THE LASSO AND THE MONKEY: FEATURE SELECTION, EXTRACTION, AND TESTING IN REPEATED LOW-DOSE CHALLENGE DATA

Andrew G. Allmon

A dissertation submitted to the faculty at the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Public Health in the Department of Biostatistics in the Gillings School of Global Public Health.

Chapel Hill
2020

Approved by:

Michael G. Hudgens

James S. Marron

Gary G. Koch

Kristina De Paris

Todd A. Schwartz

ABSTRACT

Andrew G. Allmon: The Lasso and the Monkey: Feature Selection, Extraction, and Testing in Repeated Low-Dose Challenge Data
(Under the direction of Michael G. Hudgens)

Progression of technology and computational power have led to a new age in data where the number of variables, p, is greater than the number of observations, n. These types of data, commonly called High-Dimensional Low Sample Size (HDLSS) data, are becoming prominent in statistical applications. One such HDLSS application where small samples are unavoidable is in the development and pre-clinical assessment of new drugs, such as in repeated low-dose challenge (RLC) studies. In RLC experiments, animals are assigned to an active or placebo candidate vaccine and then are repeatedly challenged (exposed) with some target pathogen, either until infection or some maximum number of challenges is reached (Nolen et al., 2015). Many times, the number of animals n in an RLC study is small (e.g. ≤ 20) and number of features p is large (e.g. ≥ 100), due to the high cost of each animal and the high number of antibody and functional measure features of interest (Chaudhury et al., 2018; Choi et al., 2015).

Penalized regression techniques, like the lasso, are sometimes used in RLC experiments where n is typically small and p is large. However, the performance of such methods is not well established for this experiment setting. The performance of the lasso, elastic net, and a newly proposed discrete survival time penalized regression model is assessed via a simulation study. These methods are also applied to a recent RLC study evaluating a candidate HIV vaccine. All three methods rarely selected true positives regardless of the effect size, number of predictors, or

the number of non-zero coefficients, with many models containing only false positives. Thus, penalized regression models should be used cautiously in the RLC setting when n is small and p is large.

To improve upon penalized regression in the RCL setting, a recently-developed high-dimensional test known as the direction-projection-permutation (DiProPerm) test is suggested and adapted to the RLC setting. The DiProPerm test was designed specifically for the HDLSS setting and has many alluring qualities. The DiProPerm test is applied to the RLC setting to test whether animals are more likely to become infected early (i.e., before the median infection time) as opposed to late, given a set of antibody and functional measurements. The DiProPerm test has never been implemented in RLC settings as a valid tool for inference until now. Simulation processes revealed the advantages of using the DiProPerm test on RLC data when n is small and p is large. An RLC study evaluating a candidate HIV vaccine is used to demonstrate the DiProPerm test on a real-world dataset.

To help disseminate the DiProPerm test to researchers, an R package was created. The *diproperm* R package can be used to conduct a DiProPerm test, display corresponding plots of interest, and look at the loadings of the binary linear classifier. The functionality of the *diproperm* package is explained and demonstrated on a real-world data set. The R package is freely available on CRAN and GitHub (https://github.com/allmondrew/diproperm) for anyone to use.

ACKNOWLEDGEMENTS

I am extremely grateful and honored to have been given the opportunity to complete my dissertation for the Biostatistics DrPH program at the University of North Carolina at Chapel Hill. I would like to first give thanks to God for providing me with the peace, encouragement, perseverance, and wisdom to complete my doctoral work.

During my graduate career, I have had the privilege and honor of learning from Dr. Michael Hudgens, my advisor and dissertation committee chair. Dr. Hudgens, thank you for your mentorship, for challenging me, and for believing in me. I would also like to thank the other members of my committee. Dr. Marron, I am grateful for your willingness to help students and your kind personality. I have enjoyed working with you and have learned a lot from you. Dr. Koch, thank you for encouraging and guiding me during one of the toughest times in my graduate career. Your connections and ability to find financial support helped me build valuable professional experience while completing my doctorate. Dr. De Paris, thank you for helping me to improve my research by sharing your expertise and for swiftly replying to my emails. Dr. Schwartz, thank you for serving on my committee and for providing feedback on my dissertation.

I would also like to acknowledge the financial support I received. Thank you to Dr. Hudgens, the Center for AIDS Research, and the NIH for providing funding for several years so

I may gain experience and focus on schoolwork. Thank you, UCB Biosciences and United Therapeutics for also sponsoring me and providing real-world experience.

I would now like to thank my family, especially my wife, Ahrang, mother and father, brother, grandparents, and cousins who believed in me and encouraged me. Thanks, Ahrang for your prayers, patience, and love which I needed during the good times and the bad. Thanks mom and dad for your love, support, and prayers as well. Both of you have sacrificed so much for me to be here right now and for this I am eternally grateful. Thanks, Jojo and Buck for your love and prayers as well and thank you Armentrout's for your support and prayers too. Lastly, I would like to thank my in-laws for also praying and supporting me during graduate school. I love you all very much and cannot wait to share my future experiences with you.

I would also like to thank all the friends I have come to know during my time at UNC and those before UNC. Thank you Tony, Garrett, Hunter, Sam, Jonathon, and Keith for your friendship, support, prayers, and for making me realize the importance of good friendships. Thanks to my Grace Community Church/Waypoint Church family, Focus, Chris, Ting-An, and small group for your prayers, support, and advice. Without all of you, this degree would have not have been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BIC	Bayesian Information Criterion
DiProPerm	Direction-Projection-Permutation
DWD	Distance-Weighted Discrimination
HDLSS	High-Dimensional Low Sample Size
LASSO	Least Absolute Shrinkage and Selection Operator
LDA	Linear Discriminant Analysis
MSE	Mean Squared Error
PCA	Principal Component Analysis
RLC	Repeated Low-Dose Challenge
SVM	Support Vector Machine

CHAPTER 1: LITERATURE REVIEW

Advancements in modern technology and computer software have dramatically increased the demand of and feasibility of collecting colossal, high-dimensional data sets. Because of this, the desire for novel statistical techniques for high-dimensional data, or big data, has never been higher. High-dimensional data present a plethora of challenges that require the creation and adaptation of new and existing statistical methods. One such challenge is that big data often has many more predictors, p, than the number of observations, n. However, it is becoming increasingly popular in biomedical research to collect data on an immense number of variables in relation to a small sample size. These data structures are known as high-dimensional, low sample size (HDLSS) data sets, or "small n, big p."

High-dimensional, low sample size data sets emerge frequently in many health-related fields. For example, in genomic studies, a single microarray experiment might produce tens of thousands of gene expressions compared to the few samples studied, often less than a hundred (Alag, 2019). In medical imaging studies, a single region of interest for analysis in an MRI or CT-scan image often contains thousands of features compared to the small number of samples studied (Limkin et al., 2017). In pre-clinical evaluations of vaccines and other experimental therapeutic agents, the number of biomarkers measured (e.g., immune responses) may be much greater than the number of samples studies (e.g., mice, rabbits, or non-human primates) (Kimball et al., 2018).

Even though we live in an era of big data for biomedical research, there are many applications where small samples in pre-clinical and human assessment are unavoidable, such as

in the development of new drugs and vaccines (Aban & George, 2015). One such application is repeated low-dose challenge (RLC) studies. In RLC experiments, animals who are assigned to an active or placebo candidate vaccine are repeatedly challenged (exposed) with some target pathogen, either until infection or until some maximum number of challenges is reached (Nolen et al., 2015). Since the maximum number of challenges is typically specified *a priori,* RLC studies can be modeled using a discrete survival time model where each challenge is thought to be one discrete time point. Many times, the number of animals in an RLC study is small (e.g. ≤ 20) and the number of features is large (e.g. ≥ 100) due to the high cost of each animal and the high number of antibody and functional measure features of interest (Chaudhury et al., 2018; Choi et al., 2015).

The analysis of these types of HDLSS data sets often requires the creation of new methods or alterations to existing methods. Many traditional methods for low dimensional data are not appropriate for HDLSS data. One major reason these methods are inappropriate is the insufficient number of samples to adequately estimate the underlying covariance. Because of the sheer size of HDLSS data sets, it is of great interest to develop methodology that can select relevant features associated with the outcome of interest (i.e., feature selection) or reduce dimensionality by condensing many features into several features without the loss of much information (i.e., feature extraction). A review of popular feature selection and feature extraction methods follows in the next section.

1.1 Feature Selection Methods

The two most common selection procedures include the least absolute shrinkage and selection operator, or lasso, and the elastic net. In the lasso, an L1 penalty constrains the coefficient estimates in such a way that variables with little to no effect on the outcome of interest "shrink" to zero. The lasso was later improved by the elastic net, which includes a

penalty parameter to control the number of L1 and L2 penalties on the coefficients. Since penalized regression techniques "shrink" small-effect estimate coefficients to zero, these methods are heavily used for variable selection in small n large p study designs. Recently, the use of these penalized regression techniques has become popular for antibody and functional measure feature selection in assessing the performance of candidate vaccines in pre-clinical HIV studies, particularly in repeated low-dose challenge experiments. However, the performance of these techniques on discrete survival time models, such as those in RLC experiments, is not well established. Nevertheless, scientists continue to use the lasso and elastic net in discrete survival time settings. In the original lasso paper, Tibshirani (1996) provided an option for continuous survival time models in the *glmnet* R package, but did not provide an option for discrete survival time models. Recently, a new method, denoted $\text{glmLasso}_{\text{dis}}$, was proposed for variable selection in discrete survival models and included a penalty term on the baseline hazard function (Groll & Tutz, 2017). More details on the $\text{glmLasso}_{\text{dis}}$ method can be found in Chapter 2.

1.1.1 The Lasso and the Elastic Net

To better explain the use of the lasso and elastic net, consider the general linear regression scenario. For n observations and p predictors, let y_{nx1} be a size n column vector of response values, x_{nxp} be an n by p matrix of covariates, and $\beta_{px1} = (\beta_1, \dots, \beta_p)$ be a column vector of coefficients. For simplicity, do not consider the intercept term in the coefficient vector. Based on the model $y = x\beta + \epsilon$, the elastic net finds $\hat{\beta} = (\hat{\beta}_1, \dots, \hat{\beta}_p)$, which solves the objective function

$$\min_{\beta \in \mathbb{R}^p} \left[\frac{-1}{2n} \sum_{i=1}^{n} (y_i - x_i\beta)^2 + \lambda \sum_{j=1}^{p} \left(\frac{1}{2}(1-\alpha)\beta_j^2 + \alpha|\beta_j| \right) \right]$$

where $\alpha \in [0,1]$ and λ is the tuning parameter. For logistic regression, $y \sim Bernoulli\left(\frac{e^{x\beta}}{1+e^{x\beta}}\right)$, the elastic net will find $\hat{\beta} = \left(\hat{\beta}_1, \ldots, \hat{\beta}_p\right)$, which solves the objective function

$$\min_{\beta \in \mathbb{R}^p} \left[\frac{-1}{n} \sum_{i=1}^{N} \left(y_i x_i \beta + \log\left(1 + e^{x_i \beta}\right) \right) + \lambda \sum_{j=1}^{p} \left(\frac{1}{2}(1-\alpha)\beta_j^2 + \alpha \left| \beta_j \right| \right) \right]$$

The tuning parameter, λ, determines the intensity of penalization on coefficients while α controls the level of L1 and L2 penalties.

The elastic net is a generalization of both ridge regression, where $\alpha = 0$, and the lasso, where $\alpha = 1$. Ridge regression shrinks coefficient estimates towards zero, but they are never equal to zero unless $\lambda = \infty$ (Hoerl & Kennard, 1970). Therefore, ridge regression will always select p predictors in the final model. This may not be a problem if prediction accuracy is the main objective, but it can complicate model interpretation in studies in which the number of variables is very large, such as HDLSS problems. The lasso avoids this major limitation of ridge regression (Tibshirani, 1996). The lasso's L1 penalty, $\lambda \sum_{j=1}^{p} |\beta|$, does have the ability to shrink coefficient estimates to zero when the tuning parameter λ is large enough. Therefore, selecting the tuning parameter, λ, is crucial. Often, K-fold cross-validation is used for selecting the λ, which minimizes the mean squared error (MSE). Other measures have also been used for selecting λ, such as AIC, BIC, or RIC (Akaike, 1998; Foster & George, 1994; Schwarz, 1978). Regardless, the lasso results in a sparser model than ridge regression and provides a more interpretable result when the number of variables is large. Zou and Hastie (2005) improved the lasso with the elastic net, which incorporates both the L1 from lasso and L2 from ridge penalties in the objective function (Zou & Hastie, 2005). Each penalty's contribution to the objective function is determined by $\alpha \in [0,1]$.

The elastic net has several advantages over the lasso, especially when $p \gg n$. For instance, the lasso is limited by the sample size, n, selecting at most n variables before saturation, while the elastic net is not limited by this constraint. In addition, if a group of predictors are highly correlated with each other, the lasso will typically select one variable from the group, often not caring about which variable is selected. However, the elastic net has the ability to select entire groups of highly-correlated variables and produce models with better predictive performance than the lasso.

1.2 Feature Extraction Methods

Albeit feature selection is a major topic of interest for HDLSS data, feature extraction is also of great interest. Feature extraction is the idea of condensing many features into fewer features while still capturing most of the relevant information. Feature extraction can be classified into two categories: unsupervised or supervised learning. Unsupervised learning classifies training data into categories without the use of a labeled data set, while supervised learning classifies training data into categories with the help of a labeled data set. For this dissertation, general overviews of principal component analysis (PCA) and linear discriminant analysis (LDA) methods will be covered. Then, we will explain a machine-learning technique known as the DiProPerm test, which we apply to the RLC paradigm in chapter 3.

1.2.1 PCA and LDA

Two common feature extraction methods include principal component analysis and linear discriminant analysis. Principal component analysis (Jolliffe, 1986) is an unsupervised learning technique that creates linear combinations of the original features. Each linear combination is said to be one principal component. The principal components are ranked in order of how much variation they explain in the original data. The first principal component, PC1, explains the most variance, followed by the second, and so on. Thus, one can reduce the dimensionality of the

original data by selecting the principal components subject to a desired cumulative amount of variance explained by the components. For instance, one may desire to keep only the principal components that cumulatively explain 80% of the original data. Normalizing the data before performing PCA is crucial because PCA is heavily sensitive to scale. Otherwise, large-scaled variables, such as cell count, will over represent the principal components. Another strength of PCA is that the linear combinations are uncorrelated with each other due to the orthogonality between components. Principal component analysis is not without limitations, though. Because each principal component is a linear combination of original features, the interpretation of the components is not clear. Also, PCA is sensitive to outliers, especially when the sample size is small (Aoshima et al., 2018). To improve the interpretability and robustness of PCA, sparse PCA was proposed (Zou et al., 2006). Sparse PCA reformulates PCA as a regression-type optimization problem and then incorporates an elastic net penalization to create sparse components. Therefore, sparse PCA not only reduces dimensionality, as does PCA, but also reduces the number of loadings, which are the coefficients in the linear combination, for interpretability.

Linear discriminant analysis (Fisher, 1936) is a supervised learning technique that also utilizes linear combinations of original features. However, LDA focuses on the separability between linear combinations instead of the cumulative explained variance. Linear discriminant analysis has become a popular tool for supervised classification because of its predictive accuracy, simplicity, and robustness (Hand, 2006). However, LDA does not perform well when $p \gg n$ because the within-class covariance of p is singular and not estimable. Also, LDA assumes that the separation in the data can be described well with linear boundaries, which is often not the case in $p \gg n$ problems. Because of these limitations, LDA is often not appropriate

in HDLSS settings. Linear discriminant analysis was later adapted to $p \gg n$ problems with the creation of sparse LDA (Clemmensen et al., 2011). Sparse LDA has the ability to perform feature selection and classification simultaneously by incorporating an elastic net penalty to the discriminant problem. Sparse LDA does improve the interpretability of $p \gg n$ problems. It reduces overfitting but has certain limitations. For instance, in sparse LDA, there is no optimal procedure for selecting the penalty parameters since the optimization problem is often not convex (Y. Wu et al., 2015). Also, when sample sizes are very low, estimates for the within-class covariance will often be biased or unstable (Cai & Liu, 2011).

Although PCA and LDA have evolved to help accommodate several $p \gg n$ problems, it would be helpful to have a procedure designed specifically for the HDLSS setting, which could be adapted to the RLC paradigm. One such method proposed for inference on HDLSS data is the direction-projection-permutation (DiProPerm) test. The DiProPerm test incorporates elements from PCA and LDA but is an exact test for HDLSS data, even in small samples.

1.3 DiProPerm

In analyses of HDLSS data, a common task is to assign data to the correct class by building a function that uses the class labels, known as a classifier. Classifiers that use two labels and a linear combination of features are known as binary linear classifiers. There are many classifiers for use in HDLSS data, such as random forests or neural networks, but these kinds of classifiers are complicated and difficult to interpret. Binary linear classifiers are popular for their simplicity and interpretability; larger coefficients translate to a more direct impact in the separation of the two classes. However, linear classifiers such as LDA have been known to suffer from what is called "data piling" in the HDLSS setting (Marron et al., 2007). Data piling occurs if data are projected onto some projection direction and many of the projections are the same, or

piled on one another. For instance, there could be a case where data is sampled from two identical high-dimensional distributions, but the binary linear classifier could find a linear combination of features such that the two classes are not identical.

The DiProPerm was developed to test whether or not a binary linear classifier detected a statistically significant difference between two high dimensional distributions. DiProPerm uses one-dimensional projections, or linear combinations, on the binary linear classifier, and then uses these projections to construct a permutation distribution to test whether the two distributions are different.

To better understand the mechanics of DiProPerm, let $U = U_1, \dots, U_n \sim F_1$ and $V = V_1, \dots, V_m \sim F_2$ be independent random samples of p dimensional random vectors from multivariate distributions F_1 and F_2 where $p \gg n, m$. The DiProPerm tests

$$H_0: F_1 = F_2 \text{ versus } H_1: F_1 \neq F_2$$

The general idea of the DiProPerm test can be explained in three steps.

1. Direction: Find the normal vector to the separating hyperplane between two samples after training a binary linear classifier.
2. Projection: Project data on to the normal vector and calculate a univariate two-sample statistic.
3. Permutation: Conduct a permutation test using the univariate statistic as follows:
    a. permute class membership after pooling samples,
    b. re-train the binary classifier and find the normal vector to the separating hyperplane,
    c. recalculate the univariate two sample statistic,

d. repeat a-c multiple times (e.g., 1,000) to determine the sampling distribution of the test statistic under the null H_0, and

e. compute p-value by comparing the observed statistic to the sampling distribution.

Different binary linear classifiers may be used in the first step of DiProPerm. Linear discriminant analysis, particularly after conducting principal component analysis, is one possible classifier for the direction step. However, using LDA with PCA in the HDLSS setting has some disadvantages, including a lack of interpretability, a sensitivity to outliers, and a tendency to find spurious linear combinations due to a phenomenon known as data piling (Aoshima et al., 2018; Marron et al., 2007). The support vector machine (SVM) is a another popular classifier (Hastie et al., 2001). The SVM finds the hyperplane that maximizes the minimum distance between data points and the separating hyperplane. However, the SVM can also suffer from data piling in the HDLSS setting. To overcome data piling, the distance-weighted discrimination (DWD) classifier was developed (Marron et al., 2007). The DWD classifier finds the separating hyperplane minimizing the average inverse distance between data points and the hyperplane. The DWD performs well in HDLSS settings with good separation and is more robust to data piling.

In the second step of DiProPerm, a univariate statistic is calculated using the projected values on to the normal vector to the separating hyperplane from the first step. Suppose $u_1, \dots, u_n$ and $v_1, \dots, v_m$ are the projected values from samples U and V respectively. One common choice for the univariate test statistic for DiProPerm includes the difference of means statistic: $|\bar{u} - \bar{v}|$. Other two-sample univariate statistics such as the two-sample t-statistic or the difference in medians are also possible for use with the DiProPerm.

The last step of the DiProPerm entails determining the distribution of the test statistic under the null hypothesis. In this step, the two samples are pooled, class labels are permuted, and

then a univariate statistic is calculated. Repeat this process multiple times (e.g., 1,000) to determine the sampling distribution of the test statistic under the null H_0. P-values are then calculated by the proportion of statistics higher than the original value.

When the DiProPerm test is implemented using the DWD classifier, it is common practice to look at the loadings of the DWD classifier (An et al., 2016; Nelson et al., 2019). The DWD loadings represent the relative contribution of each variable to the class difference. A higher absolute value of a variable's loading indicates a greater contribution for that variable to the class difference. Combining the use of the DiProPerm and evaluation of the DWD loadings in applications can provide insights into high-dimensional data and be used to generate rational hypotheses for future research.

In RLC studies, one research question can be generally stated as follows: for binary classification, given a new sample with functional measurements, can one predict whether an animal will become infected with the target pathogen early versus late? In Chapter 3, we answer this question using the DiProPerm test to compare individuals who were infected before the median time to infection versus those who were infected after the median. The DWD loadings are then assessed to pinpoint certain variables that drove the difference between early infection versus late. A user-friendly software tool is then introduced for researchers conducting a DiProPerm test.

1.4 Motivating Data Example

A motivating example used throughout the dissertation is the MIV02 data, a repeated low-dose challenge study conducted in collaboration with UNC Chapel Hill, Duke, and UC Davis (Eudailey et al., 2018). The MIV02 dataset consists of $n = 14$ rhesus monkey infants assigned to two different treatment groups: $n = 7$ on an HIV vaccine and $n = 7$ on a control

vaccine. Infants in each group were challenged weekly with 20 TCID50 of SHIV1157ipd3N4 with up to seven challenges until infection. If an infant remained uninfected after seven challenges, the dose was increased to 40 TCID50. Infants were initially challenged at six weeks of age (i.e., baseline). The maximum number of challenges an infant could receive was 20 challenges. At the end of the study, 12 of the 14 primates became infected with SHIV.

To create an analysis data set from the MIV02 data, several MIV02 data sets consisting of blood and genotype data were merged together into an $n = 14$ by $p = 50$ data set, where p is the number of week 6 pre-challenge covariates of interest without missing data. Categorical variables were coded as dummy variables for each category. After dummy variable coding, the final analysis data set was $n = 14$ by $p = 138$.

1.5 Outline

In Chapter 2, we conduct a simulation study to assess the selection performance of penalized regression techniques on HDLSS data with application to an RLC study. Many pre-clinical studies use the lasso to make associations with certain antibody measures with disease infection status. We suggest these techniques should not be used to make claims about association with certain antibody measures and HIV infection. Instead, we suggest these methods be used as hypothesis-generating methods for larger future experiments.

In Chapter 3, we explore the use of the DiProPerm test on RLC data. The type I error and power of the DiProPerm test on RLC experiments are described in a simulation study. The DiProPerm test is then adapted to the MIV02 study to test whether non-human primates are more likely to become infected early (i.e., before the median infection time) as opposed to late, given a set of antibody and functional measurements. In addition, we evaluate the DWD loadings from the test to observe which variables had the most influence on median time to HIV infection. The DiProPerm has never been implemented in the RLC paradigm as a valid tool for inference until

now. Simulation processes and real data applications reveal the advantages of the DiProPerm test on RLC data. The DiProPerm will help medical professionals conducting pre-clinical experiments in RLC studies to make better claims on which type of functional measures help prolong infection time.

In Chapter 4, we introduce an R package software tool for the analysis of RLC studies using DiProPerm. The *diproperm* R package is a user-friendly computational tool built for use by medical investigators with little coding experience. A demonstration for how to use the R package is explained, and the tool is used on a real-world data set. The R package can be used to conduct a DiProPerm test, display corresponding plots of interest, and look at the loadings of the binary linear classifier. The *diproperm* R package is freely available on CRAN and GitHub (https://github.com/allmondrew/diproperm) for anyone to use. All analyses for this dissertation, unless otherwise stated, were conducted in R version 3.6.1.

CHAPTER 2: PENALIZED REGRESSION TECHNIQUES IN SMALL-SAMPLE DISCRETE SURVIVAL TIME MODELS

2.1 Introduction

In the growing age of big data, there is increasing demand for the use of model selection techniques on high dimensional data. The least absolute shrinkage and selection operator, or lasso, is one such technique (Tibshirani, 1996). In the lasso, an L1 penalty constrains the coefficient estimates in such a way that variables with little to no effect on the outcome of interest "shrink" to zero. The lasso was later improved by the elastic net, which includes a penalty parameter to control the amount of L1 and L2 penalties on the coefficients. Including both the L1 and L2 penalties allows the elastic net to reduce the number of features selected and reduce the coefficients that are not important in predicting the outcome to improve the model's prediction over the lasso. Since penalized regression techniques "shrink" small-effect estimate coefficients to zero, these methods are often used for variable selection where the number of samples is small and the number of features is large. Recently, the use of these penalized regression techniques has become popular for antibody and functional measure feature selection in assessing the performance of candidate vaccines in pre-clinical HIV studies, particularly in repeated low-dose challenge experiments (Ackerman et al., 2018; Bradley et al., 2017; Chaudhury et al., 2018; Tomaras & Plotkin, 2017; Vaccari et al., 2016).

In repeated low-dose challenge experiments, animals are assigned to an active or placebo candidate vaccine and then are repeatedly challenged (exposed) with some target pathogen, either until infection or until some maximum number of challenges is reached (Nolen et al.,

2015). Since the maximum number of challenges is specified *a priori,* RLC studies can be modeled using a discrete survival time model where each challenge is thought to be one discrete time point. Many times, the number of animals in an RLC study is small (e.g. ≤ 20) and the number of features is large (e.g. ≥ 100) due to the high cost of each animal and the high number of antibody and functional measure features (Chaudhury et al., 2018; Choi et al., 2015).

Despite the popularity of the lasso and the elastic net, the performance of these techniques on discrete survival time models is not well established. However, scientists continue to use the lasso and elastic net in discrete survival time settings, such as repeated low-dose challenge studies. In the original lasso paper, Tibshirani et al. (1996) provided an option for continuous survival time models in the *glmnet* package, but they have not yet provided an option for discrete survival time models. Groll and Tutz (2017) proposed a model for variable selection in discrete survival models by including a penalty on the baseline hazard function. This model can be found in the R package *glmmLasso* (Groll & Tutz, 2014).

In this paper, we perform a simulation study to assess the variable selection performance of the lasso, the elastic net, and the method proposed by Groll and Tutz in discrete survival model settings. Data from a repeated low-dose challenge study is used as an example of application in real-world study designs. Section 2.2 introduces the notation and the methods used for quantifying penalized regression performance across the three scenarios. Section 2.3 explains the simulation experiments conducted for assessing performance and summarizes the results from every simulation scenario. Section 2.4 demonstrates the application of the three scenarios on a real-world RLC data set. Section 2.5 discusses the future implications and limitations of the performance assessment and provides closing remarks and a summary of the entire paper.

2.2 Methods

Consider a repeated low-dose challenge study with n animals. Each animal is repeatedly challenged with a pathogen of interest (e.g., simian HIV). After each challenge, the animal is assessed for infection. If an animal is uninfected, the challenges continue; otherwise, the challenges cease. Data from such studies is naturally handled in a discrete time survival analysis framework. In particular, $\tilde{T}_i$ denotes the number of challenges until infection if the animal was challenged indefinitely. In practice, challenges typically cease for uninfected animals after a set number of challenges, say c_{max} (in general, c_{max} may differ between animals, but for simplicity here, it is assumed to be same across animals). Thus, the discrete survival time $\tilde{T}_i$ may be right censored. That is, instead of observing $\tilde{T}_i$, we observe $T_i^{obs} = min(\tilde{T}_i, c_{max})$ as well as the event indicator $Y_i = I(\tilde{T}_i \leq c_{max})$. In addition, for each animal, we observe $X_i = (X_{i1}, \dots, X_{ip})$, a vector of p baseline covariates. The inferential goal is to characterize the extent to which one or more of the baseline covariates $X_{i1}, \dots, X_{ip}$ are associated with the time until infection $\tilde{T}_i$. Below, three methods are considered.

The first method uses a parametric discrete time survival model. Define the discrete time hazard function by

$$h(\tilde{T}_i = t | \tilde{T}_i \geq t, X_i = x_i) = P(\tilde{T}_i = t | \tilde{T}_i \geq t, X_i = x_i) \text{ for } t = 1, \dots, c_{max}$$

that is, the conditional probability for becoming infected given the animal has not been infected before time t. In general, models for discrete survival problems, given covariate vector x_i, have the form

$$h(\tilde{T}_i = t | \tilde{T}_i \geq t, X_i = x_i) = g(\gamma_{0t} + x_i\beta)$$

where $\beta = (\beta_1, \dots, \beta_p)^T$ is a column vector of coefficients and $g(\cdot)$ is an invertible function mapping from $(-\infty, \infty)$ to $[0, 1]$. For example, $g(\cdot)$ could be the inverse logit function, that is,

$$h\left(\tilde{T}_i = t \middle| \tilde{T}_i \geq t, X_i = x_i\right) = \frac{e^{\gamma_{0t}+x_i\beta}}{1+e^{\gamma_{0t}+x_i\beta}} = logit^{-1}(\gamma_{0t} + x_i\beta) \quad (1)$$

where $logit^{-1}(\gamma_{0t})$ represents the baseline hazard function at time t corresponding to animals with covariates $x_i = (0, \dots, 0)$.

A challenging aspect of modern RLC studies is that the vector of baseline covariates X_i may include a large number of covariates relative to the study's sample size. That is, p may be large compared to n. One common technique for analyzing such data is known as penalized regression. In penalized regression, the objective is to build a parsimonious predictive model with a small number of non-zero estimated regression coefficients. To achieve this objective, a penalty is included while minimizing the negative log-likelihood to determine the estimated regression coefficients, $\hat{\beta}$. In the lasso, an L1 penalty constrains $\hat{\beta}$ in such a way that covariates with little to no association with the outcome of interest "shrink" to zero. On the other hand, the elastic net incorporates both L1 and L2 penalties. Groll and Tutz (2017) proposed fitting the discrete time model (1) via penalization by adding an additional penalty term to the lasso for the baseline hazard, finding $\hat{\beta} = \left(\hat{\beta}_1, \dots, \hat{\beta}_p\right)^T$ and $\hat{\gamma} = \left(\hat{\gamma}_1, \dots, \hat{\gamma}_q\right)$, which solves

$$\min_{\beta\in\mathbb{R}^p, \gamma\in\mathbb{R}^q} \left[-\frac{1}{m}\sum_{i=1}^{n}\sum_{t=1}^{T_i^{obs}} \left\{ Y_{it}(\gamma_{0t} + x_i\beta) + \log\left(1 + e^{\gamma_{0t}+x_i\beta}\right) \right\} + \lambda \sum_{j=1}^{p} |\beta_j| + \nu \sum_{t=1}^{q} \gamma_{0t}^2 \right] \quad (2)$$

where $m = \sum_{i=1}^{n} T_i^{obs}$ is the total number of challenges across all animals, $Y_{it} = I(\tilde{T}_i = t)$ is the event indicator of the i^{th} animal at timepoint t, λ is the tuning parameter for the β penalty term, ν is the tuning parameter for the baseline hazard penalty term, and $q = \max\left\{T_i^{obs} : i = 1, \dots, n\right\}$ is the maximum observed number of challenges across all animals (Groll & Tutz, 2017). Note that if at least one animal survives up to c_{max}, then $q = c_{max}$. Groll and Tutz recommend choosing λ by minimizing the Bayesian information criterion (BIC) (Schwarz, 1978) across a set of 100 λs and selecting ν a priori.

The second method considered is penalized logistic regression, where each challenge is treated as a separate observation. In particular, for logistic regression, that is, $P(Y_t = 1|X = x) = logit^{-1}(\beta_0 + x\beta)$, the elastic net penalized regression approach finds $\hat{\beta} = \left(\hat{\beta}_1, \dots, \hat{\beta}_p\right)^T$ and $\hat{\gamma} = \left(\hat{\gamma}_1, \dots, \hat{\gamma}_q\right)$, which solves

$$\min_{\beta \in \mathbb{R}^p, \gamma \in \mathbb{R}^q} \left[-\frac{1}{m} \sum_{i=1}^{n} \sum_{t=1}^{T_i^{obs}} \left\{ Y_{it}(\gamma_{0t} + x_i\beta) + \log\left(1 + e^{\gamma_{0t} + x_i\beta}\right) \right\} + \lambda \sum_{j=1}^{p} \left\{ \frac{1}{2}(1-\alpha)\beta_j^2 \alpha |\beta_j| \right\} \right] \quad (3)$$

where $\alpha \in [0,1]$ is the elastic-net mixing parameter chosen a priori and λ is chosen via K-fold cross validation. More information for how λ was calculated for our simulations is included in Section 2.3.

A third method that could be used in this setting entails fitting a continuous time Cox model via elastic net penalization. Specifically, we find $\hat{\beta} = \left(\hat{\beta}_1, \dots, \hat{\beta}_p\right)^T$ which solves

$$\min_{\beta \in \mathbb{R}^p} \left[-\frac{1}{n} \sum_{t=1}^{c_{max}} \left\{ \sum_{j \in D_t} x_j\beta + d_t \log\left(\sum_{j \in R_t} e^{x_j\beta} \right) \right\} + \lambda \sum_{j=1}^{p} \left\{ \frac{1}{2}(1-\alpha)\beta_j^2 + \alpha|\beta_j| \right\} \right] \quad (4)$$

where R_t is the set of indices for those individuals at risk at time t. Ties are handled using Breslow's approximation, where D_t is the set of indices of individuals who fail at time t, and $d_t = |D_t|$ is the number of individuals who fail at time t (Breslow, 1974). λ is chosen in the same way as for (3) and is described in more detail below.

The operating characteristics of approaches (2), (3), and (4) on RLC data when n is small and p is large were evaluated via a simulation study. For the simulation study, all computations were performed in R 3.6.1. The package *glmmLasso* was used to implement the discrete survival selection approach (denoted $\text{glmLasso}_{\text{dis}}$) for approach (2) (Groll, 2017). The R package *glmnet* was used to implement the lasso and elastic net for approaches (3) and (4) (Friedman et al., 2010; Simon et al., 2011). Our simulations only assessed $\alpha = 0.5$ or $\alpha = 1$ for the elastic net and lasso respectively. In the glmnet function call, the parameter family was set to either "binomial" for

logistic regression or "cox" for Cox regression. For logistic and Cox regression, 10-fold cross-validation was used to determine the value of λ, chosen to be either the value that minimized the mean squared error (MSE) of the predicted responses or the maximum value of λ corresponding to an MSE within one standard error (1SE) of the minimum. Multiple observations of the same subject were constrained to stay together in each fold during cross validation. For the remainder of this paper, λ_{minMSE} and λ_{1SE} correspond to methods that use a penalty term based on minimum MSE or within 1SE. For glmLasso$_{dis}$ methods, λ was chosen such that the BIC was minimized across a set of 100 lambdas (Schwarz, 1978) and $\nu = 20$ was chosen as was recommended (Groll & Tutz, 2017).

2.3 Simulation

A simulation study was conducted to explore the effects of varying the sample size, n; the tuning parameter, α; true coefficient parameters, β; the number of non-zero predictors, k; and the number of total predictors, p, in (2), (3), and (4). Values of $p = 50, 100, 500$; $n = 10, 20, 30$; and $k = 1, 3, 5, 7$ were used across simulations. The maximum number of challenges was $c_{\max} = 10$ and the number of simulations was 500 for each method.

In order to represent realistic RLC scenarios, the baseline hazard and parameter coefficients were chosen in such a way that the average probability of infection per exposure was between 0.2 and 0.3. The various combinations of parameter coefficients and non-zero coefficients can be seen in Section 2.7. For $j = 1, \dots, p$, covariate vectors $X_i = (X_{i1}, \dots, X_{ip})$ were drawn from a multivariate normal (MVN) distribution with $\mu = 0$ and covariance matrix Σ with 1s along the diagonal and 0.05 along the off-diagonals.

The RLC data were simulated as follows.

For each $i = 1, \dots, n$,

1) Sample X_i from $MVN(0, \Sigma)$.
2) For $t = 1, \ldots, c_{max}$, sample binary event Y_{it} from $Bern\big(h(t|x_i)\big)$.
    a. If $Y_{it} = 1$, then set $T_i^{obs} = t$ and stop.
    b. Otherwise, if $Y_{it} = 0$ and $t = c_{max}$, then set $T_i^{obs} = c_{max}$ and stop.
    c. Otherwise, increment t by 1 and repeat step 2.

For approaches (2) and (3), data were structured in such a way that each challenge is an observation, that is, $X_{m\,x\,p}$where $m = \sum_{i=1}^{n} T_i^{obs}$, the total number of challenges across n animals. For (4), each observation was one animal, that is, $X_{n\,x\,p}$. For simplicity, a uniform baseline hazard, $logit^{-1}(\gamma_{0t})$, was used for all t in (2), (3), and (4) such that the mean infection probability per exposure was between 0.2 and 0.3.

Small, moderate, and large effect true parameter coefficients, β, were used for each simulation, corresponding to odds ratios of approximately 1.2, 1.5, and 2 respectively. For the purpose of this paper, we will focus on the moderate effect scenario. Small and large effect scenarios can be seen in the supplemental information document. Table 2.1 shows the true coefficients for the moderate effect models and Supplemental Table 2.1 displays the true coefficients of small and large effect models. For Table 2.2 and Supplemental Tables 2.2–2.13, a true positive was defined as a variable with a non-zero coefficient in both the true model and in the predicted model. A false positive was defined as a variable with a coefficient of zero in the true model and a non-zero coefficient in the predicted model. In addition, in Table 2.2, the percent containing true is the percentage of simulations that included all the possible true positives while potentially selecting a few false positives, whereas the percent equal to true is the percentage of simulations that selected all the true positives and no false positives. The supplemental tables do not include the percent containing true or the percent equal to true

because this information was non-informative for most models, often being less than one percent as p and k increased. Instead, Supplemental Tables 2.2–2.13 display the average number of true positive, average number of false positives, and percent of models with only false positives for (2), (3), and (4). Figure 2.1 shows the average number of true positives against the average number of false positives when $k = 1$, while Supplemental Figures 2.1–2.3 show for when $k = 3$, 5, and 7. In Figure 2.1, the glmLasso$_{dis}$ consistently selected the highest number of true positives without selecting a large number of false positives. This is also true for Supplemental Figures 2.1–2.3. Because of this, the glmLasso$_{dis}$ can be viewed as a compromise amongst all the methods in this paper.

For (3) and (4), the elastic net models selected more true positives and false positives on average than the lasso models, and λ_{minMSE} models included more true positives and more false positives on average than λ_{1SE} models. λ_{minMSE} models also had a higher percentage of containing only false positives compared to λ_{1SE} models on average. As the sample size increased, the average number of true positives selected in both the lasso and elastic net increased for (3) and (4). However, for (3), the average number of false positives selected largely increased for λ_{minMSE} models compared to λ_{1SE} models as sample size increased. For (4), there were no consistent trends for the average number of false positives selected between λ_{minMSE} and λ_{1SE} models relative to sample size.

For (2), (3), and (4), as the total number of predictors increased, the average number of true positives selected decreased, whereas the average number of false positives and the percentage of models containing only false positives increased. However, as the sample size increased, the average number of true positives selected increased for (2), (3), and (4) while the percentage of models selecting only false positives decreased. Also, as sample size increased, the

average number of false positives selected decreased for (3), but there were no consistent trends for the average number of false positives selected for (2) and (4) relative to sample size. The average number of true positives, false positives, and percentage of models containing only false positives decreased, on average, as the number of non-zero coefficients, k, increased for (2), (3), and (4).

2.4 Application

In this section, a motivating example is given using the MIV02 data mentioned in section 1.5. Results for applying (2), (3), and (4) on MIV02 data can be seen in Table 2.3. Index 27 was selected most often across all methods, and 85, 99, and 138 were the second most selected. Index 27 corresponds to activation CD69 total, while 85 is the "23_02" group from MamuDQA Haplotype 1, 99 is the "18g2" group from MamuDQB Haplotype 1, and 138 is the "TFP/TFP" group from the TRIM5 genotype. Figure 2.2 displays the raw data for CD69 by the number of challenges for the 14 primates. From the MIV02 study, CD69 was shown to be an early T-cell activation marker for SHIV, and the TRIM5 genotype TFP/TFP has been shown to confer resistance to SHIV (F. Wu et al., 2016). However, MamuDQA Haplotype I and MamuDQB Haplotype I have never been shown to be associated with SHIV infection. MamuDQA Haplotype I and MamuDQB Haplotype I could be new markers, but it is possible that in including every single allele marker in the analysis data set, some random associations were detected considering the small group sizes. Therefore, one might want to formulate future hypotheses for studies assessing the effects of the MamuDQA Haplotype I and MamuDQB Haplotype I allele on risk of HIV infection.

2.5 Discussion

Penalized regression techniques are regularly utilized for high-dimensional data variable selection, and have been increasingly used for small-sample discrete survival time models,

particularly RLC studies (Ackerman et al., 2018; Bradley et al., 2017; Chaudhury et al., 2018; Tomaras & Plotkin, 2017; Vaccari et al., 2016). To our knowledge, no prior research has been conducted for assessing how the performance of penalized regression techniques changes relative to the number of observations and predictors in small-sample discrete survival models. In this article, simulated data show that the lasso, elastic net, and glmLasso$_{dis}$ methods have a low probability of selecting true positives, especially as the number of predictors increases.

For approaches (3) and (4), the predicted models for the lasso and elastic net were dominated by false positives compared to true positives. However, approach (2) was not so dominated by false positives relative to true positives. Also, approach (2) selected fewer false positives on average than Cox and logistic lasso and elastic net models but selected fewer true positives on average as sample size decreased and the number of predictors increased. Additionally, λ_{minMSE} models selected more true positives on average than λ_{1SE} and glmLasso$_{dis}$ models but were plagued with false positives, with the average number of false positives increasing as p and n increased. One of the more alarming results is that as sample size decreased, effect size decreased, and as the number of predictors increased, lasso models selected virtually no true positives on average. This realization demonstrates that the lasso is not appropriate for RLC studies and a high level of confidence cannot be placed on the associations discovered from the lasso. Continuing to use the lasso in RLC studies could not only waste resources to test false positives, but true associations could be undetected, which would harm future research.

The sample size needed for detecting all true positives consistently is heavily dependent on the effect size; total number of predictors, p; and number of non-zero coefficients, k. Simulations showed minimal progress in detection as the sample size and effect size increased

and the total number of predictors and non-zero coefficients decreased. However, in all models, all proposed methods rarely selected true positives regardless of the effect size, number of predictors, or the number of non-zero coefficients, with many models containing only false positives. Therefore, we conclude the lasso, elastic net, and glmLasso$_{dis}$ are not appropriate for detecting associations in the small-sample discrete survival data frequently encountered in RLC studies.

Results also vary with the choice of α and λ. For example, fewer model coefficients will be set to zero as α and λ decrease. From simulations, the elastic net with λ_{minMSE} contained the highest number of true positives and false positives, on average, for approaches (3) and (4). However, there is a tradeoff between choosing λ_{minMSE} and λ_{1SE}. If a higher true positive rate is desired, then λ_{minMSE} is the better option, but if a lower false positive rate is desired, then λ_{1SE} is preferred. On the other hand, glmLasso$_{dis}$ presents a compromise between the lasso and elastic net: a higher number of true positives λ_{1SE} models in (3) and (4) but a much lower number of false positives than both λ_{minMSE} and λ_{1SE} models.

Although the elastic net can improve the detection of true positives and glmLasso$_{dis}$ can reduce the number of false positives in many scenarios, the lasso is the most cited in scientific literature, including RLC studies. However, our simulations show that for small-sample discrete survival data there is a high probability of detecting false positives, and possibly all false positives, using the lasso. Because one cannot be certain whether or not a detected covariate is a true positive or false positive, other models and techniques need to be considered for small-sample discrete survival time data. For example, one might explore the use of forward stepwise regression via bootstrapping to select the most relevant features. However, this technique would apply more to the $k = 1$ setting than to larger k settings. Alternatively, the use of HDLSS

techniques should be explored and implemented in the RLC setting since HDLSS techniques were developed specifically for the setting where $p \gg n$. Regardless, penalized regression techniques are not recommended for making scientific claims but instead should be used to generate hypotheses for future RLC or other small-sample discrete survival studies.

2.6 Acknowledgements

This work was supported by the National Institute of Allergy and Infectious Diseases (NIAID) of the National Institutes of Health (NIH) under award number R37AI054165. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

Table 2.1. Parameter effect coefficients, β, for each number of non-zero coefficients, k, in the true model with a moderate effect size

k	β
1	(log(1.5), 0,…,0)
3	(1, -1, log(1.5), 0,…,0)
5	(0.5, -0.5, 0.5, -0.5, log(1.5), 0,…,0)
7	(0.33, -0.33, 0.33, -0.33, 0.33, -0.33, log(1.5), 0,…,0)

Table 2.2. Results comparing logistic, continuous survival time Cox, and discrete survival time Cox approaches where $n = 10$, $k = 1$, $p = 50$ with moderate effect

Method	Average no. of True Positives	Average no. of False Positives	% Containing True	% Equal to True
Logistic Lasso-minMSE	0.06	1.38	6.40	0.60
Logistic Lasso-1SE	0.02	0.22	2.20	0.40
Logistic Elastic Net-minMSE	0.11	2.52	11.20	0.20
Logistic Elastic-Net-1SE	0.02	0.28	2.20	0.20
Cox Lasso-minMSE	0.09	1.76	8.60	0.60
Cox Lasso-1SE	0.04	0.49	3.80	0.80
Cox Elastic Net-minMSE	0.15	3.41	15.0	0.00
Cox Elastic-Net-1SE	0.05	0.74	5.20	0.40
$\text{glmLasso}_{\text{dis}}$	0.05	0.48	4.80	2.00

Table 2.3. Variables Selected from the MIV02 Study with n = 14, p = 50

Method	Estimate Index
Logistic Lasso-MinMSE	27
Logistic Lasso-1SE	-
Logistic Elastic Net-MinMSE	27
Logistic Elastic Net-1SE	-
Cox Lasso-MinMSE	27, 85, 99, 138
Cox Lasso-1SE	-
Cox Elastic Net-MinMSE	21, 27, 55, 85, 86, 98, 99, 135, 138
Cox Elastic Net-1SE	27, 85, 99, 138
$\text{glmLasso}_{\text{dis}}$	27

Note: '-' : No variables selected

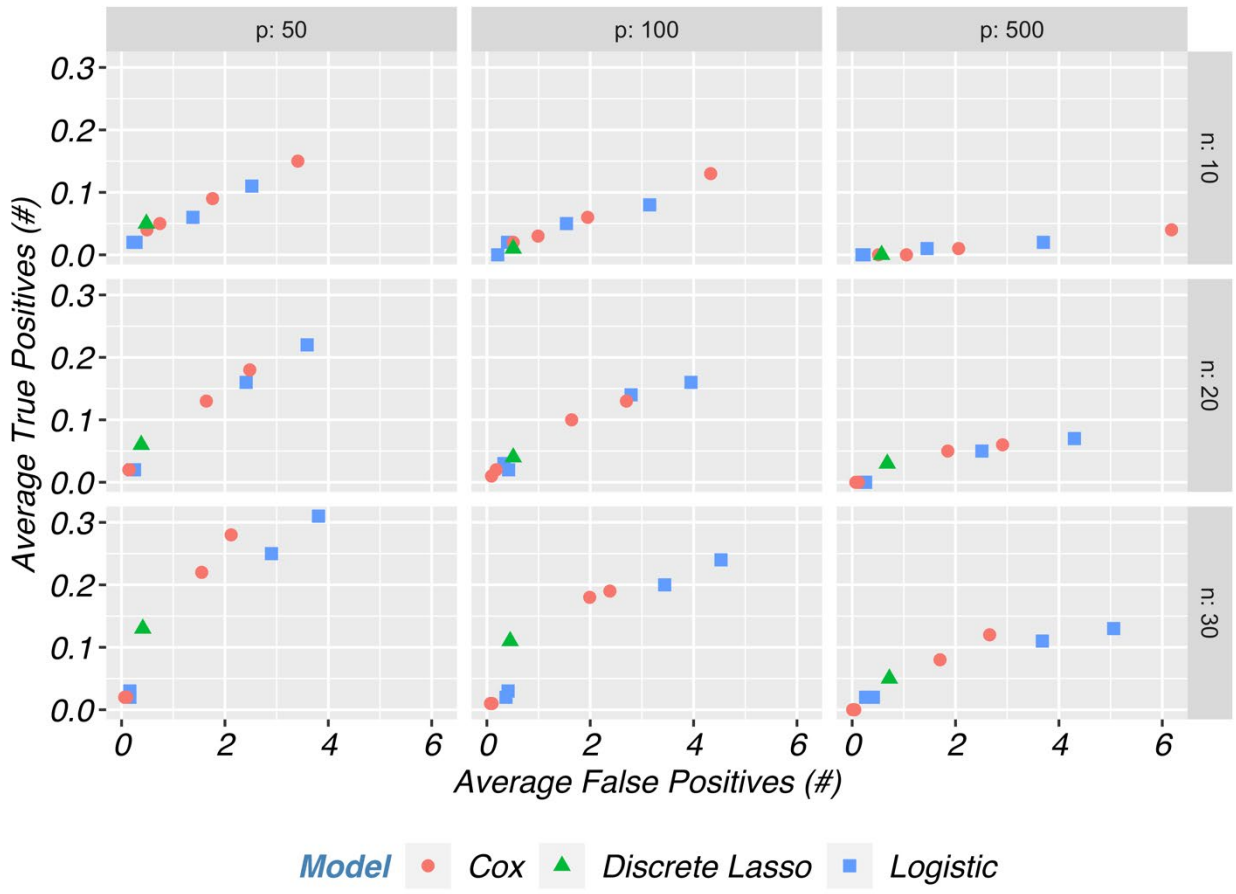


Figure 2.1. The average number of false positives versus the average number of true positives assuming a moderate effect size with $k = 1$ non-zero coefficients.

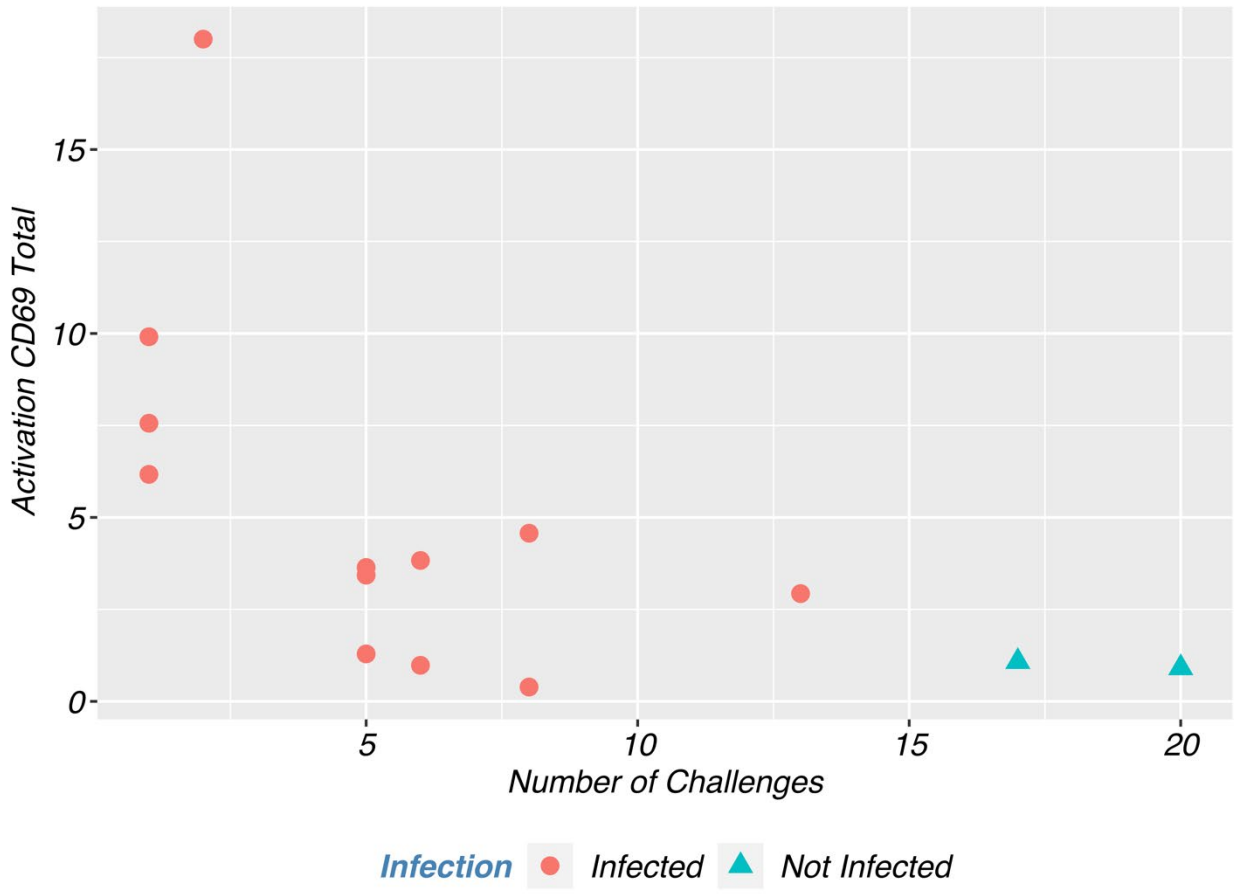


Figure 2.2. Scatterplot of CD69 by the number of challenges.

2.7 Supplemental Information

Supplemental Table 2.1. Parameter effect coefficients, β, for each number, k, of non-zero coefficients in the true model

Effect Size	k	β
Small	1	(log(1.2), 0,…,0)
	3	(1, -1, log(1.2), 0,…,0)
	5	(0.5, -0.5, 0.5, -0.5, log(1.2), 0,…,0)
	7	(0.33, -0.33, 0.33, -0.33, 0.33, -0.33, log(1.2), 0,…,0)
Moderate	1	(log(1.5), 0,…,0)
	3	(1, -1, log(1.5), 0,…,0)
	5	(0.5, -0.5, 0.5, -0.5, log(1.5), 0,…,0)
	7	(0.33, -0.33, 0.33, -0.33, 0.33, -0.33, log(1.5), 0,…,0)
Large	1	(log(2), 0,…,0)
	3	(1, -1, log(2), 0,…,0)
	5	(0.5, -0.5, 0.5, -0.5, log(2), 0,…,0)
	7	(0.33, -0.33, 0.33, -0.33, 0.33, -0.33, log(2), 0,…,0)

Supplemental Table 2.2. Results for k = 1 variables in true model with a small effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.04	1.39	37.4	0.03	1.62	36.6	0.00	1.41	38.6
Logistic Lasso-1SE	10	5	0.01	0.17	6.4	0.00	0.25	10.8	0.00	0.18	8.2
Logistic Elastic Net-minMSE	10	5	0.09	2.70	35.8	0.04	2.90	38.8	0.02	3.74	40.0
Logistic Elastic Net-1SE	10	5	0.02	0.30	5.4	0.00	0.31	7.6	0.00	0.22	4.4
Cox Lasso-minMSE	10	5	0.05	1.69	50.0	0.03	2.00	54.6	0.01	2.06	60.2
Cox Lasso-1SE	10	5	0.02	0.42	19.4	0.01	0.58	25.2	0.00	0.53	24.4
Cox Elastic Net-minMSE	10	5	0.08	3.18	53.6	0.06	4.15	58.2	0.02	5.87	67.2
Cox Elastic Net-1SE	10	5	0.02	0.63	18.6	0.01	0.97	25.4	0.00	0.93	19.6
$glmLasso_{dis}$	10	5	0.02	0.45	28.2	0.01	0.49	29.8	0.00	0.56	32.2
Logistic Lasso-minMSE	20	10	0.05	2.16	38.0	0.04	2.82	43.0	0.01	2.82	45.8
Logistic Lasso-1SE	20	10	0.01	0.23	4.2	0.01	0.27	7.0	0.00	0.27	6.6
Logistic Elastic Net-minMSE	20	10	0.07	2.56	36.4	0.05	3.63	42.2	0.02	4.41	45.0
Logistic Elastic Net-1SE	20	10	0.01	0.24	3.2	0.00	0.34	5.2	0.00	0.23	4.0
Cox Lasso-minMSE	20	10	0.05	1.57	39.6	0.02	1.52	42.0	0.00	1.66	44.2
Cox Lasso-1SE	20	10	0.00	0.12	4.4	0.01	0.15	6.2	0.00	0.14	5.8
Cox Elastic Net-minMSE	20	10	0.07	2.09	40.6	0.03	2.63	47.8	0.02	2.94	48.6

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.01	0.12	2.8	0.01	0.19	3.6	0.00	0.13	3.2
$glmLasso_{dis}$	20	10	0.01	0.37	24.6	0.01	0.53	31.4	0.01	0.53	29.6
Logistic Lasso-minMSE	30	15	0.11	2.84	37.6	0.05	2.93	41.0	0.02	3.68	44.6
Logistic Lasso-1SE	30	15	0.02	0.36	4.4	0.01	0.30	4.0	0.00	0.41	6.6
Logistic Elastic Net-minMSE	30	15	0.12	3.61	38.6	0.06	3.88	41.2	0.03	5.01	44.4
Logistic Elastic Net-1SE	30	15	0.02	0.26	2.8	0.00	0.18	3.0	0.00	0.31	3.6
Cox Lasso-minMSE	30	15	0.07	1.44	34.8	0.03	1.70	41.0	0.01	1.64	41.2
Cox Lasso-1SE	30	15	0.01	0.14	2.2	0.00	0.11	2.2	0.00	0.05	2.8
Cox Elastic Net-minMSE	30	15	0.09	1.94	35.6	0.05	2.43	44.0	0.01	2.55	45.0
Cox Elastic Net-1SE	30	15	0.01	0.05	1.4	0.00	0.08	2.0	0.00	0.02	1.8
$glmLasso_{dis}$	30	15	0.02	0.35	21.2	0.02	0.44	27.4	0.00	0.76	39.2

Supplemental Table 2.3. Results for k = 3 variables in true model with a small effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.30	1.49	20.0	0.18	1.52	27.8	0.07	1.57	33.8
Logistic Lasso-1SE	10	5	0.07	0.25	6.4	0.04	0.27	7.0	0.02	0.31	10.8
Logistic Elastic Net-minMSE	10	5	0.52	3.37	15.0	0.33	3.20	18.6	0.15	4.89	29.8
Logistic Elastic Net-1SE	10	5	0.08	0.37	3.0	0.06	0.38	3.6	0.02	0.47	4.2
Cox Lasso-minMSE	10	5	0.43	1.90	30.2	0.28	1.90	37.6	0.10	2.51	56.4
Cox Lasso-1SE	10	5	0.19	0.61	17.0	0.11	0.54	19.0	0.04	0.67	26.2
Cox Elastic Net-minMSE	10	5	0.73	4.13	21.6	0.54	4.79	29.6	0.24	7.79	57.2
Cox Elastic Net-1SE	10	5	0.27	1.13	13.2	0.15	1.01	15.6	0.08	1.47	22.4
$glmLasso_{dis}$	10	5	0.23	0.74	29.6	0.16	0.86	42.0	0.06	0.96	48.6
Logistic Lasso-minMSE	20	10	1.10	3.54	6.8	0.79	3.42	13.4	0.35	2.89	20.8
Logistic Lasso-1SE	20	10	0.39	0.62	2.0	0.21	0.44	3.4	0.11	0.40	3.8
Logistic Elastic Net-minMSE	20	10	1.33	5.82	3.6	0.93	5.80	9.0	0.44	5.54	17.2
Logistic Elastic Net-1SE	20	10	0.35	0.88	1.4	0.19	0.75	2.2	0.08	0.45	1.0
Cox Lasso-minMSE	20	10	1.02	2.59	8.2	0.78	2.64	14.6	0.36	2.39	26.0
Cox Lasso-1SE	20	10	0.31	0.35	2.6	0.16	0.25	3.6	0.08	0.22	4.8
Cox Elastic Net-minMSE	20	10	1.23	4.21	6.8	0.96	4.71	12.8	0.46	4.60	22.2

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.24	0.36	1.2	0.14	0.28	1.8	0.06	0.26	2.4
$glmLasso_{dis}$	20	10	0.68	0.70	13.8	0.52	0.74	17.4	0.25	1.06	38.4
Logistic Lasso-minMSE	30	15	1.74	5.05	1.0	1.43	5.91	4.0	0.85	5.36	11.2
Logistic Lasso-1SE	30	15	0.73	0.87	0.4	0.58	1.06	1.6	0.24	0.67	3.2
Logistic Elastic Net-minMSE	30	15	1.85	7.59	0.4	1.54	8.66	3.2	0.96	9.31	7.0
Logistic Elastic Net-1SE	30	15	0.61	0.96	0.4	0.45	1.00	1.0	0.19	0.72	1.6
Cox Lasso-minMSE	30	15	1.59	3.48	2.4	1.25	3.88	5.2	0.71	2.92	13.4
Cox Lasso-1SE	30	15	0.49	0.35	0.2	0.36	0.39	1.0	0.11	0.16	1.2
Cox Elastic Net-minMSE	30	15	1.72	5.23	0.8	1.39	5.75	4.2	0.85	5.19	10.8
Cox Elastic Net-1SE	30	15	0.43	0.37	0.4	0.25	0.27	0.6	0.06	0.19	1.0
$glmLasso_{dis}$	30	15	1.16	0.78	6.0	0.98	0.97	9.2	0.61	1.18	22.6

Supplemental Table 2.4. Results for k = 5 variables in true model with a small effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.28	1.48	21.6	0.16	1.28	26.8	0.04	1.27	30.2
Logistic Lasso-1SE	10	5	0.04	0.26	8.6	0.02	0.17	8.8	0.00	0.14	6.8
Logistic Elastic Net-minMSE	10	5	0.55	2.92	13.8	0.30	2.87	23.8	0.06	3.41	30.2
Logistic Elastic Net-1SE	10	5	0.07	0.39	5.0	0.04	0.30	3.8	0.00	0.30	4.8
Cox Lasso-minMSE	10	5	0.40	1.85	32.6	0.24	1.97	44.0	0.06	2.33	58.8
Cox Lasso-1SE	10	5	0.13	0.55	19.2	0.06	0.55	21.8	0.01	0.65	25.2
Cox Elastic Net-minMSE	10	5	0.78	3.85	23.4	0.48	4.28	34.8	0.15	7.11	60.0
Cox Elastic Net-1SE	10	5	0.19	0.89	15.8	0.12	0.98	19.6	0.04	1.42	23.8
$glmLasso_{dis}$	10	5	0.17	0.67	31.6	0.10	0.71	36.6	0.03	0.75	39.0
Logistic Lasso-minMSE	20	10	0.83	2.47	11.2	0.48	2.54	16.4	0.13	2.78	32.8
Logistic Lasso-1SE	20	10	0.15	0.27	2.8	0.09	0.44	4.6	0.01	0.35	8.6
Logistic Elastic Net-minMSE	20	10	1.02	3.60	10.0	0.67	4.27	11.2	0.24	5.08	27.0
Logistic Elastic Net-1SE	20	10	0.17	0.42	1.4	0.08	0.43	2.6	0.03	0.54	5.0
Cox Lasso-minMSE	20	10	0.71	1.72	11.6	0.43	1.84	21.4	0.13	1.98	35.6
Cox Lasso-1SE	20	10	0.10	0.17	2.3	0.07	0.14	3.0	0.01	0.17	5.2
Cox Elastic Net-minMSE	20	10	0.92	2.74	9.8	0.60	3.05	18.2	0.19	3.71	36.6

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.10	0.22	1.2	0.05	0.13	2.2	0.01	0.09	1.8
$glmLasso_{dis}$	20	10	0.33	0.58	19.6	0.23	0.69	23.8	0.06	0.93	42.2
Logistic Lasso-minMSE	30	15	1.28	3.27	7.0	0.89	3.80	11.6	0.36	3.76	22.6
Logistic Lasso-1SE	30	15	0.20	0.39	1.0	0.19	0.56	2.2	0.04	0.62	4.4
Logistic Elastic Net-minMSE	30	15	1.55	4.72	5.6	1.13	5.70	8.8	0.45	6.16	17.0
Logistic Elastic Net-1SE	30	15	0.20	0.48	0.6	0.17	0.53	1.0	0.04	0.57	3.2
Cox Lasso-minMSE	30	15	0.95	1.86	8.0	0.73	2.23	10.6	0.26	2.13	24.4
Cox Lasso-1SE	30	15	0.11	0.10	0.4	0.06	0.08	1.0	0.01	0.10	1.4
Cox Elastic Net-minMSE	30	15	1.18	2.74	5.8	0.89	3.25	9.8	0.35	3.59	23.4
Cox Elastic Net-1SE	30	15	0.11	0.12	0.4	0.04	0.07	0.2	0.01	0.05	0.6
$glmLasso_{dis}$	30	15	0.55	0.60	13.4	0.39	0.75	20.4	0.16	1.05	39.2

Supplemental Table 2.5. Results for k = 7 variables in true model with a small effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.31	1.34	18.8	0.20	1.61	29.4	0.03	1.37	33.2
Logistic Lasso-1SE	10	5	0.04	0.24	6.2	0.03	0.22	6.8	0.00	0.12	5.0
Logistic Elastic Net-minMSE	10	5	0.54	2.50	14.2	0.36	3.18	21.0	0.10	3.93	31.4
Logistic Elastic Net-1SE	10	5	0.04	0.26	3.6	0.02	0.24	4.4	0.00	0.19	3.8
Cox Lasso-minMSE	10	5	0.38	1.65	33.0	0.23	1.93	44.6	0.06	2.08	56.4
Cox Lasso-1SE	10	5	0.12	0.44	14.6	0.07	0.51	20.2	0.02	0.55	23.0
Cox Elastic Net-minMSE	10	5	0.73	3.44	26.2	0.50	4.36	36.4	0.17	6.81	61.0
Cox Elastic Net-1SE	10	5	0.16	0.70	12.0	0.14	0.88	15.8	0.04	1.19	19.6
$glmLasso_{dis}$	10	5	0.14	0.57	28.0	0.07	0.57	34.0	0.03	0.77	39.8
Logistic Lasso-minMSE	20	10	0.72	2.25	11.2	0.47	2.84	20.4	0.10	2.83	36.6
Logistic Lasso-1SE	20	10	0.12	0.41	3.0	0.10	0.53	4.8	0.01	0.40	7.2
Logistic Elastic Net-minMSE	20	10	0.98	3.36	9.4	0.72	4.54	15.0	0.18	5.06	31.8
Logistic Elastic Net-1SE	20	10	0.11	0.30	2.0	0.11	0.65	3.8	0.02	0.36	3.8
Cox Lasso-minMSE	20	10	0.58	1.49	16.2	0.40	1.92	20.4	0.08	1.85	40.0
Cox Lasso-1SE	20	10	0.07	0.16	2.0	0.04	0.18	4.4	0.00	0.14	5.0
Cox Elastic Net-minMSE	20	10	0.80	2.34	15.2	0.55	2.84	18.2	0.16	3.23	36.2

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.05	0.11	2.2	0.03	0.13	1.8	0.01	0.07	2.8
$glmLasso_{dis}$	20	10	0.25	0.52	18.2	0.14	0.53	25.8	0.06	0.86	39.4
Logistic Lasso-minMSE	30	15	1.20	3.26	10.2	0.71	3.46	14.0	0.26	3.69	30.2
Logistic Lasso-1SE	30	15	0.19	0.50	1.0	0.12	0.45	3.8	0.04	0.35	4.8
Logistic Elastic Net-minMSE	30	15	1.47	4.42	7.4	0.92	4.72	10.8	0.35	5.98	26.4
Logistic Elastic Net-1SE	30	15	0.15	0.42	0.8	0.13	0.49	2.6	0.04	0.61	3.2
Cox Lasso-minMSE	30	15	0.84	1.87	13.2	0.53	1.86	15.8	0.17	1.98	32.0
Cox Lasso-1SE	30	15	0.07	0.09	1.0	0.05	0.13	1.2	0.00	0.06	1.4
Cox Elastic Net-minMSE	30	15	1.11	2.78	10.0	0.69	2.74	13.4	0.23	3.03	32.4
Cox Elastic Net-1SE	30	15	0.07	0.13	0.8	0.04	0.10	0.6	0.00	0.01	0.4
$glmLasso_{dis}$	30	15	0.32	0.47	16.6	0.26	0.66	26.0	0.11	0.89	38.2

Supplemental Table 2.6. Results for k = 1 variables in true model with a moderate effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.06	1.38	35.2	0.05	1.54	35.0	0.01	1.45	37.2
Logistic Lasso-1SE	10	5	0.02	0.22	7.2	0.00	0.21	8.2	0.00	0.19	8.6
Logistic Elastic Net-minMSE	10	5	0.11	2.52	33.4	0.08	3.15	35.4	0.02	3.70	38.6
Logistic Elastic Net-1SE	10	5	0.02	0.28	5.6	0.02	0.40	6.2	0.00	0.23	4.0
Cox Lasso-minMSE	10	5	0.09	1.76	45.8	0.06	1.95	52.8	0.01	2.06	59.6
Cox Lasso-1SE	10	5	0.04	0.49	18.6	0.02	0.51	23.2	0.00	0.51	24.2
Cox Elastic Net-minMSE	10	5	0.15	3.41	47.0	0.13	4.33	53.6	0.04	6.18	67.6
Cox Elastic Net-1SE	10	5	0.05	0.74	17.6	0.03	0.99	24.4	0.00	1.05	21.0
$glmLasso_{dis}$	10	5	0.05	0.48	27.0	0.01	0.51	31.4	0.00	0.57	32.8
Logistic Lasso-minMSE	20	10	0.16	2.41	33.8	0.14	2.79	35.8	0.05	2.51	36.8
Logistic Lasso-1SE	20	10	0.02	0.21	3.0	0.02	0.42	6.0	0.00	0.26	5.8
Logistic Elastic Net-minMSE	20	10	0.22	3.59	30.0	0.16	3.95	33.8	0.07	4.30	36.4
Logistic Elastic Net-1SE	20	10	0.02	0.25	3.0	0.03	0.33	3.0	0.00	0.23	2.8
Cox Lasso-minMSE	20	10	0.13	1.64	35.0	0.10	1.64	36.6	0.05	1.85	41.4
Cox Lasso-1SE	20	10	0.02	0.14	4.8	0.02	0.18	5.6	0.00	0.12	4.8
Cox Elastic Net-minMSE	20	10	0.18	2.48	36.6	0.13	2.70	39.8	0.06	2.91	42.4

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.02	0.15	3.8	0.01	0.09	2.4	0.00	0.07	1.6
$glmLasso_{dis}$	20	10	0.06	0.38	21.6	0.04	0.51	28.2	0.03	0.68	36.0
Logistic Lasso-minMSE	30	15	0.25	2.90	25.8	0.20	3.44	33.0	0.11	3.68	36.8
Logistic Lasso-1SE	30	15	0.03	0.16	2.0	0.03	0.41	4.2	0.02	0.41	4.4
Logistic Elastic Net-minMSE	30	15	0.31	3.81	23.4	0.24	4.53	31.2	0.13	5.06	35.0
Logistic Elastic Net-1SE	30	15	0.02	0.16	1.4	0.02	0.37	2.6	0.02	0.26	2.4
Cox Lasso-minMSE	30	15	0.22	1.55	24.2	0.18	1.99	35.4	0.08	1.70	34.0
Cox Lasso-1SE	30	15	0.02	0.06	2.0	0.01	0.07	2.0	0.00	0.05	2.2
Cox Elastic Net-minMSE	30	15	0.28	2.12	23.0	0.19	2.38	32.4	0.12	2.66	35.0
Cox Elastic Net-1SE	30	15	0.02	0.10	1.0	0.01	0.10	1.6	0.00	0.01	0.6
$glmLasso_{dis}$	30	15	0.13	0.41	21.4	0.11	0.45	22.2	0.05	0.72	36.2

Supplemental Table 2.7. Results for k = 3 variables in true model with a moderate effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.31	1.43	21.6	0.23	1.67	26.6	0.06	1.53	34.0
Logistic Lasso-1SE	10	5	0.06	0.24	6.8	0.05	0.29	8.0	0.02	0.27	9.4
Logistic Elastic Net-minMSE	10	5	0.54	3.27	14.4	0.39	3.56	17.6	0.15	3.93	26.8
Logistic Elastic Net-1SE	10	5	0.09	0.47	4.8	0.06	0.45	5.0	0.02	0.58	4.8
Cox Lasso-minMSE	10	5	0.43	1.85	30.6	0.30	1.93	37.6	0.10	2.16	51.4
Cox Lasso-1SE	10	5	0.16	0.52	17.0	0.12	0.60	19.6	0.03	0.56	24.2
Cox Elastic Net-minMSE	10	5	0.76	4.10	18.6	0.56	4.91	29.8	0.25	6.88	51.4
Cox Elastic Net-1SE	10	5	0.25	1.01	12.4	0.17	1.06	16.8	0.05	1.17	20.8
$glmLasso_{dis}$	10	5	0.25	0.72	31.4	0.14	0.90	43.2	0.07	0.95	46.2
Logistic Lasso-minMSE	20	10	1.10	3.29	5.6	0.82	3.27	11.6	0.34	3.17	25.8
Logistic Lasso-1SE	20	10	0.30	0.53	2.0	0.24	0.49	4.4	0.08	0.42	5.4
Logistic Elastic Net-minMSE	20	10	1.27	5.16	4.2	0.97	5.55	8.0	0.45	5.85	20.0
Logistic Elastic Net-1SE	20	10	0.31	0.74	1.4	0.21	0.64	1.4	0.06	0.51	2.2
Cox Lasso-minMSE	20	10	0.99	2.40	7.4	0.76	2.64	15.0	0.32	2.35	27.8
Cox Lasso-1SE	20	10	0.28	0.35	2.0	0.20	0.30	3.2	0.05	0.24	4.0
Cox Elastic Net-minMSE	20	10	1.20	4.06	4.2	0.96	4.64	13.4	0.44	4.44	26.2

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.29	0.45	0.6	0.16	0.40	2.6	0.05	0.18	1.6
$glmLasso_{dis}$	20	10	0.69	0.71	12.4	0.53	0.85	21.0	0.23	1.13	40.4
Logistic Lasso-minMSE	30	15	1.79	4.79	0.8	1.42	5.05	4.0	0.88	5.44	12.6
Logistic Lasso-1SE	30	15	0.67	0.73	1.0	0.47	0.78	1.8	0.25	0.78	2.2
Logistic Elastic Net-minMSE	30	15	1.88	7.03	0.2	1.54	8.07	2.8	1.03	9.35	8.0
Logistic Elastic Net-1SE	30	15	0.60	0.96	0.4	0.38	0.79	1.0	0.20	0.88	1.0
Cox Lasso-minMSE	30	15	1.69	3.47	1.2	1.32	3.41	4.8	0.72	3.09	14.0
Cox Lasso-1SE	30	15	0.46	0.40	0.4	0.30	0.23	1.0	0.11	0.20	2.2
Cox Elastic Net-minMSE	30	15	1.76	4.89	0.8	1.46	5.35	3.2	0.84	5.37	12.0
Cox Elastic Net-1SE	30	15	0.42	0.44	0.8	0.25	0.31	0.2	0.08	0.16	0.6
$glmLasso_{dis}$	30	15	1.14	0.73	7.0	0.97	1.00	9.8	0.57	1.19	27.0

Supplemental Table 2.8. Results for k = 5 variables in true model with a moderate effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.28	1.31	20.0	0.12	1.44	29.6	0.04	1.40	33.8
Logistic Lasso-1SE	10	5	0.05	0.24	7.8	0.01	0.19	8.2	0.01	0.18	7.4
Logistic Elastic Net-minMSE	10	5	0.47	2.47	15.6	0.28	3.10	22.4	0.10	3.73	30.2
Logistic Elastic Net-1SE	10	5	0.07	0.33	3.4	0.03	0.38	5.4	0.01	0.24	3.4
Cox Lasso-minMSE	10	5	0.36	1.68	33.6	0.25	1.99	43.4	0.07	2.40	60.6
Cox Lasso-1SE	10	5	0.12	0.49	18.0	0.08	0.54	21.4	0.02	0.62	25.2
Cox Elastic Net-minMSE	10	5	0.72	3.65	24.8	0.51	4.54	35.6	0.19	7.04	58.8
Cox Elastic Net-1SE	10	5	0.20	0.84	16.0	0.13	1.06	18.2	0.05	1.34	22.0
$glmLasso_{dis}$	10	5	0.18	0.62	29.2	0.10	0.74	37.4	0.02	0.84	45.0
Logistic Lasso-minMSE	20	10	0.81	2.47	13.2	0.53	2.65	16.4	0.13	2.66	33.0
Logistic Lasso-1SE	20	10	0.15	0.36	3.4	0.10	0.35	4.2	0.02	0.46	8.0
Logistic Elastic Net-minMSE	20	10	1.08	3.95	9.6	0.69	4.06	11.4	0.25	5.24	28.2
Logistic Elastic Net-1SE	20	10	0.17	0.50	2.2	0.08	0.37	2.6	0.02	0.48	5.2
Cox Lasso-minMSE	20	10	0.78	2.01	12.4	0.48	1.98	19.6	0.12	2.13	36.8
Cox Lasso-1SE	20	10	0.11	0.16	1.8	0.07	0.13	2.4	0.02	0.13	4.0
Cox Elastic Net-minMSE	20	10	1.03	3.09	11.0	0.62	3.15	19.0	0.20	3.91	38.8

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.11	0.22	1.2	0.04	0.12	1.8	0.02	0.10	1.8
$glmLasso_{dis}$	20	10	0.32	0.45	16.4	0.25	0.71	25.0	0.06	0.96	40.8
Logistic Lasso-minMSE	30	15	1.43	3.85	7.0	0.92	3.67	11.2	0.39	4.02	24.8
Logistic Lasso-1SE	30	15	0.29	0.45	1.6	0.18	0.50	1.2	0.06	0.48	4.2
Logistic Elastic Net-minMSE	30	15	1.66	5.42	5.2	1.11	5.32	9.0	0.53	6.67	20.2
Logistic Elastic Net-1SE	30	15	0.28	0.48	1.2	0.15	0.45	0.8	0.04	0.44	2.8
Cox Lasso-minMSE	30	15	1.13	2.18	8.8	0.74	2.27	14.2	0.27	2.26	26.6
Cox Lasso-1SE	30	15	0.15	0.10	0.6	0.05	0.07	0.6	0.02	0.11	1.6
Cox Elastic Net-minMSE	30	15	1.39	3.36	6.4	0.92	3.16	11.0	0.43	3.88	22.0
Cox Elastic Net-1SE	30	15	0.14	0.15	0.6	0.05	0.11	0.6	0.01	0.04	1.0
$glmLasso_{dis}$	30	15	0.53	0.57	13.4	0.43	0.79	23.6	0.18	1.20	40.4

Supplemental Table 2.9. Results for k = 7 variables in true model with a moderate effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.32	1.36	19.4	0.19	1.53	29.0	0.06	1.55	34.6
Logistic Lasso-1SE	10	5	0.06	0.32	7.4	0.03	0.25	8.0	0.01	0.18	7.0
Logistic Elastic Net-minMSE	10	5	0.60	2.68	14.2	0.37	3.19	21.8	0.13	4.33	31.4
Logistic Elastic Net-1SE	10	5	0.08	0.36	3.8	0.03	0.26	5.0	0.01	0.33	4.6
Cox Lasso-minMSE	10	5	0.39	1.72	33.0	0.23	1.84	39.6	0.08	2.32	56.0
Cox Lasso-1SE	10	5	0.13	0.48	14.8	0.06	0.46	20.4	0.03	0.61	25.4
Cox Elastic Net-minMSE	10	5	0.80	3.61	25.0	0.50	4.35	34.0	0.20	7.35	58.4
Cox Elastic Net-1SE	10	5	0.19	0.79	12.2	0.14	0.93	15.8	0.06	1.55	24.8
$glmLasso_{dis}$	10	5	0.18	0.55	27.8	0.11	0.61	31.6	0.02	0.77	39.2
Logistic Lasso-minMSE	20	10	0.71	2.07	11.8	0.49	2.75	19.6	0.12	2.57	32.4
Logistic Lasso-1SE	20	10	0.10	0.24	2.4	0.08	0.40	6.6	0.01	0.34	7.8
Logistic Elastic Net-minMSE	20	10	1.10	3.42	7.0	0.71	4.21	16.6	0.19	4.85	30.2
Logistic Elastic Net-1SE	20	10	0.11	0.29	1.2	0.08	0.48	3.4	0.02	0.32	3.4
Cox Lasso-minMSE	20	10	0.68	1.64	13.0	0.42	1.81	23.2	0.10	1.88	38.2
Cox Lasso-1SE	20	10	0.09	0.17	1.8	0.04	0.10	3.4	0.01	0.09	3.8
Cox Elastic Net-minMSE	20	10	0.89	2.46	10.2	0.60	2.91	20.0	0.19	3.39	35.8

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.10	0.21	1.6	0.04	0.19	2.2	0.01	0.11	2.0
$glmLasso_{dis}$	20	10	0.31	0.53	15.2	0.16	0.63	28.6	0.06	0.85	41.4
Logistic Lasso-minMSE	30	15	1.28	3.20	7.8	0.82	3.56	13.0	0.35	4.40	25.5
Logistic Lasso-1SE	30	15	0.21	0.43	1.2	0.13	0.44	2.6	0.04	0.43	3.8
Logistic Elastic Net-minMSE	30	15	1.65	4.62	5.8	1.06	4.73	9.0	0.44	6.46	21.2
Logistic Elastic Net-1SE	30	15	0.23	0.51	1.2	0.11	0.50	1.4	0.04	0.68	2.6
Cox Lasso-minMSE	30	15	0.87	1.74	11.2	0.62	1.91	14.0	0.18	1.82	27.2
Cox Lasso-1SE	30	15	0.09	0.11	1.2	0.03	0.12	1.2	0.00	0.05	2.4
Cox Elastic Net-minMSE	30	15	1.16	2.60	10.4	0.75	2.74	13.2	0.29	3.11	23.0
Cox Elastic Net-1SE	30	15	0.09	0.12	0.6	0.04	0.15	1.0	0.01	0.05	0.8
$glmLasso_{dis}$	30	15	0.45	0.47	13.8	0.32	0.66	23.6	0.12	1.05	44.4

Supplemental Table 2.10. Results for k = 1 variable in true model with a large effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.15	1.59	33.4	0.10	1.67	35.6	0.01	1.35	34.4
Logistic Lasso-1SE	10	5	0.05	0.24	7.2	0.02	0.34	12.4	0.00	0.13	4.8
Logistic Elastic Net-minMSE	10	5	0.23	2.87	25.4	0.16	3.39	33.8	0.05	4.19	35.2
Logistic Elastic Net-1SE	10	5	0.04	0.21	3.8	0.03	0.47	7.2	0.00	0.24	2.8
Cox Lasso-minMSE	10	5	0.18	1.84	40.2	0.13	2.08	53.0	0.04	2.21	57.4
Cox Lasso-1SE	10	5	0.08	0.49	18.6	0.07	0.48	19.2	0.01	0.61	26.8
Cox Elastic Net-minMSE	10	5	0.27	3.64	39.0	0.22	4.41	51.0	0.08	6.41	64.2
Cox Elastic Net-1SE	10	5	0.11	0.90	16.2	0.09	0.81	17.2	0.02	1.22	23.2
$glmLasso_{dis}$	10	5	0.10	0.60	33.0	0.05	0.68	37.8	0.02	0.70	41.2
Logistic Lasso-minMSE	20	10	0.37	2.66	21.6	0.27	2.71	25.4	0.17	3.11	32.2
Logistic Lasso-1SE	20	10	0.05	0.30	3.6	0.06	0.40	5.0	0.03	0.29	6.2
Logistic Elastic Net-minMSE	20	10	0.43	3.80	16.4	0.33	4.14	21.4	0.21	5.32	28.8
Logistic Elastic Net-1SE	20	10	0.05	0.24	2.4	0.04	0.31	2.6	0.02	0.34	3.2
Cox Lasso-minMSE	20	10	0.35	1.72	19.0	0.26	1.78	27.4	0.14	1.95	37.8
Cox Lasso-1SE	20	10	0.05	0.12	2.0	0.05	0.14	4.0	0.02	0.13	4.0
Cox Elastic Net-minMSE	20	10	0.39	2.64	21.8	0.31	2.80	25.0	0.20	3.71	35.2

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.04	0.17	2.4	0.03	0.13	1.4	0.01	0.13	2.8
$glmLasso_{dis}$	20	10	0.26	0.50	16.6	0.19	0.60	23.8	0.10	0.83	37.0
Logistic Lasso-minMSE	30	15	0.62	3.64	11	0.54	3.86	13.4	0.35	5.16	29
Logistic Lasso-1SE	30	15	0.13	0.31	2.2	0.10	0.37	2.6	0.07	0.59	5.8
Logistic Elastic Net-minMSE	30	15	0.66	4.79	9.4	0.58	5.49	12.2	0.39	8.13	25.2
Logistic Elastic Net-1SE	30	15	0.11	0.30	1.0	0.09	0.56	1.6	0.05	055	4.6
Cox Lasso-minMSE	30	15	0.57	2.40	11.8	0.50	2.33	13.8	0.29	2.29	26.0
Cox Lasso-1SE	30	15	0.08	0.12	1.6	0.06	0.06	0.4	0.03	0.12	2.6
Cox Elastic Net-minMSE	30	15	0.60	3.43	10.8	0.52	3.46	12.4	0.34	3.97	26.0
Cox Elastic Net-1SE	30	15	0.09	0.12	0.8	0.04	0.08	1.0	0.02	0.09	0.8
$glmLasso_{dis}$	30	15	0.46	0.50	13.8	0.37	0.67	19.0	0.26	0.86	29.0

Supplemental Table 2.11. Results for k = 3 variables in true model with a large effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.30	1.22	21.0	0.22	1.62	24.6	0.06	1.55	35.2
Logistic Lasso-1SE	10	5	0.05	0.23	7.4	0.05	0.31	8.6	0.02	0.27	9.6
Logistic Elastic Net-minMSE	10	5	0.57	3.44	13.6	0.39	3.50	14.2	0.15	4.22	26.4
Logistic Elastic Net-1SE	10	5	0.07	0.42	4.2	0.05	0.47	4.2	0.03	0.63	5.2
Cox Lasso-minMSE	10	5	0.44	1.99	29.6	0.34	2.10	37.0	0.08	2.30	58.0
Cox Lasso-1SE	10	5	0.17	0.59	18.6	0.12	0.59	19.8	0.02	0.58	26.2
Cox Elastic Net-minMSE	10	5	0.76	4.23	22.2	0.61	4.93	29.6	0.25	7.13	54.0
Cox Elastic Net-1SE	10	5	0.27	1.05	13.2	0.20	1.08	15.4	0.05	1.23	21.0
$glmLasso_{dis}$	10	5	0.30	0.76	29.8	0.18	0.98	41.2	0.06	1.07	51.4
Logistic Lasso-minMSE	20	10	1.13	3.26	7.8	0.85	3.37	14.4	0.31	2.88	22.0
Logistic Lasso-1SE	20	10	0.34	0.47	1.8	0.25	0.48	4.4	0.07	0.46	7.0
Logistic Elastic Net-minMSE	20	10	1.38	5.29	5.8	1.10	6.08	9.2	0.41	5.58	18.0
Logistic Elastic Net-1SE	20	10	0.31	0.61	1.8	0.20	0.56	1.6	0.06	0.62	4.2
Cox Lasso-minMSE	20	10	1.08	2.53	7.4	0.87	2.77	14.0	0.31	2.61	30.2
Cox Lasso-1SE	20	10	0.30	0.34	2.4	0.18	0.33	3.2	0.06	0.18	4.8
Cox Elastic Net-minMSE	20	10	1.26	4.09	5.4	1.05	4.58	11.2	0.44	4.73	23.0

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.28	0.41	1.8	0.18	0.36	3.0	0.05	0.24	2.6
$glmLasso_{dis}$	20	10	0.70	0.72	13.0	0.54	0.88	20.4	0.21	1.15	42.0
Logistic Lasso-minMSE	30	15	1.97	4.92	1.2	1.61	5.49	3.4	0.85	5.10	13.2
Logistic Lasso-1SE	30	15	0.70	0.69	1.0	0.54	0.98	1.6	0.21	0.98	4.2
Logistic Elastic Net-minMSE	30	15	2.08	7.15	0.6	1.74	8.70	2.0	1.03	8.97	8.4
Logistic Elastic Net-1SE	30	15	0.62	1.07	0.6	0.47	0.93	1.2	0.13	0.66	1.6
Cox Lasso-minMSE	30	15	1.79	3.34	1.8	1.50	3.89	3.6	0.77	3.25	10.6
Cox Lasso-1SE	30	15	0.46	0.29	0.4	0.33	0.29	0.8	0.11	0.16	1.2
Cox Elastic Net-minMSE	30	15	1.99	5.17	0.8	1.61	5.59	2.2	0.90	5.18	8.6
Cox Elastic Net-1SE	30	15	0.37	0.36	0.2	0.33	0.44	0.8	0.07	0.12	1.6
$glmLasso_{dis}$	30	15	1.20	0.86	12.8	1.00	1.02	13.6	0.57	1.38	28.4

Supplemental Table 2.12. Results for k = 5 variables in true model with a large effect

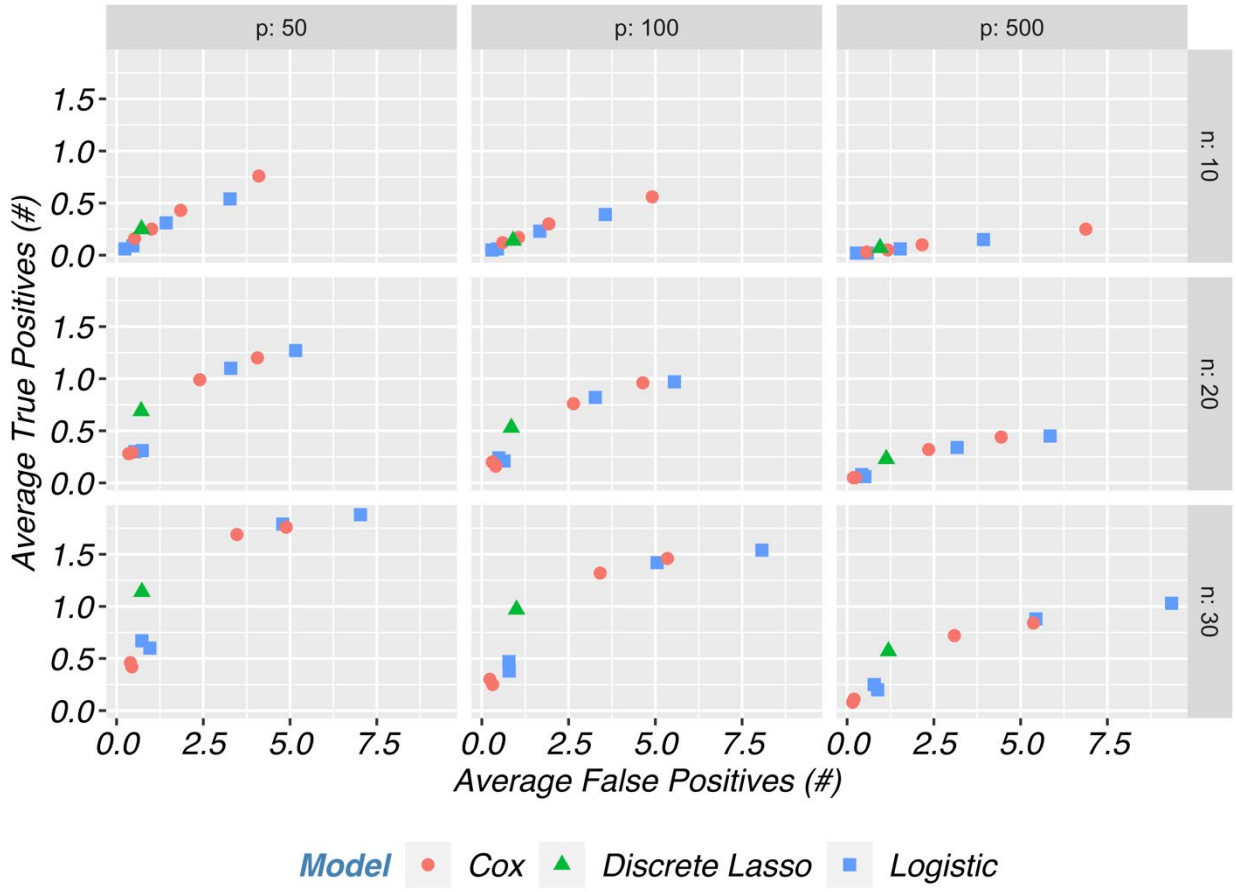
Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.32	1.39	20.8	0.16	1.31	24.0	0.06	1.62	35.4
Logistic Lasso-1SE	10	5	0.07	0.22	7.4	0.03	0.19	7.8	0.01	0.21	7.8
Logistic Elastic Net-minMSE	10	5	0.54	2.82	14.8	0.32	2.87	17.6	0.13	4.61	31.2
Logistic Elastic Net-1SE	10	5	0.11	0.46	4.0	0.04	0.29	2.8	0.01	0.32	4.4
Cox Lasso-minMSE	10	5	0.42	1.80	30.0	0.31	1.99	38.8	0.08	2.39	58.4
Cox Lasso-1SE	10	5	0.12	0.45	15.2	0.09	0.55	19.8	0.02	0.55	24.0
Cox Elastic Net-minMSE	10	5	0.82	3.67	21.0	0.62	4.69	30.6	0.23	7.60	56.0
Cox Elastic Net-1SE	10	5	0.22	0.89	13.2	0.16	1.00	15.8	0.04	1.33	24.0
$glmLasso_{dis}$	10	5	0.20	0.67	31.8	0.13	0.72	36.0	0.04	0.87	43.6
Logistic Lasso-minMSE	20	10	0.87	2.56	10.6	0.60	2.99	16.8	0.20	3.13	35.6
Logistic Lasso-1SE	20	10	0.19	0.40	3.6	0.12	0.45	5.2	0.03	0.33	7.4
Logistic Elastic Net-minMSE	20	10	1.22	4.14	7.0	0.84	4.89	12.2	0.30	5.80	29.2
Logistic Elastic Net-1SE	20	10	0.17	0.44	1.4	0.13	0.52	2.0	0.03	0.49	4.6
Cox Lasso-minMSE	20	10	0.88	1.91	11.0	0.57	2.16	16.4	0.19	2.09	35.4
Cox Lasso-1SE	20	10	0.15	0.22	2.0	0.10	0.23	3.4	0.04	0.15	4.0
Cox Elastic Net-minMSE	20	10	1.14	3.05	8.4	0.76	3.64	15.4	0.29	4.18	36.0

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.15	0.26	1.8	0.08	0.24	2.6	0.01	0.15	3.2
$glmLasso_{dis}$	20	10	0.42	0.54	16.8	0.32	0.79	27.0	0.09	1.02	42.6
Logistic Lasso-minMSE	30	15	1.79	4.15	4.8	1.18	4.26	6.0	0.49	4.64	23.0
Logistic Lasso-1SE	30	15	0.30	0.51	1.8	0.24	0.65	1.6	0.10	0.73	5.2
Logistic Elastic Net-minMSE	30	15	2.11	6.10	3.4	1.44	6.78	4.4	0.64	7.67	16.8
Logistic Elastic Net-1SE	30	15	0.27	0.55	1.0	0.22	0.64	1.0	0.09	0.99	3.2
Cox Lasso-minMSE	30	15	1.40	2.51	7.0	0.98	2.63	10.4	0.34	2.46	22.8
Cox Lasso-1SE	30	15	0.16	0.14	0.6	0.10	0.18	1.6	0.04	0.13	1.2
Cox Elastic Net-minMSE	30	15	1.69	3.76	5.4	1.21	4.05	8.6	0.48	4.24	19.0
Cox Elastic Net-1SE	30	15	0.17	0.19	0.4	0.08	0.14	0.4	0.02	0.07	0.6
$glmLasso_{dis}$	30	15	0.70	0.60	10.8	0.55	0.78	18.8	0.27	1.20	38.8

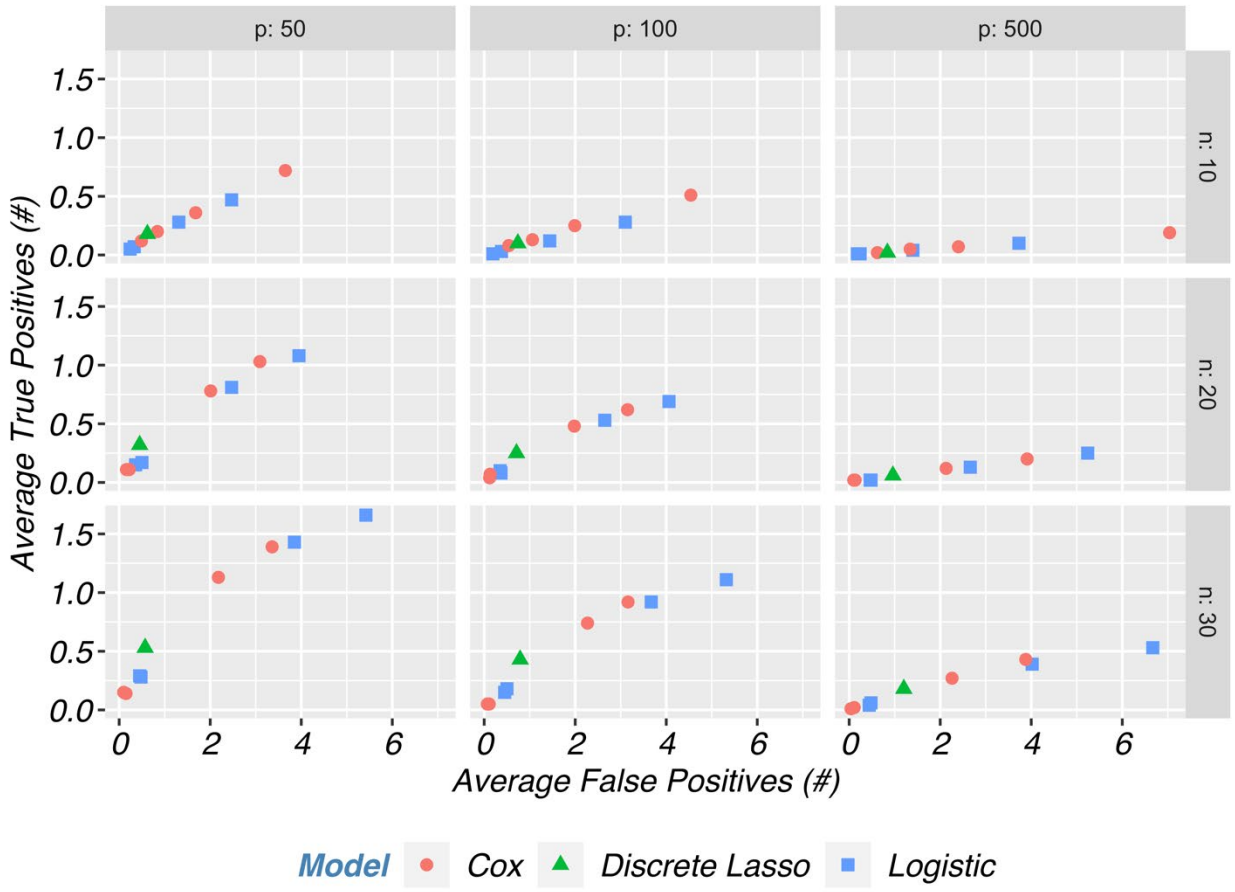
Supplemental Table 2.13. Results for k = 7 variables in true model with a large effect

Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Logistic Lasso-minMSE	10	5	0.36	1.28	19.0	0.18	1.34	24.0	0.05	1.50	32.2
Logistic Lasso-1SE	10	5	0.06	0.20	6.2	0.03	0.19	6.6	0.01	0.15	6.8
Logistic Elastic Net-minMSE	10	5	0.66	2.81	12.2	0.37	2.86	19.0	0.14	4.25	27.8
Logistic Elastic Net-1SE	10	5	0.07	0.32	3.0	0.03	0.26	3.8	0.01	0.28	3.2
Cox Lasso-minMSE	10	5	0.44	1.79	30.8	0.27	1.78	41.2	0.07	2.33	58.4
Cox Lasso-1SE	10	5	0.14	0.44	16.8	0.07	0.42	17.0	0.03	0.64	26.0
Cox Elastic Net-minMSE	10	5	0.89	3.70	20.2	0.58	4.36	33.8	0.24	7.29	54.6
Cox Elastic Net-1SE	10	5	0.27	0.93	14.0	0.15	0.81	16.2	0.06	1.50	24.0
$glmLasso_{dis}$	10	5	0.24	0.62	28.0	0.14	0.74	33.4	0.02	0.93	47.4
Logistic Lasso-minMSE	20	10	0.95	2.57	8.8	0.58	2.51	16.0	0.18	2.67	28.2
Logistic Lasso-1SE	20	10	0.21	0.43	1.4	0.14	0.43	4.4	0.02	0.23	5.4
Logistic Elastic Net-minMSE	20	10	1.34	4.12	6.2	0.83	4.17	12.2	0.27	5.12	25.8
Logistic Elastic Net-1SE	20	10	0.21	0.48	0.6	0.12	0.44	2.2	0.02	0.34	3.6
Cox Lasso-minMSE	20	10	0.79	1.77	11.8	0.50	2.08	20.2	0.15	1.90	34.6
Cox Lasso-1SE	20	10	0.11	0.17	1.6	0.11	0.23	2.6	0.03	0.11	3.4
Cox Elastic Net-minMSE	20	10	1.10	2.91	8.8	0.77	3.41	15.2	0.22	3.53	33.8

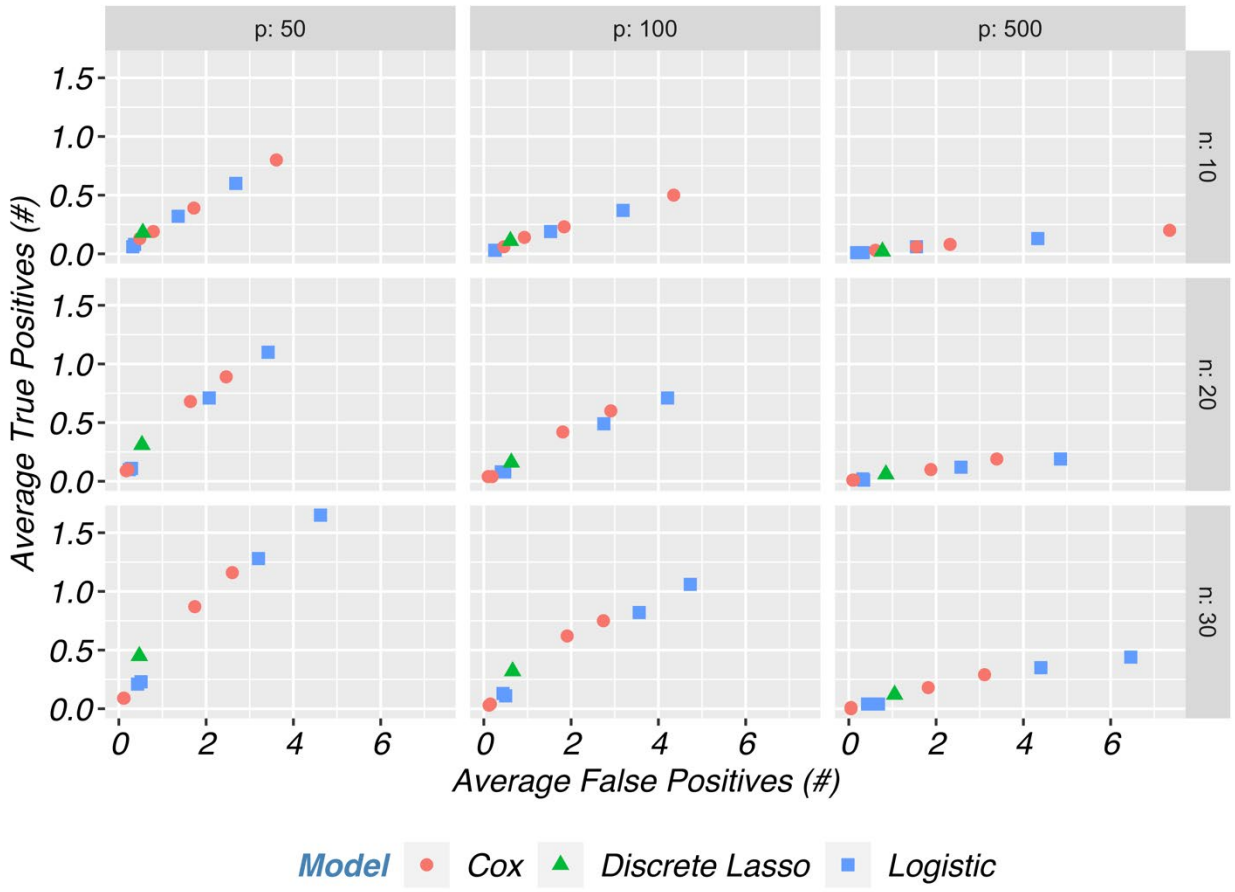
Method	N	N_{Group}	p = 50			p = 100			p = 500		
			Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables	Avg. no. true positives	Avg. no. false positives	Models w. only false variables
Cox Elastic Net-1SE	20	10	0.14	0.22	0.8	0.08	0.18	1.2	0.01	0.06	1.0
$glmLasso_{dis}$	20	10	0.42	0.58	17.6	0.26	0.72	26.4	0.11	0.93	39.6
Logistic Lasso-minMSE	30	15	1.55	3.53	6.2	1.06	4.00	10.2	0.41	4.00	24.0
Logistic Lasso-1SE	30	15	0.25	0.41	1.8	0.19	0.53	1.6	0.08	0.60	5.2
Logistic Elastic Net-minMSE	30	15	1.88	5.03	4.4	1.30	5.62	7.2	0.53	6.62	19.8
Logistic Elastic Net-1SE	30	15	0.27	0.50	0.6	0.19	0.69	2.0	0.05	0.42	1.6
Cox Lasso-minMSE	30	15	1.09	1.95	6.0	0.74	2.16	12.0	0.33	2.27	23.2
Cox Lasso-1SE	30	15	0.13	0.13	1.0	0.05	0.09	1.4	0.04	0.09	2.0
Cox Elastic Net-minMSE	30	15	1.41	2.93	6.2	0.95	3.23	10.6	0.42	3.58	18.6
Cox Elastic Net-1SE	30	15	0.12	0.13	0.6	0.04	0.09	0.8	0.03	0.10	1.0
$glmLasso_{dis}$	30	15	0.51	0.51	13.8	0.48	0.70	20.8	0.24	1.17	40.4



Supplemental Figure 2.1. The average number of false positives versus the average number of true positives assuming a moderate effect size with $k = 3$ non-zero coefficients.



Supplemental Figure 2.2. The average number of false positives versus the average number of true positives assuming a moderate effect size with $k = 5$ non-zero coefficients.



Supplemental Figure 2.3. The average number of false positives versus the average number of true positives assuming a moderate effect size with $k = 7$ non-zero coefficients.

CHAPTER 3: A MACHINE LEARNING APPROACH TO REPEATED LOW-DOSE CHALLENGE EXPERIMENTS

3.1 Introduction

Even though we live in an era of big data for biomedical research, there are many applications where small samples in pre-clinical and human assessment are unavoidable, such as in the development of new drugs and vaccines (Aban & George, 2015). One such application is repeated low-dose challenge studies. In repeated low-dose challenge experiments, animals are assigned to an active or placebo candidate vaccine and then are repeatedly challenged (exposed) with some target pathogen, either until infection or some maximum number of challenges is reached (Nolen et al., 2015). Many times, the number of non-human primates in an RLC study is small (e.g. ≤ 20) and number of features is large (e.g. ≥ 100) due to the high cost of one non-human primate and the high number of antibody and functional measure features of interest (Chaudhury et al., 2018; Choi et al., 2015). Because of the small sample size, the analysis of these types of RLC data requires new statistical methods or adaptations of existing methods. Many traditional methods for low dimensional data are not appropriate for RLC data because there is an insufficient number of samples to adequately estimate the underlying covariance. In addition, dimension reduction techniques such as principal component analysis and linear discriminant analysis are not appropriate for RLC studies because of the difficulty of interpretation, evidence of data piling in HDLSS settings, and small sample size of the training data sets (Aoshima et al., 2018).

Here, we explore the use of the direction-projection-permutation (DiProPerm) test on RLC data. The DiProPerm test is able to assess if there is a difference between two high-dimensional distributions. We adapt the DiProPerm to RLC studies to test whether animals are more likely to become infected early (i.e., before the median infection time) as opposed to late, given a set of antibody and functional measurements. If the DiProPerm test detects a difference, then the DWD loadings can be evaluated to assess which variables have the most influence on prolonging infection time. We also assess the type I error and power of the DiProPerm test on RLC experiments via a simulation experiment. Data from a repeated low-dose challenge study is used as an example of application to real-world study designs.

The remainder of this paper is organized as follows. Section 3.2 introduces notation and the methods used for adapting the DiProPerm to the RLC study setting. Section 3.3 presents simulation experiments assessing the type I error and power performance of the DiProPerm. Section 3.4 demonstrates the application of the DiProPerm test on a real-world RLC data set. Section 3.5 discusses the future implications and limitations of the performance assessment and provides closing remarks and a summary of the entire paper.

3.2 Methods

Before we define the notation for an RLC problem, we first give a brief overview of the DiProPerm test. Let $U_1, \dots, U_{n_1} \sim F_1$ and $V_1, \dots, V_{n_2} \sim F_2$ be independent random samples of p dimensional random vectors from multivariate distributions F_1 and F_2 where $p \gg n_1, n_2$ and $n = n_1 + n_2$. The DiProPerm tests

$$H_0: F_1 = F_2 \text{ versus } H_1: F_1 \neq F_2$$

The general idea of the DiProPerm can be explained in three steps.

1. Direction: Find the normal vector to the separating hyperplane between two samples after training a binary linear classifier.
2. Projection: Project data on to the normal vector and calculate a univariate two-sample statistic.
3. Permutation: Compare the univariate statistics using a permutation test:
    a. permute class membership after pooling samples,
    b. re-train binary classifier and find the normal vector to the separating hyperplane,
    c. recalculate the univariate two sample statistic,
    d. repeat a-c multiple times (e.g., 1000) to determine the sampling distribution of the test statistic under the null H_0, and
    e. compute p-value by comparing the observed statistic to the sampling distribution.

In order to transcribe the DiProPerm to the RLC paradigm, consider the following RLC setup. Suppose there are n animals in a study. Each animal is repeatedly challenged with a pathogen of interest (e.g., simian HIV). After each challenge, the animal is assessed for infection. If an animal is uninfected, the challenges continue; otherwise, the challenges cease. Data from such studies is naturally handled in a discrete time survival analysis framework. In particular, $\tilde{T}_i$ denotes the number of challenges until infection if the animal was challenged indefinitely. In practice, challenges typically cease for uninfected animals after a set number of challenges, say c_{max} (in general, c_{max} may differ between animals, but for simplicity here, it is assumed to be same across animals). Thus, the discrete survival time $\tilde{T}_i$ may be right censored. That is, instead of observing $\tilde{T}_i$, we observe $T_i^{obs} = min(\tilde{T}_i, c_{max})$ as well as the event indicator $Y_i = I(\tilde{T}_i \leq c_{max})$. In addition, for each animal, we observe $X_i = (X_{i1}, \dots, X_{ip})$ a vector of p baseline covariates.

One inferential goal in RLC studies is to characterize the extent to which one or more of the baseline covariates $X_{i1}, \dots, X_{ip}$ are associated with the time until infection $\tilde{T}_i$. The DiProPerm is a tool that can be used to describe which baseline covariates contribute the most toward prolonging infection time. That is, the DiProPerm test can be applied to RLC data to test whether or not, given a set of antibody and functional measures, animals infected early (i.e., before the median infection time) differ from those infected late (i.e., after the median infection time). To adapt the DiProPerm to RLC data, let $Z_i = I(T_i^{obs} \leq median(T_1^{obs}, \dots, T_n^{obs}))$ be an indicator for an animal being infected early and let $n = n_1 + n_2$, where n_1 is the number of animals infected before the median infection time and n_2 is the number of animals infected after the median infection time. In addition, let $U = \left(U_1^T, \dots, U_{n_1}^T\right)$ be the covariate matrix for animals infected early and let $V = \left(V_1^T, \dots, V_{n_2}^T\right)$ be the covariate matrix for animals infected late. Therefore, $U \sim F_1$ and $V \sim F_2$ where F_1 and F_2 are p dimensional multivariate distributions and the DiProPerm tests

$$H_0: F_1 = F_2 \text{ versus } H_1: F_1 \neq F_2$$

However, the use of the DiProPerm test requires one to select a direction on which to fit the binary classifier and to select a univariate statistic for the projection step. For the direction step, the distance-weighted discrimination direction is recommended for HDLSS data (Marron et al., 2007). The DWD was developed specifically for $p \gg n$ problems and has many attractive qualities such as being good at separation and robust to data piling. For the projection step, the mean difference statistic was selected as our univariate statistic. The mean difference statistic is calculated using the projected values on the normal vector separating the hyperplane from the direction step. In other words, suppose $u_1, \dots, u_{n_1}$ and $v_1, \dots, v_{n_2}$ are the projected values from samples U and V respectively. The univariate mean difference statistic is calculated as $|\bar{u} - \bar{v}|$.

For the permutation step, the infected early indicator, Z_i, is permuted after pooling the two samples, then the mean difference statistic is calculated for the new permuted sample. This process is repeated for a set number of permutations (e.g., 1,000 permutations) to produce a permutation distribution of mean difference statistics. Two-sided p-values are calculated to be the proportion of permuted statistics higher than the observed value. If the p-value is less than the level of significance, α, the DiProPerm test rejects the null hypothesis and concludes there is a significant difference between the distributions of animals infected early and those infected late given a set of antibody and functional measures. However, the DiProPerm does not indicate explicitly which antibody and functional measures are most associated with being infected early. In order to characterized which antibody and functional measures drive the separation between the infected early and infected late distributions, one can look at the loadings of the DWD classifier (An et al., 2016; Nelson et al., 2019). The DWD loadings represent the relative contribution of each variable to the class difference. A higher absolute value of a variable's loading indicates a greater contribution of that variable to the class difference.

The type I error and power of the DiProPerm on RLC studies when n is small and p is large were evaluated via a simulation study. For the simulation study, all computations were performed in R 3.6.1. All statistical tests were two-sided with $\alpha = 0.05$. The R package *diproperm* was created for the purpose of this paper and is freely available on CRAN and GitHub (https://github.com/allmondrew/diproperm).

3.3 Simulation

A simulation study was conducted to explore the effects of the type 1 error and power of the DiProPerm in RLC settings. The simulation study consisted of varying the total sample size, n; the true coefficient parameters, β; the number of non-zero predictors, k; and the number of

total predictors, p. Values of $p = 25, 50$; $n = 10, 20, 30$, and $k = 1, 3, 5, 7$ were used across simulations. Higher values of p were considered but were limited by computation time. The maximum number of challenges was $c_{\max} = 20$, the number of permutations was 1,000, and the number of simulations was 500 for each scenario. The DWD direction was chosen for the DiProPerm direction step, the mean difference univariate statistic was chosen for the projection step, and a balanced permutation was conducted in the permutation step. The permutations were balanced in the sense that after relabeling, the permuted early-infected group contains $n_1/2$ members from the observed early-infected group and $n_1/2$ members from the observed late-infected group.

Recall, in the previous chapter, that the hazard function for RLC data can be defined as

$$h\left(\tilde{T}_i = t \middle| \tilde{T}_i \geq t, X_i = x_i\right) = \frac{e^{\gamma_{0t} + x_i\beta}}{1 + e^{\gamma_{0t} + x_i\beta}} = logit^{-1}(\gamma_{0t} + x_i\beta)$$

where $logit^{-1}(\gamma_{0t})$ represents the baseline hazard function at time t corresponding to animals with covariates $x_i = (0, \dots, 0)$ and β is the regression coefficient. In order to represent realistic RLC scenarios, the baseline hazard and regression coefficients were chosen in such a way that the mean infection probability per exposure was between 0.2 and 0.3. For $j = 1, \dots, p$, covariate vectors $X_i = (X_{i1}, \dots, X_{ip})$ were drawn from a MVN distribution with $\mu = 0$ and covariance matrix Σ with 1s along the diagonal and 0.05 along the off-diagonals.

The RLC data were simulated as follows.

For each $i = 1, \dots, n$,

1) Sample X_i from $MVN(0, \Sigma)$.
2) For $t = 1, \dots, c_{max}$, sample binary event Y_{it} from $Bern\big(h(t|x_i)\big)$.
    a. If $Y_{it} = 1$, then set $T_i^{obs} = t$ and stop.

b. Otherwise, if $Y_{it} = 0$ and $t = c_{max}$, then set $T_i^{obs} = c_{max}$ and stop.

c. Otherwise, increment t by 1 and repeat step 2.

3) If $T_i^{obs} \leq median(T_1^{obs}, \ldots, T_n^{obs})$ then $Z_i = -1$, otherwise $Z_i = 1$.

Small, moderate, and large effect true parameter coefficients, β, were used for each simulation. Table 3.1 shows the various combinations of linear effects and non-zero coefficients used for the small, moderate, and large effect scenarios.

For assessing type I error, we let $k = 0$ such that the two distributions between the early and late infected animals were identical. The proportion of times the DiProPerm test's p-value was less than $\alpha = 0.05$ over all simulations was considered the type I error, that is, the number of times one incorrectly rejects the null hypothesis when it is true. The results for type I error are summarized in Table 3.2. All scenarios in Table 3.2 had a type I error less than or equal to α. Thus, the DiProPerm preserved type I error regardless of n or p. In other words, if the DiProPerm detected a difference between the two distributions of early and late infection, then the probability this detection was a false positive was at most $\alpha = 0.05$ even when n is small and p is large.

For assessing the power of the DiProPerm in RLC settings, we let $k \neq 0$ such that the two distributions of early and late infected animals were not identical. Then, the proportion of times the DiProPerm test's p-value was less than $\alpha = 0.05$ over all simulations was considered the power – that is, the number of times one correctly rejects the null hypothesis in favor of the alternative that the distributions are not identical. Figure 3.1 shows the power of the DiProPerm test across each simulation scenario.

From Figure 3.1, the power decreased as the number of predictors increased. However, the power increased as the sample size, effect size, and number of non-zero coefficients

increased for both $p = 25$ and $p = 50$. For $n = 10$, the power of $p = 25$ was marginally higher than $p = 50$ even as the number of non-zero coefficients and effect size increased. However, for $n = 30$, the number of non-zero coefficients had a major impact on the power of $p = 25$ versus $p = 50$, with $p = 25$ having a roughly 10–20% higher power than $p = 50$ as k increased. The maximum power achieved across all scenarios for when $n = 10$ was around 20%, for when $n = 20$ was about 30%, and for when $n = 30$ was about 60%. A power of 80% was achieved when $n = 30$, $k = 8$, and $p = 50$ with a large effect in one direction. That is, the β vector consisted of eight positive large effects. A power of 82% was achieved when $n = 50$, $k = 10$, and $p = 50$ assuming a large effect size, five positives, and five negatives.

In order to assess whether the DiProPerm's power could be improved, we compared the DiProPerm's power to that of the correlation test for when $k = 1$. Figure 3.2 shows the power curves of the DiProPerm versus the correlation test for when $p = 25$ and $p = 50$. As the number of predictors increased, the power slightly decreased for both the correlation test and the DiProPerm. However, as the sample size and effect size increased, the power of the correlation test was greater than the DiProPerm's, especially in the moderate and large effect size scenarios. For $n = 10$, the correlation test's power was marginally higher than the DiProPerm's power, especially when the effect size was small or moderate. However, when $n = 30$, the correlation test's power was higher than the DiProPerm's, achieving a maximum power of 85% compared to the DiProPerm's 16% for when $n = 30$ with a large effect size. Therefore, when n is really small, if one were an oracle and knew which non-zero coefficient was associated with the outcome, one would only do a little better than with the DiProPerm, suggesting that minimal improvement can be achieved for the DiProPerm when n is small and p is large.

3.4 Application

In this section, a motivating example is given using the MIV02 data mentioned in section 1.5. The median time until infection was five challenges, with $n_1 = 7$ monkeys infected early and $n_2 = 7$ infected late. Figure 3.3 shows the results of the DiProPerm on the MIV02 data set. The top graph is the observed projection score distribution of the two classes, the two middle graphs are the projection score distributions of the permutation with the smallest and largest test statistic value, and the bottom graph is the test statistic permutation distribution and the observed statistic value marked by the red dotted line. The class label "-1" designates monkeys infected early and the label "1" indicates monkeys infected late. From the observed projection score distribution plot, there is overlap between the two distributions for monkeys infected early and those infected late in the DWD direction. Also, the minimum and maximum mean difference value graphs overlap, suggesting that the two distributions might be similar. Looking at the permutation distribution, the observed mean difference statistic on the DWD direction was 258.37 (p-value $= 0.762$). Therefore, the permutation p-value suggests there is no difference between the distribution of monkeys infected early and those infected late given their antibody and functional features.

Despite not detecting a difference between the two distributions, we continue with caution to assess the loadings of the DWD direction. Table 3.3 shows a list of the five highest absolute value DWD loadings along with their names and indexes in the data set. The top five loadings were CD3, activation CD69 total, CD20, eosinophil count, and the TFP/TFP category of the TRIM5 genotype, in that order. The main target cells for HIV/SIV/SHIV are CD3 and CD4 T cells. However, CD3 is a marker of all T cells, including the CD3 and CD8 T cells, which cannot be infected. Therefore, it is unknown why CD3 frequencies should help prolong infection

when there is no detection of CD4 and CD8 T cells. From the MIV02 study, CD69 was shown to be an early T-cell activation marker for SHIV, and the TRIM5 genotype TFP/TFP has been shown to confer resistance to SHIV (F. Wu et al., 2016). Thus, the inclusion of CD69 and TFP/TFP is not unreasonable, since lower counts of CD69 and the presence of the TFP/TFP TRIM5 genotype help prolong infection time. However, CD20 is a surface expressed on all B cells. B cells are the cells that produce antibodies, but they also present viral antigens to T cells that can then exert antiviral functions. Therefore, if B cells play a role in prolonging infection time, it is likely more indirect. For eosinophils, there is no former evidence suggesting eosinophils may help prolong infection time. Some of the top five variables of the DWD loadings for prolonging infection time are consistent with what has been shown in the literature, while the other variables may or may not help prolong infection time. Therefore, one might want to formulate future hypotheses for these five variables in future RLC studies for prolonging infection time.

3.5 Discussion

The DiProPerm test is a machine learning approach for assessing whether or not two high-dimensional distributions are identical when the sample size is small and number of predictors is large. The DiProPerm test has never before been adapted to the RLC paradigm when n is small and p is large. In this article, we suggest a way to adapt the DiProPerm to RLC studies by separating animals into two classes (i.e., early vs. late infections) based on median infection time. The power and type I error of the DiProPerm were evaluated via a simulation study with an application to a real-world RLC data set. Simulated data showed that the DiProPerm protected against false positives regardless of the sample size and number of predictors. Thus, if the DiProPerm detects that there is a difference between the early and late infected distributions, then there is a high probability that this difference is a true positive.

The sample size needed for achieving a high power is heavily dependent on the number of predictors, the number of non-zero coefficients, and the effect sizes in the data. The DiProPerm did not achieve 80% power until either the sample size was as large as 50 and number of large effect non-zero predictors was 10 – five positives and five negatives – or the sample size was 30 with eight non-zero, large effect positive predictors out of total of 50 predictors. Simulations showed that the power increased as the sample size, number of non-zero coefficients, and effect size increased but decreased as the number of predictors increased. This suggests that for RLC problems with a lot of predictors and low sample size, the DiProPerm is less powerful. However, compared with the correlation test, we argue that there is not much room for improvement for the DiProPerm in these settings when n is small and p is large. That is, when n is really small, if one were an oracle and knew which non-zero coefficient was associated with the outcome, one would only do a little better than with the DiProPerm using the correlation test. Therefore, the DiProPerm improves upon the inference of RLC studies when n is small and p is large.

As shown in the previous chapter, penalized regression techniques have a low chance of selecting true positives and a high chance of selecting false positives for RLC data when n is small and p is large. However, the DiProPerm improves upon penalized regression for RLC data by having a higher chance of detecting a true signal and a lower chance of detecting a false positive. The DiProPerm also has more power than penalized regression techniques for RLC data, detecting fewer false negative signals than penalized regression. Additionally, the DiProPerm allows one to explore which variables may contribute most toward prolonging an animal's time until infection, which is information penalized regression techniques cannot reliably provide. If the DiProPerm detects a difference between early and late infected animals,

then the highest absolute loadings in the DWD direction are the variables that are driving most of this separation between distributions. And because the DiProPerm preserves type I error, one can conclude with a high probability that these variables are indeed associated with prolonging infection time. The DiProPerm, for RLC studies when n is small and p is large, is a very strong tool that can discover potential associations between various antibody and functional covariates and prolonged infection time, thereby improving the development of future vaccine candidates. The DiProPerm should not be used for making scientific claims in RLC data about which variables are associated with time to infection. Rather, the DiProPerm can be used to generate rational, data-driven hypotheses for future RLC studies.

3.6 Acknowledgements

This work was partially supported by National Institutes of Health grants P01 AI117915, R37AI054165, and P30 AI50410. The content of this paper is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

Table 3.1. Linear effect coefficients, β, for each number, k, of non-zero coefficients in the true model

Effect Size	k	β
Small	1	(log(1.5), 0,…,0)
	3	(1, -1, log(1.5), 0,…,0)
	5	(1, -1, 1, -1, log(1.5), 0,…,0)
	7	(1, -1, 1, -1, 1, -1, log(1.5), 0,…,0)
Moderate	1	(log(2), 0,…,0)
	3	(1, -1, log(2), 0,…,0)
	5	(1, -1, 1, -1, log(2), 0,…,0)
	7	(1, -1, 1, -1, 1, -1, log(2), 0,…,0)
Large	1	(log(3), 0,…,0)
	3	(1, -1, log(3), 0,…,0)
	5	(1, -1, 1, -1, log(3), 0,…,0)
	7	(1, -1, 1, -1, 1, -1, log(3), 0,…,0)

Table 3.2. Type I error assessment for the DiProPerm

N	$p = 50$	$p = 100$	$p = 200$
10	0.05	0.05	0.04
20	0.04	0.04	0.05
30	0.03	0.03	0.04

Table 3.3. Top 5 DWD loadings from fitting the DiProPerm on MIV02 data set

Variable	Index	DWD Loading
CD3	2	0.43
Activation CD69 Total	27	-0.36
CD20	1	0.35
Eosinophils	20	-0.22
TRIM5 Genotype ("TFP/TFP")	138	-0.15

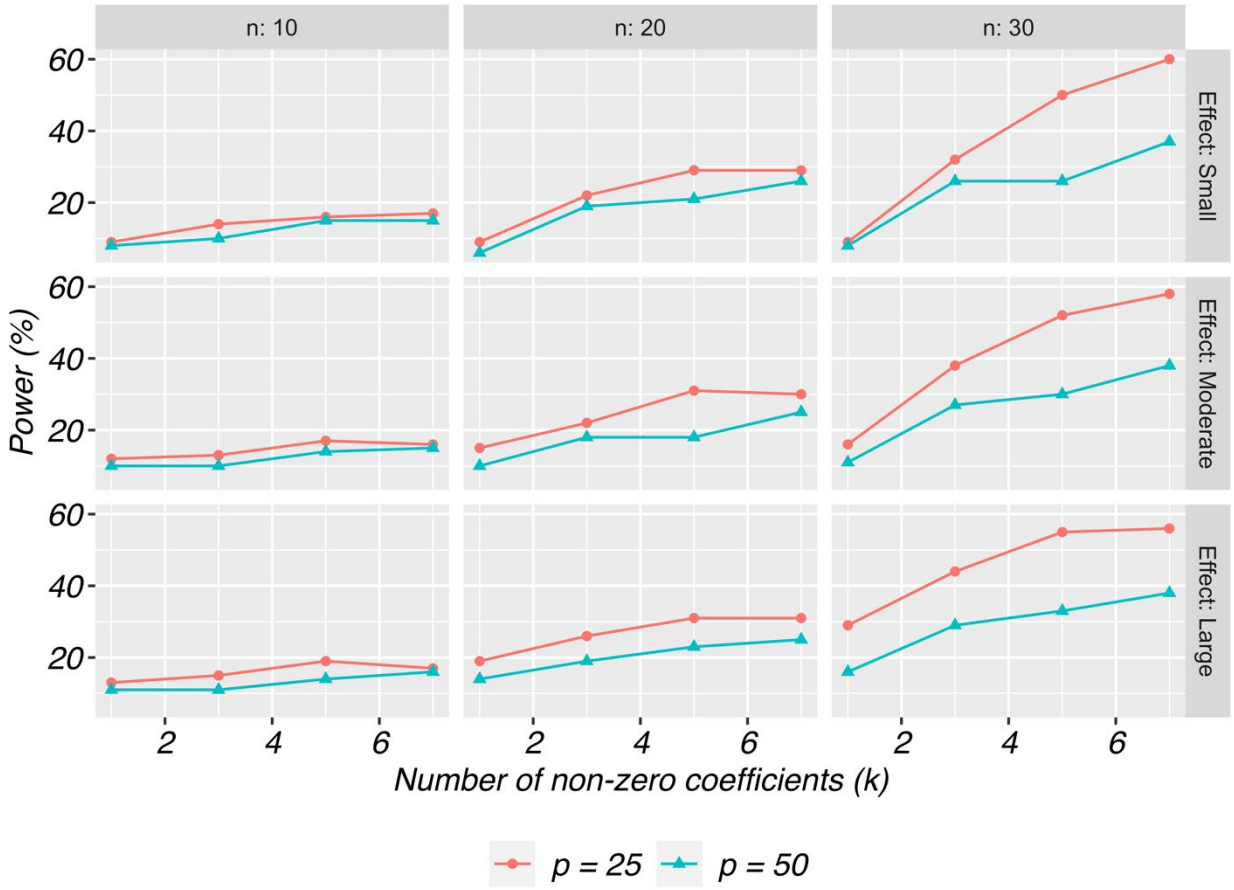


Figure 3.1. Power assessment of the DiProPerm by varying sample size, effect size, and the number of non-zero coefficients.

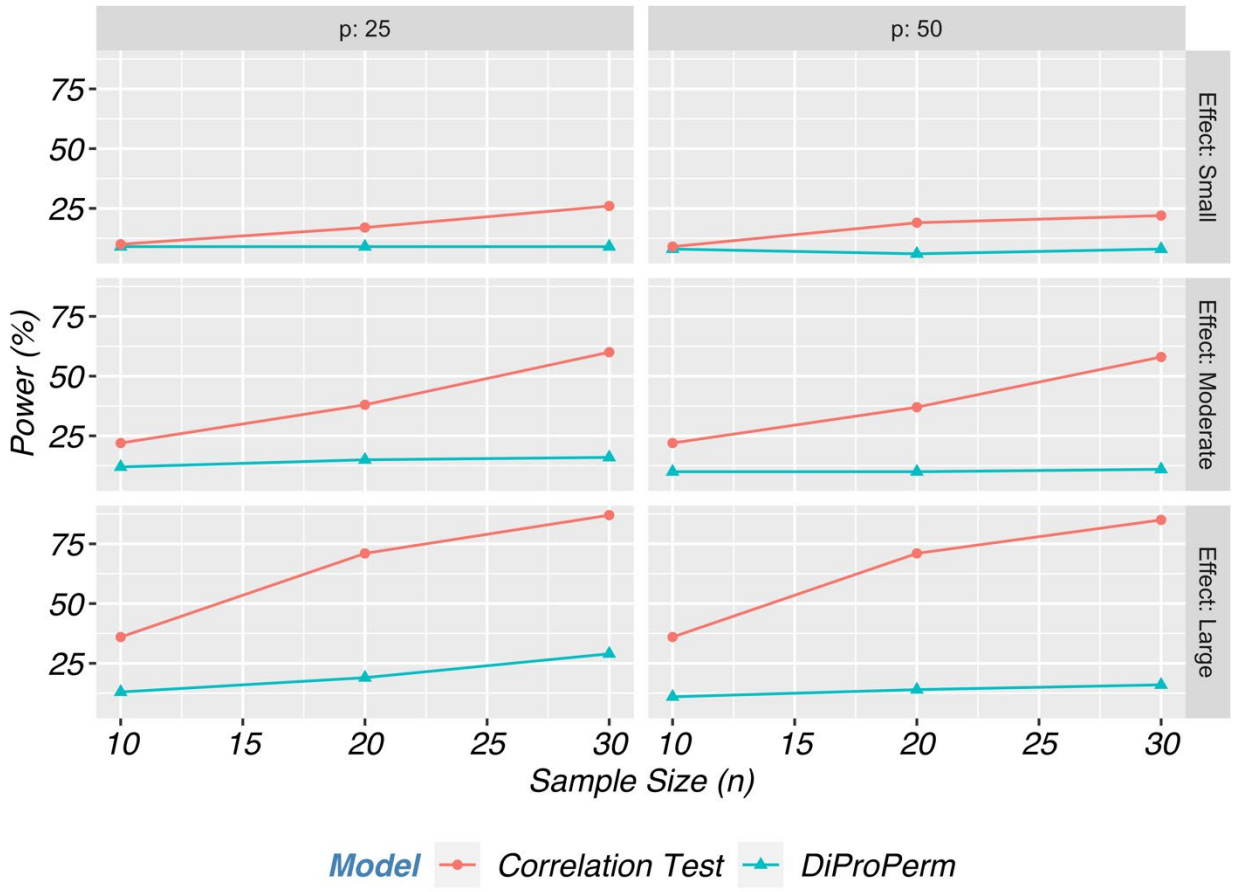


Figure 3.2. Power comparison between the correlation test and the DiProPerm by varying sample size, effect size, and the number of predictors for when $k = 1$.

Figure 3.3. Diagnostic plot for fitting the DiProPerm on the MIV02 study data. The top graph is the observed projection score distribution of the two classes, the two middle graphs are the projection score distributions of the permutation with the smallest and largest test statistic value, and the bottom graph is the test statistic permutation distribution and the observed statistic value marked by the red dotted line.

CHAPTER 4: DIPROPERM: A SOFTWARE PACKAGE FOR THE DIPROPERM TEST

4.1 Introduction

Advancements in modern technology and computer software have dramatically increased the demand and feasibility to collect high-dimensional data sets. Such data possess challenges which require the creation of new and adaptation of existing statistical methods. One such challenge is that we may observe many more predictors, p, than the number of observations, n, especially in small sample size studies. These data structures are known as high-dimensional, low sample size (HDLSS) data sets, or "small n, big p".

High-dimensional low sample size data emerge frequently in many health-related fields. For example, in genomic studies, a single microarray experiment might produce tens of thousands of gene expressions compared to the few samples studied, often being less than a hundred (Alag, 2019). In medical imaging studies, a single region of interest for analysis in an MRI or CT-scan image often contains thousands of features compared to the small number of samples studied (Limkin et al., 2017). In pre-clinical evaluation of vaccines and other experimental therapeutic agents, the number of biomarkers measured (e.g., immune responses) may be much greater than the number of samples studies (e.g., mice, rabbits, or non-human primates) (Kimball et al., 2018).

One common task in the HDLSS setting entails constructing a classifier which appropriately assigns samples to the correct class. For example, in pre-clinical studies investigators may wish to predict whether an animal survives to a certain time point based on high-dimensional biomarker data. When the data are to be partitioned into two classes, binary

linear classifiers have been shown to be especially useful in HDLSS settings and preferable to more complicated classifiers because of their ease of interpretability (Aoshima et al., 2018). However, linear classifiers may find spurious linear combinations in HDLSS settings (Marron et al., 2007). That is, a binary linear classifier may find, for two identical high-dimensional distributions, a linear combination of features which incorrectly suggests the two classes are different. Thus, it is important to assess whether a binary linear classifier is detecting a statistically significant difference between two high-dimensional distributions.

4.2 DiProPerm

The direction-projection-permutation (DiProPerm) test was developed to test whether or not a binary linear classifier detected a difference between two high-dimensional distributions (Wei et al., 2016). DiProPerm uses one-dimensional projections of the data based on the binary linear classifier to construct a univariate test statistic, and then permutes class labels to determine the sampling distribution of the test statistic under the null. Importantly, the DiProPerm test is exact, i.e., the type I error is guaranteed to be controlled at the nominal level for any sample size.

To better understand the mechanics of DiProPerm, let $U_1, \dots, U_n \sim F_1$ and $V_1, \dots, V_m \sim F_2$ be independent random samples of p dimensional random vectors from multivariate distributions F_1 and F_2 where p may be larger than m and n. The DiProPerm tests

$$H_0: F_1 = F_2 \text{ versus } H_1: F_1 \neq F_2$$

The general idea of the DiProPerm can be explained in three steps.

1. Direction: Find the normal vector to the separating hyperplane between two samples after training a binary linear classifier.
2. Projection: Project data on to the normal vector and calculate a univariate two-sample statistic.

3. Permutation: Compare the univariate statistics using a permutation test:
    a. permute class membership after pooling samples,
    b. re-train binary classifier and find the normal vector to the separating hyperplane,
    c. recalculate the univariate two sample statistic,
    d. repeat a-c multiple times (e.g., 1000) to determine the sampling distribution of the test statistic under the null H_0, and
    e. compute p-value by comparing the observed statistic to the sampling distribution.

Different binary linear classifiers may be used in the first step of DiProPerm. Linear discriminant analysis, particularly after conducting principal component analysis, is one possible classifier for the direction step. However, using LDA with PCA in the HDLSS setting has some disadvantages, including a lack of interpretability, a sensitivity to outliers, and a tendency to find spurious linear combinations due to a phenomenon known as data piling (Aoshima et al., 2018; Marron et al., 2007). Data piling occurs if data are projected onto some projection direction and many of the projections are the same, or piled on one another. The support vector machine (SVM) is a another popular classifier (Hastie et al., 2001). The SVM finds the hyperplane that maximizes the minimum distance between data points and the separating hyperplane. However, the SVM can also suffer from data piling in the HDLSS setting. To overcome data piling, the distance-weighted discrimination (DWD) classifier was developed (Marron et al., 2007). The DWD classifier finds the separating hyperplane minimizing the average inverse distance between data points and the hyperplane. The DWD performs well in HDLSS settings with good separation and is more robust to data piling.

In the second step of DiProPerm, a univariate statistic is calculated using the projected values on to the normal vector to the separating hyperplane from the first step. Suppose $u_1, \dots, u_n$

and $v_1, \ldots, v_m$ are the projected values from samples $U_1, \ldots, U_n$ and $V_1, \ldots, V_m$ respectively. One common choice for the univariate test statistic for DiProPerm includes the difference of means statistic, $|\bar{u} - \bar{v}|$. Other two-sample univariate statistics such as the two-sample t-statistic or difference in medians are also possible for use with the DiProPerm.

The last step of DiProPerm entails determining the distribution of the test statistic under the null. In this step, the two samples are pooled, class labels are permuted, then a univariate statistic is calculated. Repeat this process multiple times (say 1000) to determine the sampling distribution of the test statistic under the null H_0. Two-sided p-values are then calculated by the proportion of statistics higher than the original value.

When the DiProPerm test is implemented using the DWD classifier, it is common practice to look at the loadings of the DWD classifier (An et al., 2016; Nelson et al., 2019). The DWD loadings represent the relative contribution of each variable to the class difference. A higher absolute value of a variable's loading indicates a greater contribution for that variable to the class difference. Combining the use of the DiProPerm and evaluation of the DWD loadings in applications can provide insights into high-dimensional data and be used to generate rational hypotheses for future research.

The DiProPerm test has been used in several areas of biomedical research including osteoarthritis and neuroscience (An et al., 2016; Bendich et al., 2016; Nelson et al., 2019). However, currently there does not exist an R package which implements DiProPerm. Therefore, we developed *diproperm*, a simple, free, publicly available R software package to analyze data from two high-dimensional distributions. *diproperm* displays diagnostic plots for a specified univariate statistic and calculates p-values for the DiProPerm test. The loadings for the binary

linear classifier are also available for display in order from highest to lowest relative to their contribution toward the separation of the two distributions.

The remainder of this paper is organized as follows. Section 4.3 describes the use of the *diproperm* package and provides an example on simulated data. Section 4.4 demonstrates the use of the *diproperm* package on a real-world data set. Section 4.5 provides closing remarks and a summary of the entire paper.

4.3 The *diproperm* package

The *diproperm* package is comprised of three functions:

- DiProPerm(): Conducts a DiProPerm test
- plotdpp(): Plots diagnostics from the DiProPerm test
- loadings(): Returns the variable indices with the highest loadings in the binary classification. The absolute values of the loading values indicate a variable's relative contribution toward the separation between the two classes.

4.3.1 *diproperm* example

The example below creates a Gaussian data set containing 100 samples, 2 features, clustered around 2 centers with a standard deviation of 2. The class labels are then re-classified to -1 and 1 to match the input requirements of DiProPerm(). The DiProPerm test is then conducted using the DWD classifiers, the mean difference univariate statistic, and 1000 permutations. The results from DiProPerm() are then displayed with plotdpp(). Last, the top five indices of the highest absolute loadings are listed.

```
devtools::install_github("elbamos/clusteringdatasets")
library(clusteringdatasets)

cluster.data <- make_blobs(n_samples = 100, n_features = 2, centers = 2,
cluster_std = 2)

X <- cluster.data[[1]]
y <- cluster.data[[2]]
```

```
y[y==2] <- -1

dpp <- DiProPerm(X,y,B=1000,classifier = "dwd",univ.stat = "md")

plotdpp(dpp)

loadings(dpp,loadnum = 5)
```

4.3.2 Description

The main function to be called first by the user is DiProPerm(), which takes in an $n \times p$ data matrix and a vector of n binary class labels both provided by the user. Factor variables for the data matrix must be coded as 0/1 dummy variables and the class labels for the vector of binary class labels must be coded as -1 and 1. By default the DiProPerm() uses the DWD classifier, the mean difference as the univariate statistics, and 1000 balanced permutations. The permutations are balanced in the sense that after relabeling, the new -1 group contains $n/2$ members from the original -1 group and $n/2$ members not from the original -1 group. DiProPerm() implements DWD from the genDWD function in the *DWDLargeR* package (Lam et al., 2018a, 2018b). The penalty parameter, C, in the genDWD function is calculated using the penaltyParameter function in *DWDLargeR*. More details on the algorithm used to compute genDWD and penaltyParameter can be found in Lam et al. (2018a). Another option included in DiProPerm() for the binary linear classifier is "md", mean difference direction. Users can also implement an unbalanced, randomized permutation design if desired. DiProPerm() uses parallel processing to delegate computation to the number of cores on the user's computer for increased efficiency. DiProPerm() returns a list of the observed data matrix, vector of observed class labels, observed test statistic, projection scores used to compute the observed test statistic, the loadings of the binary classification, the z-score, cutoff value for an α level of significance, the p-value for the DiProPerm test, a list of each permutation's projection scores and permuted class labels, and a vector of permuted test statistics the size of the number of permutations used.

After fitting the DiProPerm(), the user can use plotdpp() to create a panel plot for assessing the diagnostics of the DiProPerm test. plotdpp() takes in a DiProPerm list and the user may specify which diagnostics they would like to display. By default, plotdpp() displays a facet plot with the observed score distribution, the projection score distribution of the permutation with the smallest test statistic value, the projection score distribution of the permutation with the largest test statistic value, and the test statistic permutation distribution. For the permutation distribution plot, the z-score, cutoff value, observed test statistic and p-value are displayed on the graph. Larger, individual graphs may be displayed by using the plots option in plotsdpp(). Additional graphs include the projection score distributions for the first permutation and second permutations. The diagnostic plots show the user the characteristics of their data and facilitate the visual assessment of the separation of the two high-dimensional distributions being tested.

Lastly, after calling the DiProPerm(), the user may call the loadings() function. The loadings() function returns the variable indexes in the data matrix which have the highest absolute loadings in the binary classification. Higher absolute loading values indicate a greater contribution for a particular variable toward the separation between the two classes. By default, loadings() returns the indices for all variables sorted by their absolute loading value. Therefore, the top variable index is the variable which contributes the most toward the separation of the two classes and the last variable is the one which contributes the least. The user may also change the number of loadings displayed.

4.4 Application

To illustrate use of the *diproperm* package, consider the mushrooms data set which is freely available from the UCI Machine Learning Repository (Dua & Graff, 2019) and within *diproperm*. This data set includes various characterizations of 23 species of gilled mushrooms in the Agaricus and Lepiota family. Each mushroom species is labeled as either definitely edible or

definitely poisonous/unknown. There are $n = 8124$ mushrooms total, and $p = 112$ binary covariates coded as 0/1 corresponding to 22 categorical attributes. Below we demonstrate the diproperm package functionality using data from the first $n = 50$ mushrooms in the data set.

Step 1: Load and clean the data

```
devtools::install_github("allmondrew/diproperm")
library(diproperm)
data(mushrooms)
```

The above code installs the *diproperm* package and loads the mushroom data into R. Now check the structure of the data to make sure it is compatible with DiProPerm().

```
dim(mushrooms$X)
[1] 112 8124

table(mushrooms$y)
  -1    1
4208 3916
```

The vector of class labels must be -1 or 1 for DiProPerm() which is the case for this data; however, the data set is in $p \times n$ format. For DiProPerm(), the dataset must be in $n \times p$ format. This can be done using the transpose function from the *Matrix* package in R (Bates & Maechler, 2019). After taking the transpose, we subset the data and vector of class labels to the first 50 observations and store the results.

```
X <- Matrix::t(mushrooms$X)
X <- X[1:50,]
y <- mushrooms$y[1:50]
```

Step 2: Conduct DiProPerm

Now that the data is in the proper format the call to DiProPerm() is as follows.

```
dpp <- DiProPerm(X=X,y=y,B=1000)

 Algorithm stopped with error 2.35e-08

 sample size = 50, feature dimension = 112
 positive sample = 12, negative sample = 38
 number of iterations = 51
 time taken = 0.39
 error of classification (training) = 0.00 (%)
```

```
 primfeas = 3.49e-10
 dualfeas = 0.00e+00
 relative gap = 2.89e-07
```

Characteristics of the DWD algorithm used to find the solution for the observed data are displayed by DiProPerm(). The algorithm took 51 iterations and 0.39 seconds to converge to the tolerance threshold with a zero percent classification error on the training data set. The runtime for 1000 permutations was less than 3 minutes on a four-core machine but would be faster on a machine with more cores. The dpp object stores the output list from DiProPerm() described in the package. Storing the information allows us to plot the diagnostics in the next step.

Step 3: Plot diagnostics

```
plotdpp(dpp)
```

Figure 4.1 displays the default diagnostics for a DiProPerm list. From the observed projection score distribution, one can see clear separation between the two classes. Also, from the projected score distributions of the permutations which yield the smallest and largest test statistic, we see the score distributions overlap well so there is some visual justification that the distributions in the observed plot are truly different. Lastly, the bottom plot shows the sampling distribution under the null is located around 0.4 while the observed test statistic is greater than 2. Each individual plot can also be output by the following set of commands.

```
plotdpp(dpp,plots="obs")
plotdpp(dpp,plots="min")
plotdpp(dpp,plots="max")
plotdpp(dpp,plots="permdist")
```

The permutation p-value in Figure 4.1 suggests that the two high-dimensional distributions of mushroom attributes are indeed different between the two classes. Also displayed is a z-score, calculated by fitting a Gaussian distribution to the test statistic permutation distribution. The mushroom data z-score 13.2 indicates the observed test statistic is approximately 13 standard deviations from the expected value of the test statistic under the null

hypothesis. Finally, the cutoff value 0.678 is displayed, corresponding to the critical value for a hypothesis test at the 0.05 significance level.

Step 4: Examine loadings

In order to assess which variables contributed most toward the separation in step 3 we can print the top five contributors with the following code.

```
loadings(dpp,loadnum = 5)

  index sorted_loadings
1    29       0.5395016
2    37       0.3170037
3    36      -0.2481763
4   111       0.2228389
5    20      -0.2087244
```

The top five contributors toward the separation seen in the observed distribution in Figure 4.1 are indices 29, 37, 36, 111, and 20. These indices correspond to a pungent odor, narrow gill size, broad gill size, urban habitat, and yellow cap color, respectively. These results are similar to previous analyses which have also found odor, gill size, habitat, and cap color predictive of mushroom edibility (Pinky et al., 2019; Wibowo et al., 2018).

4.5 Summary

Binary linear classifiers can suffer from finding spurious separating directions in the HDLSS setting, i.e., data may be sampled from two identical distributions but the binary linear classifier may find a linear combination of features such that the two classes appear to be very different. The DiProPerm test was created to test whether or not the separation induced by the binary linear classifier is truly separate or just a result of over-fitting. The *diproperm* package allows the user to visually assess and empirically test if there is a difference between the high-dimensional distributions of the two classes and, if so, evaluate the key features contributing to the separation between the classes.

4.6 Acknowledgements

This work was supported by the National Institute of Allergy and Infectious Diseases (NIAID) of the National Institutes of Health (NIH), under award number R37AI054165. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

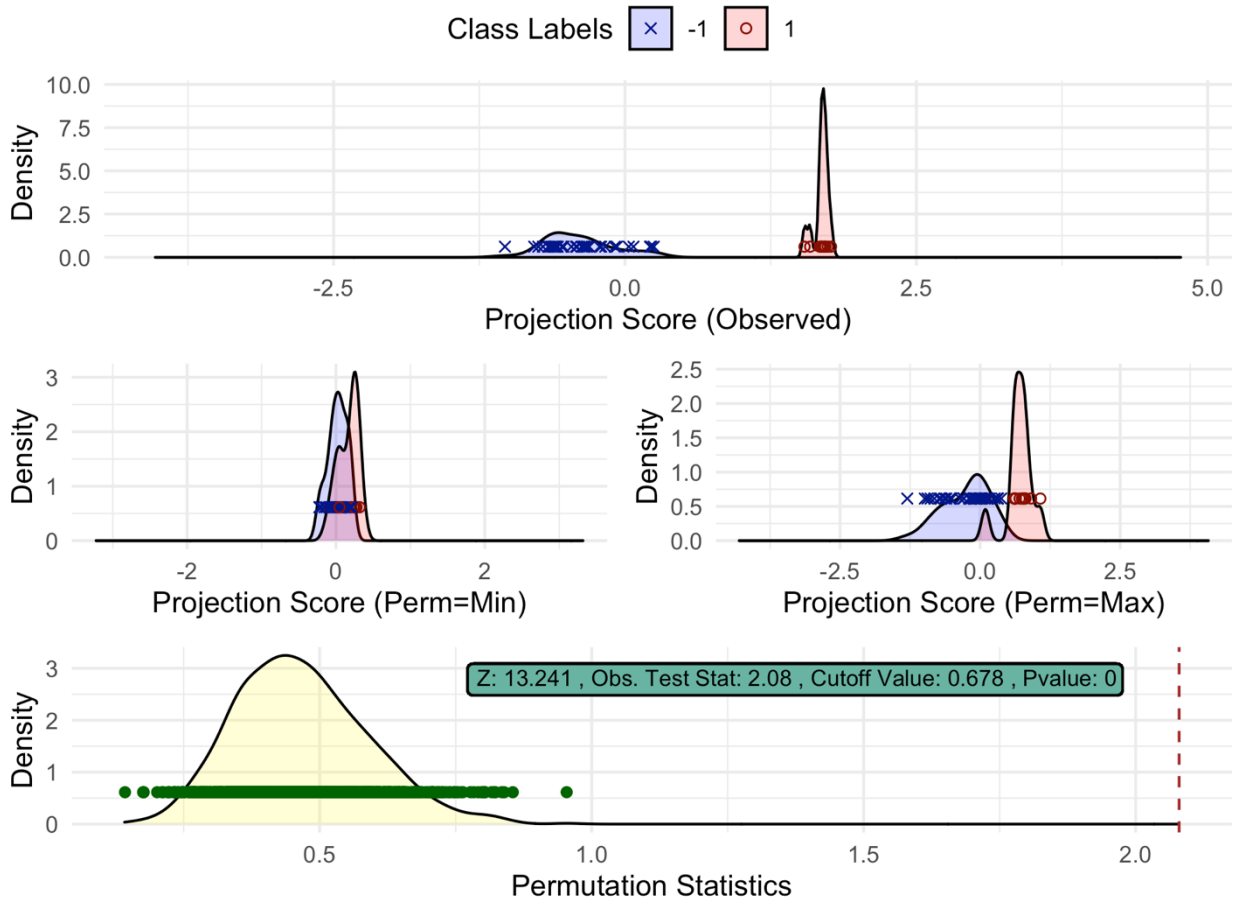


Figure 4.1. The diagnostic plot from plotdpp() for the mushrooms data set. The top graph is the observed projection score distribution of the two classes, the two middle graphs are the projection score distributions of the permutation with the smallest and largest test statistic value, and the bottom graph is the test statistic permutation distribution and the observed statistic value marked by the red dotted line.

CHAPTER 5: CONCLUSION

The exponential growth of modern technology and computational efficiency have exacerbated the need and ability to collect large data sets. However, there are many applications in public health research where small samples in pre-clinical and human assessment are unavoidable, particularly for the development of new drugs and vaccines. One such application is the repeated low-dose challenge study. In RLC experiments, animals who are assigned to an active or placebo candidate vaccine are repeatedly challenged (exposed) with some target pathogen, either until infection or until some maximum number of challenges is reached (Nolen et al., 2015). It is becoming increasingly popular in RLC studies to collect an immense number of variables in relation to a hyper-small sample size. In this dissertation, the performance of penalized regression techniques was described on RLC data when n is small and p is large, a novel method known as the direction-projection-permutation (DiProPerm) test was adapted to RLC data, and an associated *diproperm* R package was created and demonstrated on a real-world data set.

Penalized regression techniques, like the lasso, are sometimes used in RLC experiments where n is typically small and p is large. However, the performance of such methods is not well established for this experiment paradigm. In chapter 2, the lasso, elastic net, and a newly proposed discrete survival time penalized regression model were compared using simulated RLC data. The performances of these models were evaluated in simulation experiments and applied to a recent RLC study evaluating a candidate HIV vaccine. All three models rarely selected true positives regardless of the effect size, number of predictors, or the number of non-zero

coefficients, with many models containing only false positives. Thus, penalized regression models should be used cautiously in the RLC setting.

In chapter 3, the use of the DiProPerm test on RLC data was explored. The DiProPerm test was adapted to the RLC paradigm to test whether animals are more likely to become infected early (i.e., before the median infection time) as opposed to late given a set of antibody and functional measurements. The type I error and power of the DiProPerm test on RLC experiments were described in a simulation study. The classifier's loadings from the DiProPerm were evaluated to determine which variables had the most influence on median time to HIV infection. Simulation processes and real data applications revealed the advantages of the DiProPerm over penalized regression techniques on RLC data when n is small and p is large. The DiProPerm can help medical professionals conducting pre-clinical experiments in RLC studies to generate reasonable hypotheses regarding which types of functional measures help prolong infection time.

The goal of this dissertation was to improve upon the inferences regarding RLC data for when n is small and p is large. Simulation studies showed that penalized regression techniques like the lasso are unfavorable for use on RLC data because of the low probability of selecting a true positive and the high probability of selecting a false positive. Therefore, RLC investigators should be cautious when utilizing penalized regression techniques. Next, the DiProPerm test was adapted to the RLC paradigm and its characteristics were described by a simulation study. The DiProPerm test preserved type I error but had a low power for most scenarios unless all covariates had large effects in one direction (i.e., all positive or all negative effects). The DiProPerm test was applied to a real-world data set and the implications for the highest absolute value loadings of the classifier were described for the data set. Thus, the DiProPerm can be used for inference on RLC data when n is small and p is large to derive rational, data-driven

hypotheses for future research. Additionally, a DiProPerm R package was created for use by RLC investigators and anyone else interested in using the DiProPerm. The *diproperm* R package was demonstrated and applied to a real-world data set. Moving forward, penalized regression techniques on RLC studies are not recommended to make claims about the associations between covariates and the outcome. The DiProPerm test is a better alternative than penalized regression for use on RLC data when n is small and p is large and can be implemented with a user-friendly R package to make rational hypotheses for prospective research.

REFERENCES

Aban, I. B., & George, B. (2015). Statistical considerations for preclinical studies. *Experimental Neurology*, *270*, 82–87. https://doi.org/10.1016/j.expneurol.2015.02.024

Ackerman, M. E., Das, J., Pittala, S., Broge, T., Linde, C., Suscovich, T. J., Brown, E. P., Bradley, T., Natarajan, H., Lin, S., Sassic, J. K., O'Keefe, S., Mehta, N., Goodman, D., Sips, M., Weiner, J. A., Tomaras, G. D., Haynes, B. F., Lauffenburger, D. A., … Alter, G. (2018). Route of immunization defines multiple mechanisms of vaccine-mediated protection against SIV. *Nature Medicine*, *24*(10), 1590–1598. https://doi.org/10.1038/s41591-018-0161-0

Akaike, H. (1998). *Information Theory and an Extension of the Maximum Likelihood Principle BT - Selected Papers of Hirotugu Akaike* (E. Parzen, K. Tanabe, & G. Kitagawa (eds.); pp. 199–213). Springer New York. https://doi.org/10.1007/978-1-4612-1694-0_15

Alag, A. (2019). Machine learning approach yields epigenetic biomarkers of food allergy: A novel 13-gene signature to diagnose clinical reactivity. *PLOS ONE*, *14*(6), e0218253. https://doi.org/10.1371/journal.pone.0218253

An, H., Marron, J. S., Schwartz, T. A., Renner, J. B., Liu, F., Lynch, J. A., Lane, N. E., Jordan, J. M., & Nelson, A. E. (2016). Novel statistical methodology reveals that hip shape is associated with incident radiographic hip osteoarthritis among African American women. *Osteoarthritis and Cartilage*, *24*(4), 640–646. https://doi.org/10.1016/j.joca.2015.11.013

Aoshima, M., Shen, D., Shen, H., Yata, K., Zhou, Y.-H., & Marron, J. S. (2018). A survey of high dimension low sample size asymptotics. *Australian & New Zealand Journal of Statistics*, *60*(1), 4–19. https://doi.org/10.1111/anzs.12212

Bates, D., & Maechler, M. (2019). *Matrix: Sparse and Dense Matrix Classes and Methods*. https://cran.r-project.org/package=Matrix

Bendich, P., Marron, J. S., Miller, E., Pieloch, A., & Skwerer, S. (2016). Persistent Homology Analysis of Brain Artery Trees. *The Annals of Applied Statistics*, *10*(1), 198–218. https://doi.org/10.1214/15-AOAS886

Bradley, T., Pollara, J., Santra, S., Vandergrift, N., Pittala, S., Bailey-Kellogg, C., Shen, X., Parks, R., Goodman, D., Eaton, A., Balachandran, H., MacH, L. V., Saunders, K. O., Weiner, J. A., Scearce, R., Sutherland, L. L., Phogat, S., Tartaglia, J., Reed, S. G., … Haynes, B. F. (2017). Pentavalent HIV-1 vaccine protects against simian-human immunodeficiency virus challenge. *Nature Communications*, *8*, 1–15. https://doi.org/10.1038/ncomms15711

Breslow, N. (1974). Covariance Analysis of Censored Survival Data. *Biometrics*, *30*(1), 89–99. https://doi.org/10.2307/2529620

Cai, T., & Liu, W. (2011). A Direct Estimation Approach to Sparse Linear Discriminant Analysis. *Journal of the American Statistical Association*, *106*(496), 1566–1577.

https://doi.org/10.1198/jasa.2011.tm11199

Chaudhury, S., Duncan, E. H., Atre, T., Storme, C. K., Beck, K., Kaba, S. A., Lanar, D. E., & Bergmann-Leitner, E. S. (2018). Identification of Immune Signatures of Novel Adjuvant Formulations Using Machine Learning. *Scientific Reports*, *November*, 1–12. https://doi.org/10.1038/s41598-018-35452-x

Choi, I., Chung, A. W., Suscovich, T. J., Rerks-Ngarm, S., Pitisuttithum, P., Nitayaphan, S., Kaewkungwal, J., O'Connell, R. J., Francis, D., Robb, M. L., Michael, N. L., Kim, J. H., Alter, G., Ackerman, M. E., & Bailey-Kellogg, C. (2015). Machine Learning Methods Enable Predictive Modeling of Antibody Feature:Function Relationships in RV144 Vaccinees. *PLOS Computational Biology*, *11*(4), e1004185. https://doi.org/10.1371/journal.pcbi.1004185

Clemmensen, L., Witten, D., Hastie, T., & Ersbøll, B. (2011). Sparse Discriminant Analysis. *Technometrics*, *53*(4), 406–413. http://www.jstor.org/stable/41714953

Dua, D., & Graff, C. (2019). *UCI Machine Learning Repository*. https://archive.ics.uci.edu/ml/datasets/Mushroom

Eudailey, J. A., Dennis, M. L., Parker, M. E., Phillips, B. L., Huffman, T. N., Bay, C. P., Hudgens, M. G., Wiseman, R. W., Pollara, J. J., Fouda, G. G., Ferrari, G., Pickup, D. J., Kozlowski, P. A., Van Rompay, K. K. A., De Paris, K., & Permar, S. R. (2018). Maternal HIV-1 Env Vaccination for Systemic and Breast Milk Immunity To Prevent Oral SHIV Acquisition in Infant Macaques. *MSphere*, *3*(1), 1–21. https://doi.org/10.1128/msphere.00505-17

Fisher, R. A. (1936). Th Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, *7*(2), 179–188. https://doi.org/10.1111/j.1469-1809.1936.tb02137.x

Foster, D. P., & George, E. I. (1994). The Risk Inflation Criterion for Multiple Regression. *Ann. Statist.*, *22*(4), 1947–1975. https://doi.org/10.1214/aos/1176325766

Friedman, J. H., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software; Vol 1, Issue 1 (2010)* . https://www.jstatsoft.org/v033/i01

Groll, A. (2017). *Package ' glmmLasso .'* https://cran.r-project.org/web/packages/glmmLasso/index.html

Groll, A., & Tutz, G. (2014). Variable selection for generalized linear mixed models by L1-penalized estimation. *Statistics and Computing*, *24*(2), 137–154. https://doi.org/10.1007/s11222-012-9359-z

Groll, A., & Tutz, G. (2017). Variable selection in discrete survival models including heterogeneity. *Lifetime Data Analysis*, *23*(2), 305–338. https://doi.org/10.1007/s10985-016-9359-y

Hand, D. J. (2006). Classifier Technology and the Illusion of Progress. *Statist. Sci.*, *21*(1), 1–14. https://doi.org/10.1214/088342306000000060

Hastie, T., Tibshirani, R., & Friedman, J. H. (2001). *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. New York : Springer. https://catalog.lib.unc.edu/catalog/UNCb4019902

Hoerl, A. E., & Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, *12*(1), 55–67. https://doi.org/10.2307/1267351

Jolliffe, I. T. (1986). *Principal Component Analysis*. New York : Springer-Verlag, 1986. https://catalog.lib.unc.edu/catalog/UNCb2119567

Kimball, A. K., Oko, L. M., Bullock, B. L., Nemenoff, R. A., van Dyk, L. F., & Clambey, E. T. (2018). A Beginner's Guide to Analyzing and Visualizing Mass Cytometry Data. *The Journal of Immunology*, *200*(1), 3 LP – 22. https://doi.org/10.4049/jimmunol.1701494

Lam, X. Y., Marron, J. S., Sun, D., & Toh, K.-C. (2018a). *DWDLargeR: Fast Algorithms for Large Scale Generalized Distance Weighted Discrimination*. https://cran.r-project.org/package=DWDLargeR

Lam, X. Y., Marron, J. S., Sun, D., & Toh, K.-C. (2018b). Fast Algorithms for Large-Scale Generalized Distance Weighted Discrimination. *Journal of Computational and Graphical Statistics*, *27*(2), 368–379. https://doi.org/10.1080/10618600.2017.1366915

Limkin, E. J., Sun, R., Dercle, L., Zacharaki, E. I., Robert, C., Reuzé, S., Schernberg, A., Paragios, N., Deutsch, E., & Ferté, C. (2017). Promises and challenges for the implementation of computational medical imaging (radiomics) in oncology. *Annals of Oncology*, *28*(6), 1191–1206. https://doi.org/10.1093/annonc/mdx034

Marron, J. S., Todd, M. J., & Ahn, J. (2007). Distance-Weighted Discrimination. *Journal of the American Statistical Association*, *102*(480), 1267–1271. https://doi.org/10.1198/016214507000001120

Nelson, A. E., Fang, F., Arbeeva, L., Cleveland, R. J., Schwartz, T. A., Callahan, L. F., Marron, J. S., & Loeser, R. F. (2019). A machine learning approach to knee osteoarthritis phenotyping: data from the FNIH Biomarkers Consortium. *Osteoarthritis and Cartilage*, *27*(7), 994–1001. https://doi.org/https://doi.org/10.1016/j.joca.2018.12.027

Nolen, T. L., Hudgens, M. G., Senb, P. K., & Koch, G. G. (2015). Analysis of repeated low-dose challenge studies. *Statistics in Medicine*, *34*(12), 1981–1992. https://doi.org/10.1002/sim.6462

Pinky, N., Islam, S. M., & Alice, R. (2019). Edibility Detection of Mushroom Using Ensemble Methods. *International Journal of Image, Graphics and Signal Processing*, *11*, 55–62. https://doi.org/10.5815/ijigsp.2019.04.05

Schwarz, G. (1978). Estimating the Dimension of a Model. *Ann. Statist.*, *6*(2), 461–464.

https://doi.org/10.1214/aos/1176344136

Simon, N., Friedman, J. H., Hastie, T., & Tibshirani, R. (2011). Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent. *Journal of Statistical Software; Vol 1, Issue 5 (2011)* . https://doi.org/10.18637/jss.v039.i05

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, *58*(1), 267–288. http://www.jstor.org/stable/2346178

Tomaras, G. D., & Plotkin, S. A. (2017). Complex immune correlates of protection in HIV-1 vaccine efficacy trials. *Immunological Reviews*, *275*(1), 245–261. https://doi.org/10.1111/imr.12514

Vaccari, M., Gordon, S. N., Fourati, S., Schifanella, L., Liyanage, N. P. M., Cameron, M., Keele, B. F., Shen, X., Tomaras, G. D., Billings, E., Rao, M., Chung, A. W., Dowell, K. G., Bailey-Kellogg, C., Brown, E. P., Ackerman, M. E., Vargas-Inchaustegui, D. A., Whitney, S., Doster, M. N., … Franchini, G. (2016). Adjuvant-dependent innate and adaptive immune signatures of risk of SIVmac251 acquisition. *Nature Medicine*, *22*(7), 762–770. https://doi.org/10.1038/nm.4105

Wei, S., Lee, C., Wichers, L., & Marron, J. S. (2016). Direction-Projection-Permutation for High-Dimensional Hypothesis Tests. *Journal of Computational and Graphical Statistics*, *25*(2), 549–569. https://doi.org/10.1080/10618600.2015.1027773

Wibowo, A., Rahayu, Y., Riyanto, A., & Hidayatulloh, T. (2018). Classification algorithm for edible mushroom identification. *2018 International Conference on Information and Communications Technology (ICOIACT)*, 250–253. https://doi.org/10.1109/ICOIACT.2018.8350746

Wu, F., Kirmaier, A., White, E., Ourmanov, I., Whitted, S., Matsuda, K., Riddick, N., Hall, L. R., Morgan, J. S., Plishka, R. J., Buckler-White, A., Johnson, W. E., & Hirsch, V. M. (2016). TRIM5α Resistance Escape Mutations in the Capsid Are Transferable between Simian Immunodeficiency Virus Strains. *Journal of Virology*, *90*(24), 11087 LP – 11095. https://doi.org/10.1128/JVI.01620-16

Wu, Y., Wipf, D., & Yun, J.-M. (2015). Understanding and evaluating sparse linear discriminant analysis. *Artificial Intelligence and Statistics*, 1070–1078.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(2), 301–320. https://doi.org/10.1111/j.1467-9868.2005.00503.x

Zou, H., Hastie, T., & Tibshirani, R. (2006). Sparse Principal Component Analysis. *Journal of Computational and Graphical Statistics*, *15*(2), 265–286. https://doi.org/10.1198/106186006X113430