, while the remaining 59 samples were from healthy patients.
- Each sample of the data-set contained 47231 ILMN probes corresponding to 39655 unique RefSeq IDs.

The preprocessing of the human data has been done using the R-package "lumi" [Du et al., 2008, Barbosa-Morais et al., 2010] and included the following steps:

1. Log-transformation;
2. Detection call filtering with detection-threshold, set to 0.01, keeping genes that were detected in at least 90% of the control samples or in at least 4% of the infected samples.
3. Summarizing multiple probes for a single RefSeq gene ID, by taking the mediane.

A.1.3 Homology mapping from mouse to human genes

- Of 18744 mouse sequences:
  - 15587 have a homologous sequence found in human,
  - 15328 of which are available on the human BeadChip of which:
- 7683 mouse sequences point to a unique human sequence,
- 6832 mouse sequences point to more than one human sequence,
- 813 mouse sequences point to human sequences pointed by other mouse sequences

Here is how the mapping has been produced:

1. Generate unique nuID identifiers for each mouse and human probe sequence
2. Map the mouse and human nuIDs to the corresponding ProbeID, Entrez gene ID, gene symbol, and RefSeq using the lumi package for R. The RefSeq is the key that should be used for homology mapping to corresponding human RefSeq.
3. Use the BioMart getLDS() function (R package Biomart) to create a homology-linked dataset of the mouse and human datasets available in the Ensembl database. Instructions on how to create a linked-dataset are available in the biomart package vignette available at http://www.bioconductor.org/packages/2.12/bioc/manuals/biomaRt/man/biomaRt.pdf, page 8.
   The query sent to the ensemble database selects all matching couples of the following mouse and human sequence attributes: refseq_ncrna_predicted (for RefSeq's starting with XR), refseq_mrna_predicted (for RefSeq's starting with XM), refseq_mrna (for RefSeq's starting with NM), refseq_ncrna (for RefSeq's starting with NR).
4. Based on the matching couples found in the step 3, create a mapping table, which is available as an attached Excel file Homology-Mouse2Human.xlsx and as an exported R-object-file mouse2human.RData.

Based on the above procedure and the preprocessing steps described in the previous two sections, a combined one-to-one mapped "Mouse-Human" dataset was generated containing 2589 gene-entries. This data-set was quantile normalized and stored as an exported R-object file 'm.bgaffy.01.medianacc.h.01.medianacc.quantile.RData'.

Bibliography

Joshua Attenberg, Kilian Weinberger, and Anirban Dasgupta. Collaborative Email-Spam Filtering with the Hashing Trick. pages 1–4, 2009.

Orli G. Bahcall. Human disease: Malaria GWA study brings progress for infectious disease genetics. *Nature Reviews Genetics*, 10(7):428–429, July 2009. ISSN 1471-0056. doi: 10.1038/nrg2627.

Nuno L Barbosa-Morais, Mark J Dunning, Shamith a Samarajiwa, Jeremy F J Darot, Matthew E Ritchie, Andy G Lynch, and Simon Tavaré. A re-annotation pipeline for Illumina BeadArrays: improving the interpretation of gene expression data. *Nucleic acids research*, 38(3):e17, January 2010. ISSN 1362-4962.

Stephen Boyd, Neal Parikh, and Eric Chu. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

Patrícia Brasil, Anielle de Pina Costa, Renata Saraiva Pedro, Clarisse da Silveira Bressan, Sidnei da Silva, Pedro Luiz Tauil, and Cláudio Tadeu Daniel-Ribeiro. Unexpectedly long incubation period of Plasmodium vivax malaria, in the absence of chemoprophylaxis, in patients diagnosed outside the transmission area in Brazil. *Malaria journal*, 10(1):122, January 2011. ISSN 1475-2875.

P Bühlmann and M Mächler. *Computational statistics*, volume 2008. 2011.

Pan Du, Warren a Kibbe, and Simon M Lin. lumi: a pipeline for processing Illumina microarray. *Bioinformatics (Oxford, England)*, 24(13):1547–8, July 2008. ISSN 1367-4811.

Theodoros Evgeniou and Charles A Micchelli. Learning Multiple Tasks with Kernel Methods. 6:615–637, 2005.

Jelle J Goeman. L1 penalized estimation in the Cox proportional hazards model. *Biometrical journal. Biometrische Zeitschrift*, 52(1):70–84, February 2010. ISSN 1521-4036.

Trevor. Hastie, Robert. Tibshirani, and JJH Friedman. *The elements of statistical learning*. 2001.

Youssef Idaghdour, Jacklyn Quinlan, Jean-Philippe Goulet, Joanne Berghout, Elias Gbeha, Vanessa Bruat, Thibault de Malliard, Jean-Christophe Grenier, Selma Gomez, Philippe Gros, Mohamed Chérif Rahimy, Ambaliou Sanni, and Philip Awadalla. Evidence for additive and interaction effects of host genotype and infection in malaria. *Proceedings of the National Academy of Sciences of the United States of America*, 109(42):16786–93, October 2012. ISSN 1091-6490.

Meghana Kshirsagar, Jaime Carbonell, and Judith Klein-Seetharaman. Multitask learning for host-pathogen protein interactions. *Bioinformatics (Oxford, England)*, 29(13):i217–26, July 2013. ISSN 1367-4811. doi: 10.1093/bioinformatics/btt245.

S. Land and J.H. Friedman. Variable fusion: a new method of adaptive signal regression. 1996.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012.

S. C. Parija and I. Praharaj. Drug resistance in malaria. *Indian J Med Microbiol*, 2011.

P Pongsumpun and P Mumtong. MATHEMATICAL MODEL FOR THE INCUBATION OF THE PLASMODIUM VIVAX MALARIA. *INTERNATIONAL JOURNAL OF APPLIED*, pages 42–48, 2011.

H Ranson, R N'Guessan, and J Lines. Pyrethroid resistance in African anopheline mosquitoes: what are the implications for malaria control? *Trends in Parasitology*, 2011.

RT Rockafellar. *Convex analysis*. 1997.

P Schlagenhauf-Lawlor. *Travelers' Malaria*. Pmph USA Ltd Series. BC Decker, 2007.

Eran Segal, Michael Shapira, Aviv Regev, Dana Pe'er, David Botstein, Daphne Koller, and Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34(2):166–76, June 2003. ISSN 1061-4036. doi: 10.1038/ng1165.

RJ Tibshirani. The lasso problem and uniqueness. *Electronic Journal of Statistics*, pages 1–25, 2013.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, February 2005. ISSN 1369-7412.

Christian Timmann, Thorsten Thye, Maren Vens, Jennifer Evans, Jürgen May, Christa Ehmen, Jürgen Sievertsen, Birgit Muntau, Gerd Ruge, Wibke Loag, Daniel Ansong, Sampson Antwi, Emanuel Asafo-Adjei, Samuel Blay Nguah, Kingsley Osei Kwakye, Alex Osei Yaw Akoto, Justice Sylverken, Michael Brendel, Kathrin Schuldt, Christina Loley, Andre Franke, Christian G Meyer, Tsiri Agbenyega, Andreas Ziegler, and Rolf D Horstmann. Genome-wide association study indicates two novel resistance loci for severe malaria. *Nature*, 489(7416):443–6, September 2012. ISSN 1476-4687. doi: 10.1038/nature11334.

Hans C van Houwelingen, Tako Bruinsma, Augustinus a M Hart, Laura J Van't Veer, and Lodewyk F a Wessels. Cross-validated Cox regression on microarray gene expression data. *Statistics in medicine*, 25(18):3201–16, September 2006. ISSN 0277-6715.

Jeremy West, Dan Ventura, and Sean Warnick. A Theoretical Foundation for Inductive Transfer, 2007.

WHO. World Malaria Report. 2012.

Christian Widmer. Multitask Learning in Computational Biology. pages 207–216, 2012.

J. Zhou, J. Chen, and J. Ye. *MALSAR: Multi-tAsk Learning via StructurAl Regularization*. Arizona State University, 2011.

Jiayu Zhou, Jun Liu, Vaibhav a Narayan, and Jieping Ye. Modeling disease progression via multi-task learning. *NeuroImage*, 78:233–48, September 2013. ISSN 1095-9572.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, April 2005. ISSN 1369-7412.