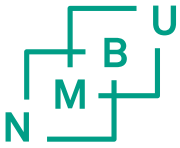

U
B
M
N

Abstract

Variable ranking can be important for the analysis of high-dimensional data. Identifying a subset of relevant variables can be useful both for subsequent model construction and for further investigation of the variables. Many methods for variable ranking and selection exist, but most do not consider interactions between the explanatory variables. In this thesis, three methods for variable ranking and two-way interaction detection in high-dimensions are proposed.

The first method, called Pseudoloadings Ranking (PR), is based on a kernel Partial Least Squares (PLS) model, while the other two are based on the regular PLS algorithm. Interaction Ranking (IR) is an extension of well known filter methods for PLS. Random Interaction Ranking (RIR) ranks the variables by repeatedly selecting and evaluating subsets of variables. The ability of the methods to identify relevant variables was determined by simulation studies, and compared to an existing method. The PR method was unsuccessful in finding important interactions, while he IR and RIR methods had good performances and outperformed the existing method. The use of IR is however limited by memory requirements.

The use of IR and RIR was illustrated by applying them to a gene expression dataset from *Populus tremula*. Some methods for evaluating the findings were proposed.

Sammendrag

Variabelrangering kan være en viktige del av analyse av høydimensjonale data. Identifisering av en liten mengde relevante variable kan være nyttig både for modellbygging og videre utforskning av variablene. Det finnes mange metoder for variabelrangering og -seleksjon, men de fleste tar ikke hensyn til samspill mellom variablene. I denne oppgaven presenteres tre metoder for variabelrangering i situasjoner der det er samspill mellom variablene .

Den første metoden, kalt Pseudoloadings Ranking (PR), er basert på en kernel Partial Least Squares (PLS) modell, mens de to andre er basert på den vanlige PLS algoritmen. Interaction Ranking (IR) er en utvidelse av kjente filtreringsmetoder for PLS. Random Interaction Ranking (RIR) rangerer variablene ved å velge ut og evaluere tilfeldige variabeldelmengder mange ganger. Metodenes evne til å identifisere de relevante variablene ble studert ved simulerings studier. PR metoden lykkes ikke i å finne viktige samspill, mens IR og RIR hadde gode resultater, også sammenlignet med en eksisterende metode. Bruk av IR metoden kan likevel være begrenset av minnebruk.

IR og RIR ble også anvendt på genekspresjonsdata fra *Populus tremula*. Noen metoder for evaluering av resultatene ble foreslått.

Acknowledgements

This thesis is written for the Biostatistics group at the Norwegian University of Life Sciences (NMBU). First, I would like to express my deep gratitude to my supervisor Professor Solve Sæbø for his advice and encouragement. I would also like to thank my co-supervisor Professor Torgeir Hvidsten for his help and for the gene expression data used in this thesis. In addition I wish to thank Associate Professor Trygve Almøy for his contribution to the dataset generation part and for his teaching in my first statistics course, which greatly influenced my later educational choices. I am also grateful for all the teaching I have received at NMBU, and especially to all the staff in the Biostatistics group.

Finally, I wish to thank my parents, my siblings, and my partner, Jonas, for all their support.

Céline Marie Løken Cunen

Oslo, May 2014

Contents

Chapter 1

Notation

In this thesis matrices and vectors are denoted with boldface letters; matrices as upper case letters, like $\mathbf{X}$, and vectors as lower case letters, like $\mathbf{y}$.

The thesis concerns a regression problem with a response variable $\mathbf{y}$ (of dimension $n \times 1$) and a set of explanatory variables $\mathbf{X}$ (of dimension $n \times p$). The relationship between $\mathbf{X}$ and $\mathbf{y}$ is assumed to be linear

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

where $\boldsymbol{\beta}$ is the vector of regression coefficients and $\boldsymbol{\varepsilon}$ is the vector of standard normally distributed error terms.

Chapter 2

Introduction

2.1 Problem

Modern biology generates high-dimensional data, often with many more variables than samples (the $p \gg n$ problem). A typical example is analysis of gene expression with microarrays or RNAseq. These methods measure the expression of thousands of genes, but only in perhaps a hundred different samples. Gene expression is controlled by transcription factors (TFs) and a common problem is identifying which TFs control which genes. In this case, the expression of a gene is the response, while the expression of a large number of putative TFs are the explanatory variables.

High-dimensional data presents challenges for analysis. Traditional methods, like ordinary least squares regression, cannot be applied. When the number of variables is greater than the number of samples, the ordinary least square solution is not unique and the variance of the coefficient estimate is infinite (James et al., 2013). A higher number of variables also increases the risk of overfitting the model, that is choosing a model with poor prediction

ability for future observations. The problem of overfitting is related to the so-called "curse of dimensionality", the principle that adding variables to a model will increase the test error, unless the additional variables are relevant to the response variable (James et al., 2013). Models with many variables are also difficult to interpret.

A common solution to the problems associated with high-dimensional data is variable selection. Variable selection methods reduce the dimensionality of the problem by removing variables that are unrelated to the response. Variable selection counters the problems stated above: it facilitates understanding of the biological system and improves predictive performances (Guyon and Elisseeff, 2003). In this thesis I distinguish between variable selection and variable ranking. Variable selection methods attempt to construct and fit the best possible model according to some goal, often prediction. This involves both the selection of the most relevant explanatory variables and the exclusion of relevant, but redundant, variables (Guyon and Elisseeff, 2003). Variable ranking methods rank all potential variables according to their relevance for prediction of the response, but do not propose a model.

The purpose of ranking the variables can be twofold. Ranking can be a way to reduce the number of variables in order to apply some variable selection methods for construction of a good model. Many variable selection methods perform better on smaller sets of variables (Li et al., 2012). Additionally, ranking the variables can be a way to select a subset of variables for further study. In many applications, obtaining a smaller set of candidate variables, which can be investigated by for example experiments, can be very

useful.

The complexity of the variable selection (and ranking) increases when interactions between the variables are assumed. Interactions means that variables influence the response in combination with other variables. A variable can then be deemed irrelevant for predicting the response when considered separately, but may become very relevant when it is coupled with other variables. In this thesis, I limit myself to two-way interactions, using the following model

$$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \sum_{j \neq k} \beta_{jk} x_{ij} x_{ik} + \epsilon_i \qquad i = 1, ..., n$$

with p explanatory variables and $\frac{p(p-1)}{2}$ two-way interaction terms and where $\epsilon \sim N(0, \sigma)$.

Another approach to high-dimensional data is to reduce the dimensionality by multivariate projection methods. Projection methods are methods where the p variables are projected onto an a-dimensional subspace of lower dimension ($a < p$) (James et al., 2013). This is accomplished by constructing a derived variables, which are linear combinations of the original variables. Partial Least Squares and kernel Partial Least Squares are projection methods that will be presented in the following section.

The methods presented in this thesis attempt to identify a small number of relevant variables among many noise variables. In addition it is assumed that interactions between the explanatory variables are important for the prediction of the response. The methods are based on PLS and kernel PLS and perform a ranking of all the explanatory variables. The methods will be

compared with each other and with an existing method by simulation study and by applying them to a gene expression dataset from *Populus tremula*.

The rest of this chapter provides some useful theory, which will be needed in rest of the thesis. Chapter 3 presents the three variable ranking methods developed for this thesis and gives some information about the *Populus* data and the simulation studies. The results of both simulations and real data analysis, are given in Chapter 4, and are discussed in Chapter 5.

2.2 Projection Methods

Multivariate projection methods are a class of methods suitable for situations where there are more variables than observations or when the explanatory variables are correlated. The key idea is to reduce the dimensionality of the problem by computing a derived variables (where $a < p$, and p is the original number of variables) which are linear combinations, or projections, of the original variables (James et al., 2013).

$$\text{Original variables: } \mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\text{p}], \text{ where } \mathbf{x}_\text{j} \text{ is } n \times 1.$$

$$\text{Derived variables: } \mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_\text{a}], \text{ where } \mathbf{z}_\text{j} \text{ is } n \times 1$$

$$\text{and } \mathbf{z}_\text{j} = \alpha_1 \mathbf{x}_1 + \alpha_2 \mathbf{x}_2 + \ldots + \alpha_p \mathbf{x}_\text{p}, \text{ for some constants } \alpha_1, \alpha_2, \ldots, \alpha_p.$$

In a regression problem, the derived variables can be used as predictor variables instead of the original variables.

Before presenting Partial Least Squares (PLS), I will briefly introduce the most common projection methods, Principal Component Analysis (PCA)

and Principal Component Regression (PCR). The goal of PCA is to find a smaller set of a uncorrelated variables maintaining a maximum of the variation in the $\mathbf{X}$ matrix. These derived variables are called latent variables or components. The components represent the directions of maximum variation in the $\mathbf{X}$ data: the first principal component finds the direction in the space spanned by the columns of $\mathbf{X}$ along which the observations vary the most, the second principal component finds the direction of maximal variation that is orthogonal to the direction of the first component, and so on.

In PCR the components from PCA are used as explanatory variables in a linear regression model. This approach depends on the assumption that the directions of maximal variation in $\mathbf{X}$ are also good predictors of $\mathbf{y}$ (James et al., 2013). This is not always the case and then PLS might be a better solution.

2.2.1 Partial Least Squares

Partial Least Squares (PLS), or projection to latent structures, is a class of methods introduced by Herman Wold (1975). As with PCR, the goal is to find a smaller set of derived variables which can explain the observed response. PLS regression constructs components which retain a maximum of the covariance between the explanatory variables ($\mathbf{X}$) and the response variables ($\mathbf{y}$), instead of just considering the $\mathbf{X}$ matrix as in PCR. In both methods the components are linear combinations of the original variables. In PLS the weights in the linear combinations are proportional to the covariance between $\mathbf{X}$ and $\mathbf{y}$ (Helland, 1988). Least square regression is then applied to $\mathbf{y}$ and the new set of explanatory variables $\mathbf{T}$ (of dimension $n \times a$, with $a < p$). Unlike PCR, PLS regression requires an iterative computation of the

components.

There are several algorithms for PLS regression, see Rosipal and Krämer (2006). The algorithm described below requires a single vector of response observations ($\mathbf{y}$ is $n \times 1$), and is sometimes referred to as PLS1. Assume mean centered matrices $\mathbf{X}_0$ and $\mathbf{y}_0$. For each component $h = 1, 2, ..., a$:

1. Compute loading weights $\mathbf{w}_{\text{h}}$

$$\mathbf{w}_{\text{h}} = \mathbf{X}_{\text{h-1}}^T \mathbf{y}_{\text{h-1}}$$

$$\mathbf{w}_{\text{h}} \leftarrow \mathbf{w}_{\text{h}} / \|\mathbf{w}_{\text{h}}\|$$

   The weights are normalized to length 1. Entry j in the $\mathbf{w}_{\text{h}}$ vector is proportional to the covariance between $\mathbf{y}_{\text{h-1}}$ and column j in $\mathbf{X}_{\text{h-1}}$.

2. Compute the scores $\mathbf{t}_{\text{h}}$

$$\mathbf{t}_{\text{h}} = \mathbf{X}_{\text{h-1}} \mathbf{w}_{\text{h}}$$

   The score vector $\mathbf{t}_{\text{h}}$ has equal dimensions as the columns of $\mathbf{X}$ and is a linear combination of the columns of $\mathbf{X}_{\text{h-1}}$ according to the weights given by $\mathbf{w}_{\text{h}}$. The $\mathbf{t}_{\text{h}}$ vectors are the extracted components, *i.e.* the new explanatory variables.

3. Compute the loading vectors $\mathbf{p}_{\text{h}}$ and $\mathbf{q}_{\text{h}}$

$$\mathbf{p}_{\text{h}} = \mathbf{X}_{\text{h-1}}^T \frac{\mathbf{t}_{\text{h}}}{\mathbf{t}_{\text{h}}^T \mathbf{t}_{\text{h}}}$$

$$\mathbf{q}_{\text{h}} = \mathbf{y}_{\text{h-1}}^T \frac{\mathbf{t}_{\text{h}}}{\mathbf{t}_{\text{h}}^T \mathbf{t}_{\text{h}}}$$

   $\mathbf{p}_{\text{h}}$ is a $p \times 1$ vector, which is computed by regressing the variables in $\mathbf{X}_{\text{h-1}}$ on the score vector $\mathbf{t}_{\text{h}}$.

4. Deflate $\mathbf{X}_{\text{h-1}}$ and $\mathbf{y}_{\text{h-1}}$

$$\mathbf{X}_{\text{h}} = \mathbf{X}_{\text{h-1}} - \mathbf{t}_{\text{h}}\mathbf{p}_{\text{h}}^{T}$$

$$\mathbf{y}_{\text{h}} = \mathbf{y}_{\text{h-1}} - \mathbf{t}_{\text{h}}\mathbf{q}_{\text{h}}$$

   Before computing the next component, the contribution of the current component must be removed from the $\mathbf{X}$ matrix, so that the next component will explain parts of $\mathbf{X}$ that were not explained by the current component. Deflation of $\mathbf{y}$ is not actually necessary when $\mathbf{y}$ is $n \times 1$.

5. If more components are needed, return to 1.

In each iteration of the algorithm, the loading weights, score vectors and loading vectors are saved in the matrices $\mathbf{W}_a$, $\mathbf{T}_a$, $\mathbf{P}_a$, $\mathbf{Q}_a$.

$$\mathbf{W}_a = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_a]$$

$$\mathbf{T}_a = [\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_a]$$

$$\mathbf{P}_a = [\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_a]$$

$$\mathbf{Q}_a = [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_a]$$

These matrices are then used to compute the vector of estimated regression coefficients $\hat{\boldsymbol{\beta}}$ and the fitted values $\hat{\mathbf{y}}$

$$\hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \\ \vdots \\ \hat{\beta}_p \end{bmatrix} = \mathbf{W}_a(\mathbf{P}_a^T\mathbf{W}_a)^{-1}\mathbf{Q}_a^T$$

$$\hat{\mathbf{y}} = \bar{y} + \mathbf{X}_0\hat{\boldsymbol{\beta}}$$

where $\bar{y}$ is the mean of the response vector $\mathbf{y}$, and where $\mathbf{X}_0$ is the mean centered matrix of explanatory variables that was used in the algorithm. The predicted response for an independent set of explanatory variables is given by

$$\hat{\mathbf{y}} = \bar{y} + \mathbf{X}_{\text{test}}\hat{\boldsymbol{\beta}}$$

where $\mathbf{X}_{\text{test}}$ is the matrix of independent observations of the explanatory variables, with columns centered by the column means of $\mathbf{X}$.

2.2.2 Kernel Partial Least Squares

Kernel PLS is a variant of PLS which was first designed to speed up computations when applying PLS to large matrices (Rännar et al., 1994; Lindgren et al., 1993). This is achieved by combining two steps in the algorithm

$$\left.\begin{array}{l}\text{Step 1} : \mathbf{w} = \mathbf{X}^T\mathbf{y} \\ \\ \text{Step 2} : \mathbf{t} = \mathbf{X}\mathbf{w}\end{array}\right\} \quad \rightarrow \mathbf{t} = \mathbf{X}\mathbf{X}^T\mathbf{y} = \mathbf{K}\mathbf{y} \tag{2.1}$$

and allowing the deflation of $\mathbf{K}$ rather than $\mathbf{X}$. The resulting $\mathbf{K}$ matrix has dimensions $n \times n$, and is thus a small matrix if there are few observations.

Rosipal and Trejo (2001) introduced kernel-based PLS methods for non-linear models. Their idea is to use a non-linear function $\Phi()$ to map the data into a suitable high-dimensional space where ordinary linear PLS can be applied (Rosipal and Krämer, 2006).

$\Phi()$ maps $\mathcal{X}$-space data into space $\mathcal{F}$:

$$\Phi : \mathcal{X} \rightarrow \mathcal{F}$$

PLS is then applied to $\boldsymbol{\Phi}$, the matrix of $\mathcal{F}$-space data. According to (2.1), we now need to calculate the Gram matrix $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$. In order to retain

fast computations the Kernel trick is applied. The trick allows us to avoid the matrix multiplication $\mathbf{\Phi}\mathbf{\Phi}^T$, by computing the elements i, j of $\mathbf{K}$ directly, with a kernel function $k()$. Several kernel functions can be used, depending on which feature space $\mathcal{F}$ we want to map the data onto. For example

$$\text{Gaussian kernels: } K(\mathbf{x'}_i, \mathbf{x'}_j) = e^{-(\frac{\|\mathbf{x'}_i - \mathbf{x'}_j\|^2}{d})} \tag{2.2}$$

$$\text{Polynomial kernels: } K(\mathbf{x'}_i, \mathbf{x'}_j) = (\mathbf{x'}_i \mathbf{x'}_j^T + 1)^a \tag{2.3}$$

where $\mathbf{x'_i}, \mathbf{x'_j}$ are rows i and j of mean centered $\mathbf{X}$ and d and a are positive constants.

Example 1. *Assume a* 2×1 *response vector* $\mathbf{y}$ *and a matrix* $\mathbf{X}$ *with two explanatory variables*

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2] = \begin{bmatrix} x_{1,1} & x_{2,1} \\ x_{1,2} & x_{2,2} \end{bmatrix}$$

The response is assumed to be related to the two-way interactions between the variables in $\mathbf{X}$*. In order to include two-way interaction terms in the regression model, the* $\mathbf{X}$ *matrix can be mapped to a suitable higher-dimensional space* $\mathcal{F}$ *by a function* $\Phi()$*. Then one gets the matrix of mapped data*

$$\mathbf{\Phi} = [\mathbf{1}, \sqrt{2}\mathbf{x}_1, \sqrt{2}\mathbf{x}_2, \sqrt{2}\mathbf{x}_1\mathbf{x}_2, \mathbf{x}_1{}^2, \mathbf{x}_2{}^2]$$

with six variables instead of two. If the normal PLS-algorithm is applied to this matrix, the score vectors will be linear combinations of both the two original variables and the interaction term (and the squared terms).

If $\mathbf{X}$ *is large,* $\mathbf{\Phi}$ *will be extremely large.. The Kernel trick allows us to*

avoid computing $\mathbf{\Phi}\mathbf{\Phi}^T$ *by directly calculating K instead*

$$\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T = \begin{bmatrix} (1 + x_{1,1}^2 + x_{1,2}^2)^2 & (1 + x_{1,1}x_{1,2} + x_{2,1}x_{2,2})^2 \\ (1 + x_{1,1}x_{1,2} + x_{2,1}x_{2,2})^2 & (1 + x_{1,2}^2 + x_{2,2}^2)^2 \end{bmatrix}$$

Here the polynomial kernel function of second order is used ($(\mathbf{x'}_1\mathbf{x'}_2^T + 1)^2$ *where* $\mathbf{x'}_1$ *and* $\mathbf{x'}_2$ *are the row vectors of* $\mathbf{X}$*). It can be confirmed that using this kernel function yields the same* $\mathbf{K}$ *matrix as* $\mathbf{\Phi}\mathbf{\Phi}^T$ *would.*

In this thesis I use polynomial kernels (mostly of second order: $a = 2$) because I am interested in interactions between variables (see example 1). The following algorithm, requiring a single vector of response observations ($\mathbf{y}$ is $n \times 1$), was used. Assume mean centered matrices $\mathbf{X}_0$ and $\mathbf{y}_0$, and a matrix $\mathbf{K}_0$ computed according to (2.3). For each component $h = 1, 2, ..., a$:

1. Compute $\mathbf{X}$-scores $\mathbf{t}_\text{h}$

$$\mathbf{t}_\text{h} = \mathbf{K}_\text{h-1}\mathbf{y}_\text{h-1}$$

$$\mathbf{t}_\text{h} \leftarrow \mathbf{t}_\text{h}/\|\mathbf{t}_\text{h}\|$$

   The scores are normalized to length 1. The score vector $\mathbf{t}_\text{h}$ has equal dimensions as the columns of $\mathbf{X}$ and is a linear combination of the columns of $\mathbf{\Phi}_\text{h-1}$ according to the weights given by $\mathbf{\Phi}_\text{h-1}^T\mathbf{y}$. The $\mathbf{t}_\text{h}$-vectors are the extracted components, *i.e.* the new explanatory variables.

2. Compute weight vectors $\mathbf{c}_\text{h}$

$$\mathbf{c}_\text{h} = \mathbf{y}_\text{h-1}^T\mathbf{t}_\text{h}$$

   When $\mathbf{y}$ is $n \times 1$, $\mathbf{c}_\text{h}$ is a scalar. It is proportional to the covariance between $\mathbf{y}$ and $\mathbf{t}$.

3. Compute $\mathbf{y}$-scores $\mathbf{u}_{\mathrm{h}}$

$$\mathbf{u}_{\mathrm{h}} = \mathbf{y}_{\mathrm{h\text{-}1}}\mathbf{c}_{\mathrm{h}}$$

$$\mathbf{u}_{\mathrm{h}} \leftarrow \mathbf{u}_{\mathrm{h}}/\|\mathbf{u}_{\mathrm{h}}\|$$

4. Deflate $\mathbf{K}_{\mathrm{h\text{-}1}}$ and $\mathbf{y}_{\mathrm{h\text{-}1}}$

$$\mathbf{K}_{\mathrm{h}} = (I - \mathbf{t}_{\mathrm{h}}\mathbf{t}_{\mathrm{h}}^T)\mathbf{K}_{\mathrm{h\text{-}1}}(I - \mathbf{t}_{\mathrm{h}}\mathbf{t}_{\mathrm{h}}^T)$$

$$\mathbf{y}_{\mathrm{h}} = \mathbf{y}_{\mathrm{h\text{-}1}} - \mathbf{t}_{\mathrm{h}}\mathbf{t}_{\mathrm{h}}^T\mathbf{y}_{\mathrm{h\text{-}1}}$$

   where $\mathbf{I}$ is the n-dimensional identity matrix.

5. If more components are needed, return to 1.

In each iteration of the algorithm, the score vectors and weight vectors are saved in the matrices $\mathbf{T}$, $\mathbf{U}$, $\mathbf{C}$.

$$\mathbf{T}_a = [\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_a]$$

$$\mathbf{U}_a = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_a]$$

$$\mathbf{C}_a = [\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_a]$$

These matrices are then used to compute the fitted values $\hat{\mathbf{y}}$ (the $\hat{\boldsymbol{\beta}}$-vector is not computed, since it belongs in the $\mathcal{F}$-space)

$$\hat{\mathbf{y}} = \bar{y} + \mathbf{K}_0\mathbf{U}_a(\mathbf{T}_a^T\mathbf{K}_0\mathbf{U}_a)^{-1}\mathbf{T}_a^T\mathbf{y}$$

where $\bar{y}$ is the mean of the response vector $\mathbf{y}$ and $\mathbf{K}_0$ is computed with the mean centered matrix of explanatory variables that was used in the algorithm. The predicted response for an independent set of explanatory variables is given by

$$\hat{\mathbf{y}} = \bar{y} + \mathbf{K}_{\mathrm{test}}\mathbf{U}_a(\mathbf{T}_a^T\mathbf{K}_0\mathbf{U}_a)^{-1}\mathbf{T}_a^T\mathbf{y}$$

where K_{test} is computed with a mean centered matrix of independent observations of the explanatory variables.

Kernel PLS allows us to construct non-linear models, but outputs less information than ordinary PLS. We only get the scores $\mathbf{t}$, but not the loading weights $\mathbf{w}$ or the loading vectors $\mathbf{p}$. In other words, we obtain the new explanatory variables $\mathbf{t}$, but there is no simple way of finding out how these variables were constructed, *i.e.* with what non-linear combinations of the explanatory variables.

2.3 Variable Selection with PLS

As stated in Section 2.1, variable selection methods reduce dimensionality by selecting only the variables that are the most relevant to the response. With that in mind, variable selection in combination with PLS regression might seem unnecessary, as the loading weights from PLS regression already reflect the importance of the variables in explaining the response. In principle, the PLS regression should find the directions in the variable space spanned by the relevant variables, and avoid the directions spanned by noise variables. However, PLS methods are not variable selection methods, the components from the PLS regression are linear combinations of *all* the original variables. The resulting models always include all the original variables, and not a small subset of relevant variables. This is a problem if one wishes to obtain a model obeying the sparsity principle, the assumption that only a small subset of all the variables measured are generating the observed response (Chun and Keleş, 2010). Thus, one might want to combine variable selection methods with PLS, to obtain sparse, biologically interpretable models.

Another motivation for variable selection with PLS, is that results from PLS can be poor in situations with very large p and small n (Mehmood et al., 2012). PLS estimators are not asymptotically consistent in situations where p is larger than n and p grows faster than n (Chun and Keleş, 2010). A consistent estimator gives estimates converging to the true parameter when the number of observations n increases indefinitely (Miller et al., 2004). When there are large numbers of irrelevant variables, the directions in the PLS model are affected by the noise variables (Chun and Keleş, 2010). Variable selection can then improve the performance of the PLS model.

Methods for variable selection are often separated into three main categories: filter-, wrapper- and embedded methods (Mehmood et al., 2012). In a PLS setting, filter methods rank the variables based on the output from the PLS regression algorithm. Wrapper methods assess subsets of variables according to their predictive performances (Guyon and Elisseeff, 2003) and often alternate between model fitting and variable selection (Mehmood et al., 2012). Embedded methods modify the PLS regression algorithm such that variable selection is performed automatically. In this thesis, the methods discussed fall into the filter- and wrapper categories, and embedded methods will not be discussed any further.

2.3.1 Filter methods

Filter methods are fast and easy to compute. The variables are ranked according to some measure from the output of the PLS algorithm, often the loading weight vectors $\mathbf{w}$ or the regression coefficients $\hat{\boldsymbol{\beta}}$. For a given number of components, the variables with loading weight or $\hat{\beta}$ larger than some threshold in absolute value are selected (Mehmood et al., 2012).

Keep variable $\mathbf{x_j}$ from $\mathbf{X}$ $(n \times p)$ if:

$$|\hat{\beta}_j| > threshold$$

A common variant of this very simple filter method, is to compute T-statistics for the $\hat{\beta}$s based on jackknife variance estimates.

Jackknifing

Jackknifing is a resampling method which can be used for variable selection with PLS (Karaman et al., 2013). Let the vector of regression coefficients computed with all the n observations be called $\hat{\boldsymbol{\beta}}$. The variance of the estimated regression coefficient is estimated by computing the $\hat{\boldsymbol{\beta}}_i$ vector for each of n different jackknife samples $(i = 1, 2, ..., n)$. The ith jackknife sample contains all the observations except the ith observation, which is removed (see Section 2.4.2 on cross-validation).

Let: $\hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \\ \vdots \\ \hat{\beta}_j \\ \vdots \\ \hat{\beta}_p \end{bmatrix}$, from each of the n jackknife samples we get: $\hat{\boldsymbol{\beta}}_i = \begin{bmatrix} \hat{\beta}_{i,1} \\ \hat{\beta}_{i,2} \\ \vdots \\ \hat{\beta}_{i,j} \\ \vdots \\ \hat{\beta}_{i,p} \end{bmatrix}$

The jackknife estimate of the variance of $\hat{\beta}_j$ (Tukey, 1958):

$$\widehat{var(\hat{\beta}_j)} = s(\hat{\beta}_j)^2 = \frac{n-1}{n} \sum_{i=1}^{n} (\hat{\beta}_{i,j} - \bar{\beta}_{.j})^2$$

Where $\bar{\beta}_{.j}$ is the mean of the n $\hat{\beta}_{i,j}$. A T-statistic for each $\hat{\beta}_j$ can then be computed as:

$$T_j = \frac{\bar{\beta}_{.j}}{s(\hat{\beta}_j)}$$

The T-statistics are used for selecting the most relevant variables.

Keep variable $\mathbf{x_j}$ from $\mathbf{X}$ $(n \times p)$ if:

$$|T_j| > threshold$$

The advantage of jackknifing compared to simply selecting the variable with the largest regression coefficients, is that it takes into account the uncertainty of the regression coefficient estimates (Karaman et al., 2013).

The performance of filter methods is in general dependent on the threshold, and choosing a good threshold can be difficult, see Mehmood et al. (2012) for some suggestions.

2.3.2 Wrapper methods

The main difference between filter- and wrapper methods is that in wrapper methods the PLS model is fitted several times, for different subset of variables. Wrapper methods can rank the variables based on simple measures as in the filter methods or based on predictive performance (see Section 2.4 for how to evaluate predictive performance). The ideal wrapper method (with PLS) would be to fit a PLS model for all possible subsets of the p variables. This means 2^p different models and this number grows exponentially with the number of variables. This approach guarantees to find the best possible subset of the variables,but it is not practicable for large numbers of variables.

Since evaluating all possible subsets is generally infeasible, wrapper methods use different search algorithms that only explore a subspace of the total search space. The search algorithms can either be deterministic or randomized, and this defines two categories of wrapper methods (Mehmood et al., 2012). Randomized search algorithms use some level of randomness in selecting the variables in the subsets (Mehmood et al., 2012) and therefore generally produce different solutions for each run. Deterministic methods produce the same solution for every run. A typical deterministic wrapper methods is Backward variable elimination PLS. The Genetic algorithm is an example of a randomized wrapper method.

Methods presented here and in Mehmood et al. (2012) generally do not consider interactions between the variables. This motivates the development of methods for variable selection with PLS that can detect variables which influence the response through interactions.

2.4 Model Evaluation

Many variable selection methods depend on comparing models constructed with different subsets of variables. Therefore it is necessary to have ways to evaluate the performance of the models. A simple measure of the training error is often used to evaluate models in introductory courses in statistics.

Training mean squared error (training MSE):

$$MSE_{train} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where $\hat{y}_i$ are the fitted values. Training MSE is a measure of how well the training data fits the model, but can be a bad measure of model performance. A complicated model can fit the training data perfectly, but it is more interesting how well the model predicts new observations (James et al., 2013). Instead of measuring the training error, we need to estimate the test error, the error of prediction when the model is tested on independent validation data.

Test mean squared error (test MSE):

$$MSE_{test} = \frac{1}{n}\sum_{i=1}^{n}(y_{\text{val,i}} - \hat{y}_{\text{val,i}})^2$$

Where $y_{\text{val,i}}$ are the true response values in the validation dataset, and $\hat{y}_{\text{val,i}}$ are the response values estimated by the model. Test MSE can be used directly to compare the performance of different models: a small test MSE means that the model accurately predicts the observations in the validation dataset.

Two other performance measures, related to test MSE, will be used in this thesis. The $R^2_{prediction}$ statistic is a proportion and hence should take

values between 0 and 1. It compares the test MSE of the actual model with the test MSE of the null model. The null model is a model containing no explanatory variables, which means that all response values in the validation data are predicted by the mean of the response values in the training data. If the actual model has lower test MSE than the null model, $R^2_{prediction}$ will be close to 1. If otherwise $R^2_{prediction}$ will be close to zero or even negative. Negative values occur when the actual model has larger test MSE than the null model, in practise negative values are usually set equal to zero.

$$R^2_{\text{prediction}} = 1 - \frac{\sum_{i=1}^{n-m}(y_{\text{val,i}} - \hat{y}_{\text{val,i}})^2}{\sum_{i=1}^{n-m}(y_{\text{val,i}} - \bar{y}_{\text{train}})^2}$$

The second performance measure is the Root mean square error of prediction (RMSEP).

$$RMSEP = \sqrt{\frac{\sum_{i=1}^{n-m}(y_{\text{val,i}} - \hat{y}_{\text{val,i}})^2}{n}}$$

2.4.1 Validation set

The most intuitive way to estimate the test error is to randomly divide the available data into a training set and a validation set. The model is fitted with the training set and the responses of the observations in the validation set are predicted using the fitted model. The performance measures are evaluated with these predicted values and the responses in the validation set. The model with largest $R^2_{prediction}$ or smallest RMSEP can be selected. A problem with this approach is that the estimated test error can vary a lot depending on which observations were included in the training or validation sets (James et al., 2013). Another problem is that only a subset of the observations are used for fitting the model, and then information might be lost. An alternative method is cross-validation, it can be used for estimating test MSE with training data only (Stone, 1974).

2.4.2 Leave-one-out cross-validation

To get a good estimate of the test MSE without applying the model on a validation set, one can use leave-one-out cross-validation (LOOCV). The model is fitted n times, in each iteration one of the observations is removed from the data used in the model fitting. The model is thus fitted with the remaining $n - 1$ observations. The response value of the removed observation is then predicted by the fitted model and an estimate of the prediction error for that observation is obtained. This is repeated for every observation, and the LOOCV estimate of the test MSE is the mean of these squared prediction errors (James et al., 2013).

LOOCV estimate of test MSE:

$$MSE_{test} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_{(i)})^2$$

Where $\hat{y}_{(i)}$ is the predicted value of y_i by a model fitted with all observations except observation i.

2.5 Logistic regression

The variable ranking methods presented in this thesis were studied by simulations studies (see Section 3.3). In order to determine which factors influence the performance of the methods, the simulation results were analysed with a logistic regression model.

Logistic regression is used when the response is binary, for example a variable Z, with two possible outcomes: $Z = 0$ ("failure") or $Z = 1$ ("success") (Agresti, 2007). Let π be the probability of success:

$$\pi = P(Z = 1)$$

In logistic regression one models the log-odds of success. A model with explanatory variables as factors (categorical variables) might look like

$$\log(\frac{\pi}{1-\pi}) = \alpha + \beta_i^X + \beta_j^W + \beta_{ij}^{XW}$$

The parameter β_i^X represents the effect of category i of factor X on the log-odds. The parameter β_{ij}^{XW} represents the interaction between the factors X and W. When the parameters in the model are estimated, it is common to choose one category as a reference level and only estimate the parameters belonging to the other categories of the factor. For example if factor X has

two categories, $i = 1, 2$, the β_1^X can be set to 0 and β_2^X then represents the difference in log-odds between category 2 and 1. If β_2^X is positive, it means that the probability π of success is larger with category 2 of factor X than with category 1.

The importance of the terms in the model can be evaluated by likelihood-ratio tests. For example the likelihood-ratio test statistic for the factor X equals

$$LR = -2\log(\frac{l_0}{l_1})$$

where l_0 is the maximized value of the likelihood function under the null hypothesis ($\beta_2^X = 0$) and l_1 is the maximized value of the likelihood function when the parameter does not need to be equal to 0 (Agresti, 2007). This test statistic has a large sample chi-squared distribution under the null hypothesis. If it is large enough, the null hypothesis is rejected and the factor X is considered important for the response.

2.6 Sliced Inverse Regression for Interaction Detection (SIRI)

Sliced Inverse Regression for Interaction Detection (SIRI) is an existing procedure for variable ranking and selection when interactions between the explanatory variables are assumed. It was introduced by Jiang and Liu (2013) and in that paper the authors investigate the properties of SIRI both as a variable ranking method and as a variable selection method. In the context of my thesis, the variable ranking part of SIRI is the most relevant. The authors compare SIRI to other variable ranking methods (ISIS and DC-SIS) by simulation studies. SIRI outperforms the other methods in the scenario

where the response is controlled by an interaction term (Jiang and Liu, 2013). The performance of SIRI was compared to the performance of the methods introduced in this thesis (see Section 4.5).

SIRI is an iterative procedure alternating between a variant of Sure independence Screening (SIS) and stepwise selection steps based on likelihood-ratio tests (Jiang and Liu, 2013). These tests are based on inverse models, *i.e.* where one models the conditional distribution of the explanatory variables given the response (instead of the modelling the conditional distribution of the response given $\mathbf{X}$).

SIS is a method proposed by Fan and Lv (2008). The goal is to rank the explanatory variables and reduce the number of variables (p) from very large to below sample size (n) in a fast and efficient way (Fan and Lv, 2008). After this filtering step, well-studied variable selection methods can be used to construct a good model. The SIS method ranks the variables according to their correlation with the response, and keeps the varibles with a correlation higher than some threshold. In the paper the authors prove that the SIS methods has the sure screening property, *i.e.* the probability that all important variables are selected tends to 1 as n increases (Fan and Lv, 2008). However this property does not hold for variables that are individually uncorrelated with the response, but take part in important interactions. An iterative extension of SIS called ISIS is, according to the authors, able to handle such cases.

Instead of using the correlation, the SIS procedure in SIRI uses a different test statistic for ranking the explanatory variables. The test statistic is

based on the conditional distribution of the explanatory variables given the response within different *slices*. Slices are disjoint groups of sorted response observations. The slices are supposed to contain approximately the same number of observations. The variables are scored by evaluating the conditional variance of each variable within the different slices, variables with either different means or different variance across slices are supposed to get high scores (Jiang and Liu, 2013). This scoring method is able to identify variables which are part of important interaction, even though they may appear irrelevant to the response when considered separately. The authors propose to use the variable screening to reduce the number of variables from p to $n/log(n)$.

Chapter 3

Materials and Methods

This chapter first presents the three methods that were developed for this thesis. Then the datasets which were used to study the performance of the methods are described, both a real dataset and the method for dataset generation. The last section describes the protocol for the simulation studies.

3.1 New methods

The objective of the three methods described in this section is to identify the (few) important variables in a large set of noise variables. The important variables are assumed to be relevant predictors of the response in combinations with other variables. Thus we want to find two-way interactions among variables that are important for the prediction of the response. All three methods produce a ranking of all the explanatory variables in the dataset.

3.1.1 Pseudoloadings Ranking (PR)

The PR method is mainly intended to be a method for an initial reduction of the number of explanatory variables. The method may be considered as

a filter method, as described in 2.3. A fast and efficient method capable of an initial reduction of the number of variables would be very useful in combination with other (slower) methods, like IR and RIR.

1. Fit a polynomial kernel PLS model (of 2. order) with $\mathbf{X}$ and $\mathbf{y}$.

2. Compute the *pseudoloadings*. Kernel PLS do not provide loading vectors like ordinary PLS, but we define pseudoloading vector as

$$\mathbf{p}_{\mathrm{h}}^{*} = \mathbf{X}_{0}^{\mathbf{T}}\mathbf{t}_{\mathrm{h}}$$

   where $h = 1, 2, \ldots, a$ and a is the number of components chosen by the user. $\mathbf{X}_0$ is the mean centered $\mathbf{X}$ matrix and $\mathbf{t}_{\mathrm{h}}$ is the h-th score vector from the kernel PLS model.

3. Compute distances. Let $\mathbf{P}$ be a $p \times a$ matrix with the $\mathbf{p}_{\mathrm{h}}^{*}$ vectors as columns. The pseudoloadings can be used to compute an euclidean distance measure d for each variable.

$$\text{Distance for variable } j,\ j = 1, 2, ...p\text{:} \quad d_j = \sqrt{p_1^* + p_2^* + ... + p_a^*}$$

   where $p_h^*, h = 1, 2..., a$ are the a pseudoloading values corresponding to variable j.

4. Filter. Filtration can be performed by discarding the variables with the smallest distance, and we end up with a $m \times r$ matrix $\mathbf{X}_{\text{reduced}}$ with r smaller than p.

The idea behind this method is that variables with second order interactions which are important for the prediction of y will presumably get large pseudoloadings and therefore large distance values. Figure 3.1 displays the pseudoloadings and the corresponding distance measures when $a = 2$.

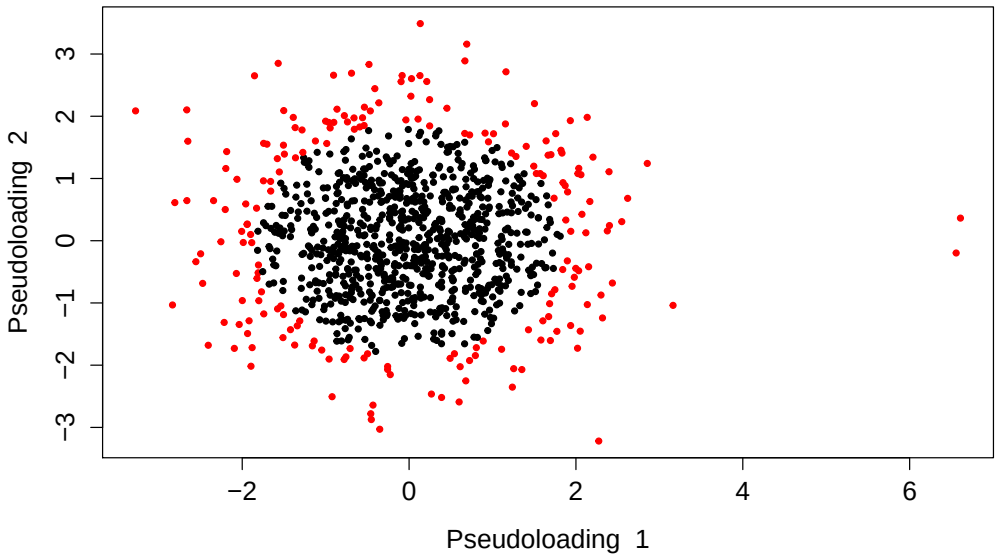


Figure 3.1: Pseudoloading values for 1000 variables, computed with the first two components. Distances are calculated with these two pseudoloadings and the variables corresponding to the 20% largest distances are marked in red. The relevant variables for this example are the two furthest points along the x-axis.

There is only one argument that can be varied in the PR method: the number of components in the kernel PLS model. This argument will be a factor in the simulation studies (Section 3.3).

3.1.2 Interaction Ranking (IR)

The IR method is a somewhat slower filter method than PR. It can be considered an extension of the common filtering methods for PLS described in Section 2.3.1.

1. Compute the matrix $\mathbf{Q}$ of all two-way interactions between the p ex-

planatory variables in $\mathbf{X}$. This matrix has $\frac{p(p-1)}{2}$ columns.

If: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ...\mathbf{x}_\mathrm{p}]$

Then: $\mathbf{Q} = [\mathbf{x}_1 * \mathbf{x}_2, \mathbf{x}_1 * \mathbf{x}_3, \ldots, \mathbf{x}_1 * \mathbf{x}_\mathrm{p}, \mathbf{x}_2 * \mathbf{x}_3, \ldots, \mathbf{x}_\mathrm{p} * \mathbf{x}_\mathrm{p}]$

Where $*$ is the element-wise multiplication of the vectors.

2. Combine $\mathbf{X}$ and $\mathbf{Q}$. This large matrix, $\mathbf{Q}_{\text{extended}}$, has $p + \frac{p(p-1)}{2}$ columns.

$$\mathbf{Q}_{\text{extended}} = [\mathbf{X}, \mathbf{Q}]$$

3. Fit a (leave-one-out cross-validated) PLS model with $\mathbf{Q}_{\text{extended}}$ and $\mathbf{y}$ and compute a score for all the variables and the interactions. This score is either based on the $\hat{\boldsymbol{\beta}}$-vector or on the T-statistics from jackknifing (see Section 2.3.1).

4. The variables or interaction terms with the largest scores (in absolute value) are assumed to be the most relevant predictors of $\mathbf{y}$.

There are two arguments that potentially can affect the performance of IR: the number of components in the PLS model and the scoring method (based on the $\hat{\boldsymbol{\beta}}$-vector or on the T-statistics from jackknifing). These will be investigated with the simulation studies.

3.1.3 Random Interaction Ranking (RIR)

The RIR method is related to randomized wrapper methods (see 2.3.2). The general idea is to repeatedly evaluate random subsets of the explanatory variables. The variables are scored according to both the predictive performance of the subset and the importance of each variable (with interactions).

Unlike the PR and IR methods, the RIR method requires the evaluation of predictive performances, $R^2_{\text{prediction}}$, for the ranking of the variables. In order to calculate $R^2_{\text{prediction}}$ the available data has to be divided into a training set and a validation set. This division can be done in several ways, for example by sampling new sets in every iteration. An alternative is to have the same training and validation sets for all iterations, and that alternative is presented here. One then gets the matrices $\mathbf{X}_{\text{train}}$ $(m \times p)$ and $\mathbf{X}_{\text{val}}$ $((n-m)\times p)$, and the vectors $\mathbf{y}_{\text{train}}$ $(m\times 1)$ and $\mathbf{y}_{\text{val}}$ $((n-m)\times 1)$, where $m < n$.

In each iteration choose a random set of k explanatory variables, with $k << p$:

1. We get the matrices $\mathbf{X}_{\text{k,train}}$ $(m \times k)$ and $\mathbf{X}_{\text{k,val}}$ $((n-m) \times k)$. For example:

$$\mathbf{X}_{\text{k,train}} = [\mathbf{x}_3, \mathbf{x}_5, \mathbf{x}_{11}, ...\mathbf{x}_{\text{k}}]$$

2. Compute the matrix $\mathbf{Q}_t$ of all two-way interactions between the k variables, this matrix has m rows and $\frac{k(k-1)}{2}$ columns.

$$\mathbf{Q}_t = [\mathbf{x}_3 * \mathbf{x}_5, \mathbf{x}_3 * \mathbf{x}_{11}, \ldots, \mathbf{x}_3 * \mathbf{x}_{\text{k}}, \mathbf{x}_5 * \mathbf{x}_{11}, \ldots, \mathbf{x}_{\text{k}} * \mathbf{x}_{\text{k}}]$$

   Where $*$ is the element-wise multiplication of the vectors. Also compute a similar matrix $\mathbf{Q}_v$ $((n-m) \times \frac{k(k-1)}{2})$ with the validation set.

3. Combine $\mathbf{X}_{\text{k,train}}$ and $\mathbf{Q}_t$. This matrix, $\mathbf{Q}_{\text{train}}$, has $k + \frac{k(k-1)}{2}$ columns.

$$\mathbf{Q}_{\text{train}} = [\mathbf{X}_{\text{train}}, \mathbf{Q}_t]$$

   Combine $\mathbf{X}_{\text{k,val}}$ and $\mathbf{Q}_v$ in a similar fashion to obtain $\mathbf{Q}_{\text{val}}$.

4. Fit a PLS model with $\mathbf{Q}_{\text{train}}$ and $\mathbf{y}_{\text{train}}$, evaluate the model on $\mathbf{Q}_{\text{val}}$ and $\mathbf{y}_{\text{val}}$ and compute $R^2_{\text{prediction}}$ for each number of components up to a chosen number a.

5. Choose the number of components with maximum $R^2_{\text{prediction}}$, let the maximal R^2 be called R^2_{max}.

6. Either perform a leave-one-out (LOO) cross-validation and compute the jackknife T-statistics or use the $\hat{\beta}$ from the PLS model as an importance score S. One gets $k + \frac{k(k-1)}{2}$ importance scores, one for each of the variables and interaction terms in $\mathbf{Q}_{train}$.

7. Score each of the k variables in the set with a score which is combination of the general performance of the PLS model in step 4 and a score for each variable: the largest S among all the interactions (or the main effect) a variable is involved in. The score for variable j ($j = 1, 2, \ldots, k$) is equal to

$$Score_j = S^2_{\text{max}} \times R^2_{\text{max}}$$

$$S_{\text{max}} = max(|S_1|, \ldots, |S_k|)$$

where $|S_1|, \ldots, |S_k|$ are the absolute values of the scores computed in step 6 corresponding to the variable j and to all the interactions which variable j is involved in.

In each iteration the scores for the variable subset are added to the scores from previous iterations. The procedure is repeated a large number of times, until all the variable have been scored several times. The variables with the highest scores are assumed to be the most relevant for predicting the response (or involved in relevant interactions).

There are several arguments that potentially can affect the performance of RIR: the number of variables in each iteration (v), the number of iterations (g), the scoring method (using $\hat{\beta}$s or T-statistics from jackknifing) and the

filter threshold (see below). These will be investigated with the simulation studies.

Filtering step in RIR

If the number of relevant explanatory variables is small compared to the total number of variables, most of the iterations in RIR will concern subsets of variables which do not contain any relevant variables. A filtering step can be added to the method in order to avoid using time to score subset with low (or even negative) $R^2_{\text{prediction}}$:

1. Before starting to run the iterations, the $R^2_{\text{prediction}}$ belonging to *all* the variables is computed. $R^2_{\text{prediction, all}}$ is calculated by fitting a polynomial kernel PLS model of 2. order with $\mathbf{X}_{\text{train}}$ and $\mathbf{y}_{\text{train}}$, and testing it on $\mathbf{X}_{\text{val}}$ and $\mathbf{y}_{\text{val}}$.

2. Prior to the scoring step in each iteration (before step 6 above), the maximal $R^2_{\text{prediction}}$ belonging to the subset is compared to $R^2_{\text{prediction, all}}$.

$$\text{If: } R^2_{\text{max}} < (R^2_{\text{prediction, all}} - threshold) \tag{3.1}$$

   Then the belonging subset of variables is not scored, and the algorithm jumps directly to the next iteration.

The *threshold* can be chosen by the user. For the simulation the threshold is constant and equal to 0.1.

3.2 Data

3.2.1 Gene expression data

The methods described above were applied to a gene expression dataset from Swedish aspen (*Populus tremula*). The data were collected by scientists from Umeå Plant Science Centre (UPSC) (Torgeir Hvidsten, personal communication).

The data consist of 130 samples from 5 tree clones. The trees were about 15 meters tall and 47 years when they were cut in July 2010 in Sweden. The 26 samples from each tree were taken at different stages of wood development, from the vascular cambium to cell death. In addition a few samples were taken from the phloem.

Figure 3.2 above displays a cross-section of a tree stem. The phloem is the food-conducting tissue of plants, it conducts sugars and other nutrients and constitutes the outer part of the tree stem (Raven et al., 2005). The xylem is the water-conducting tissue, and in trees this tissue is generally known as wood (more precisely secondary xylem is wood) (Raven et al., 2005). The vascular cambium is a region of embryonic tissue where new cells are formed. During each growing season cell layers are added to the (secondary) phloem and xylem from the vascular cambium. After their formation in the vascular cambium, the xylem cells undergo expansion, maturation and finally programmed cell death (Hertzberg et al., 2001). This dataset thus allows us to study the expression of genes across different developmental stages.

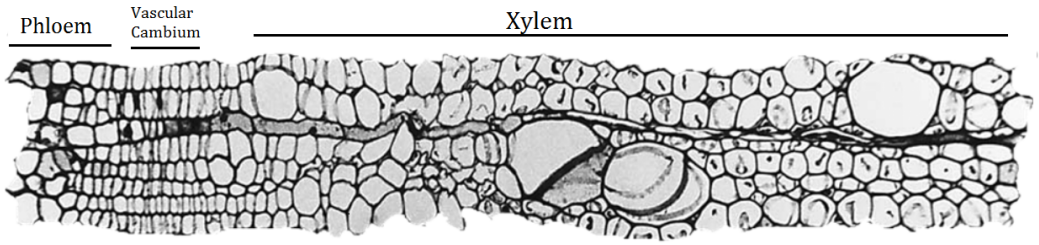


Figure 3.2: Tissues in cross-section of tree stem, from Bhalerao et al. (2003)

RNA-seq was used to determine the level of expression in the different stages. The reads were mapped to the *Populus trichocarpa* transcriptome and the expression level of each gene was quantified as Fragments Per Kilobase of transcript per Million mapped reads (FPKM). Figure 3.3 gives the expression levels in log(FPKM) across the developmental stages for two different genes.

The gene expression dataset contains 14 117 genes and 856 putative transcription factors. Transcription factors are proteins that regulate the transcription of genes by binding to DNA sequences neighbouring the regulated genes. The expression levels of the transcription factors are thus the explanatory variables in the statistical analysis, whereas the expression levels of the 14 117 genes are the response variables. If a transcription factor B regulates a specific gene A, one assumes that the expression levels of A and B will be correlated. If two transcription factors B and C regulate A together, one assumes that the product of the expression levels of B and C will be correlated to the expression level of A. The analysis was done on one gene (*i.e.* response variable) at the time. The results for ten of the genes is presented in Section 4.6.

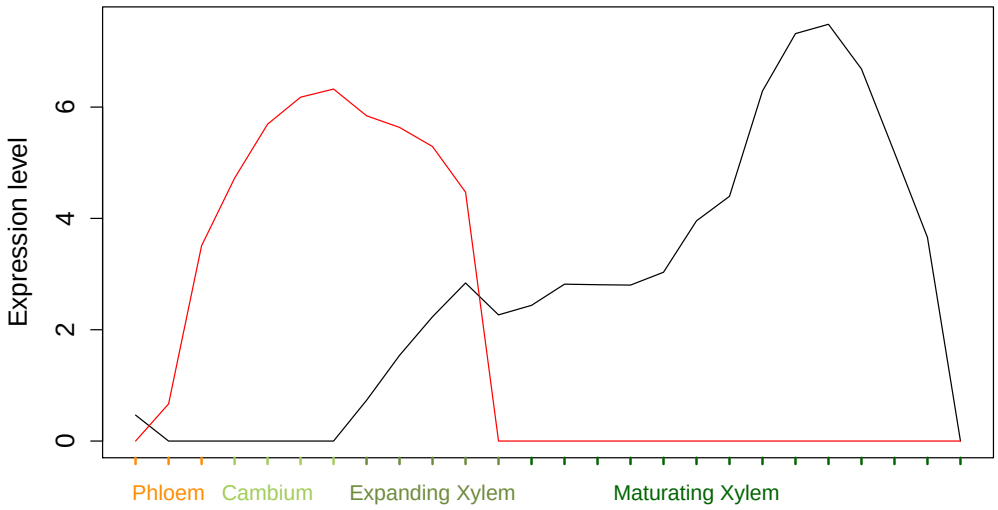


Figure 3.3: Expression levels in log(FPKM) of the genes POPTR_0013s11170 and POPTR_0012s14430 (in red) across developmental stages, from phloem to mature xylem.

Evaluation of gene expression results

As stated above, the methods in this thesis produce a ranked list of all the explanatory variables in the analysis. In the gene expression dataset, the explanatory variables are the expression levels of the transcription factors (TFs). The methods allow the user to find a small set of top ranked TFs for further investigation. When the variable ranking methods are applied to a real dataset, the true relevant variables are not known. In this section two methods for evaluation of the results from real datasets are presented. Both methods use Monte Carlo tests.

Method 1 - Predictive performance If the variable ranking methods perform well, a subset of top ranked variables should have a better predictive performance than random subsets of explanatory variables of equal size M. The following hypotheses are tested:

H_0: the top ranked TFs are a random set of TFs

H_1: the top ranked TFs have higher predictive performance than a random set of TFs

Let $R^2_{top,prediction}$ be the predictive performance of the M top ranked variables (calculated with a polynomial kernel PLS model of 2. order, to allow for interactions), and let $R^2_{random,prediction}$ be the predictive performance of a random subset of M explanatory variables. Thousand random subsets of M TFs are drawn and their predictive performance is evaluated. A p-value for the test can be calculated as

$$p - value_1 \approx \frac{N}{1000}$$

where N is the number of subsets where $R^2_{top,prediction} \leq R^2_{random,prediction}$.

Method 2 - Number of important interactions The variable selection methods should find pairs of explanatory variables which have interactions that are important for the prediction of the response, and the methods should find more such pairs than there would be in a random subset of variables of equal size. The following hypotheses are tested:

H_0: the top ranked TFs are a random set of TFs

H_1: the top ranked TFs have more pairs with important interactions than a random set of TFs

Let num_{top} be the number of "important interactions" (see definition below) among all the pairs of the M top ranked variables , and let num_{random}

be the number of "important interactions" among all the pairs of a random subset of M explanatory variables. Thousand random subsets of M TFs are drawn and their number of important two-way interactions is calculated. A p-value for the test can be

$$p - value_2 \approx \frac{N}{1000}$$

where N is the number of subsets where $num_{top} \leq num_{random}$.

One definition of "important interactions" can be: pairs of variables where the model with the interaction term is significantly better than the model without.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 \qquad \text{compared to}$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

This can be determined by a Williams' t-test for the difference between two non-independent Pearson correlations (Williams, 1959). Let

$$r_{12} = cor(\mathbf{y}_{val}, \hat{\mathbf{y}}_{val,inter})$$

$$r_{13} = cor(\mathbf{y}_{val}, \hat{\mathbf{y}}_{val,main})$$

$$r_{23} = cor(\hat{\mathbf{y}}_{val,inter}, \hat{\mathbf{y}}_{val,main})$$

where $cor()$ is the correlation, $\mathbf{y}_{val}$ is a vector of independent response observations, $\hat{\mathbf{y}}_{val,inter}$ are the predicted values of the response vector from a model with two main effects and an interaction term and $\hat{\mathbf{y}}_{val,main}$ are the predicted values of the response vector from a model with only main effects. Then a test statistic is defined as

$$T = (r_{12} - r_{13}) \sqrt{\frac{(m-1)(1+r_{23})}{2\left(\frac{m-1}{m-3}\right)|R| + \frac{(r_{12}+r_{13})^2}{4}(1-r_{23})^3}}$$

where $|R| = (1 - r_{12}^2 - r_{13}^2 - r_{23}^2) + 2r_{12}r_{13}r_{23}$ and m is the number of independent observations in $\mathbf{y}_{val}$. The test statistic T is t-distributed with $m - 3$ degrees of freedom, the distribution can be used to assess which pairs of variables have important interactions (with p-value for this T-test below 0.05).

3.2.2 Dataset generation

In order to assess the performance of the methods in 3.1, the methods were applied to simulated data. When constructing artificial data, it is desirable to be able to quantify the amount of information and noise in the data, in order to investigate the performance of the methods in different scenarios: data with low and high information content. For data with linear relationships without interactions, this is relatively straightforward. The response variable y and the relevant explanatory variables (henceforth called x_1 and x_2) can be drawn together from a multivariate normal distribution.

$$\begin{bmatrix} y \\ x_1 \\ x_2 \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_y \\ \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \sigma_y^2 & \sigma_{y,1} & \sigma_{y,2} \\ \sigma_{y,1} & \sigma_1^2 & \sigma_{1,2} \\ \sigma_{y,2} & \sigma_{1,2} & \sigma_2^2 \end{bmatrix} \right)$$

Where σ_y^2, σ_1^2 and σ_2^2 are the variances of y, x_1 and x_2 respectively. The relationship between the response variable and the relevant variables is decided by the covariances $\sigma_{y,1}$ and $\sigma_{y,2}$. The covariance between the two explanatory variables must be given so that the variance matrix above is positive-definite. The true regression coefficients ($\boldsymbol{\beta}$) of the linear model can be calculated:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon \qquad \epsilon \sim N(0, \sigma)$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \boldsymbol{\Sigma_{XX}^{-1} \sigma_{XY}} \qquad \beta_0 = \mu_y - \boldsymbol{\beta}^T \boldsymbol{\mu}_x \tag{3.2}$$

$$\text{Where } \boldsymbol{\Sigma_{XX}} = \begin{bmatrix} \sigma_1^2 & \sigma_{1,2} \\ \sigma_{1,2} & \sigma_2^2 \end{bmatrix}, \boldsymbol{\sigma_{XY}} = \begin{bmatrix} \sigma_{y,1} \\ \sigma_{y,2} \end{bmatrix} \text{ and } \boldsymbol{\mu_x} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$$

When the variables are drawn from a multivariate normal distribution (as here), the true amount of noise and the true R^2 can be calculated by the rules of conditional normal distributions (Bickel and Doksum, 2001).

$$\sigma^2 = var\left(y \middle| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = \sigma_y^2 - \boldsymbol{\sigma_{XY}^T \Sigma_{XX}^{-1} \sigma_{XY}} \tag{3.3}$$

$$R^2 = \frac{\boldsymbol{\sigma_{XY}^T \Sigma_{XX}^{-1} \sigma_{XY}}}{\sigma_y^2} \tag{3.4}$$

Data with a significant interaction term is more complicated to construct.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \epsilon$$

$$\text{Where } \epsilon \sim N(0, \sigma)$$

Although $\begin{bmatrix} y \\ x_1 \\ x_2 \\ x_1 x_2 \end{bmatrix}$ is not generally multivariate normally distributed, the definitions in (3.3) and (3.4) may still be considered approximatively true. The matrices $\boldsymbol{\Sigma_{XX}}$ and $\boldsymbol{\sigma_{XY}}$ are now equal to

$$\boldsymbol{\Sigma_{XX}} = \begin{bmatrix} \sigma_1^2 & \sigma_{1,2} & \sigma_{1,3} \\ \sigma_{1,2} & \sigma_2^2 & \sigma_{2,3} \\ \sigma_{1,3} & \sigma_{2,3} & \sigma_3^2 \end{bmatrix} \text{ and } \boldsymbol{\sigma_{XY}} = \begin{bmatrix} \sigma_{y,1} \\ \sigma_{y,2} \\ \sigma_{y,3} \end{bmatrix}$$

The variance of the interaction is represented by σ_3^2, $\sigma_{1,3}$ and $\sigma_{2,3}$ are the covariances between the interaction x_1x_2 and x_1 and x_2 respectively. The covariance between the interaction and the response is $\sigma_{y,3}$. These quantities can not be chosen indiscriminately because they are dependent on the parameters of x_1 and x_2. To determine the parameters of the interaction as a function of the parameters of x_1 and x_2, we use the three theorems below.

Mean, variance and covariance of quadratic forms. *(Rencher and Schaalje, 2008). Let* $\mathbf{z} \sim \mathbf{N_p}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ *and* $\mathbf{A}(\mathbf{p} \times \mathbf{p})$ *be non stochastic, then:*

$$E(\boldsymbol{z^T A z}) = trace(\boldsymbol{A\Sigma}) + \boldsymbol{\mu^T A \mu}$$

$$var(\boldsymbol{z^T A z}) = 2trace((\boldsymbol{A\Sigma})^2) + 4\boldsymbol{\mu^T A^T \Sigma A \mu}$$

$$cov(\boldsymbol{z}, \boldsymbol{z^T A z}) = 2\boldsymbol{\Sigma A \mu}$$

To apply the theorems to this problem one can choose

$$\mathbf{z} = \begin{bmatrix} y \\ x_1 \\ x_2 \end{bmatrix} \text{ and } \mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 \end{bmatrix}$$

and then one can calculate (with μ_3 the expectation of the interaction)

$$\mu_3 = E(x_1x_2) = \sigma_{1,2} + \mu_1\mu_2 \tag{3.5}$$

$$\sigma_3^2 = var(x_1x_2) = \sigma_{1,2}^2 + \sigma_1^2\sigma_2^2 + \mu_2^2\sigma_1^2 + 2\mu_1\mu_2\sigma_{1,2} + \mu_1^2\sigma_2^2 \tag{3.6}$$

$$\sigma_{1,3} = cov(x_1, x_1x_2) = \mu_1\sigma_{1,2} + \mu_2\sigma_1^2 \tag{3.7}$$

$$\sigma_{2,3} = cov(x_2, x_1x_2) = \mu_1\sigma_2^2 + \mu_2\sigma_{1,2} \tag{3.8}$$

$$\sigma_{y,3} = cov(y, x_1x_2) = \mu_1\sigma_{y,2} + \mu_2\sigma_{y,1} \tag{3.9}$$

The formulas (3.5) to (3.9) can be applied to calculate the parameters for the interaction given the parameter values for x_1 and x_2. However, (3.9)

is problematic for our application. In fact this formula implies that the interaction term does not provide any additional information apart from the information in x_1 and x_2. This was discovered when calculating the true regression coefficients for x_1, x_2 and x_1x_2.

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = \boldsymbol{\Sigma_{XX}^{-1}\sigma_{XY}} = \frac{1}{d} \begin{bmatrix} (\sigma_{y,1}\sigma_2^2 - \sigma_{y,2}\sigma_{1,2})(\sigma_1^2\sigma_2^2 + \sigma_{1,2}^2) \\ (\sigma_{y,2}\sigma_1^2 - \sigma_{y,1}\sigma_{1,2})(\sigma_1^2\sigma_2^2 + \sigma_{1,2}^2) \\ 0 \end{bmatrix}$$

Where d is the determinant of the matrix $\mathbf{\Sigma_{XX}}$.

The calculations above demonstrated that the true regression coefficient of the interaction, β_3, is always equal to zero for this set-up. Moreover the covariance between y and the interaction, given $\left[\begin{smallmatrix} x_1 \\ x_2 \end{smallmatrix}\right]$, can be shown to be equal to zero.

$$cov\left((y, x_1x_2) \middle| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = 0$$

This indicates that, for this set-up, x_1 and x_2 contain all the information, and the interaction term provides nothing in addition. A solution to this problem is to add a parameter, γ, to the covariance between y and the interaction. The parameter γ represents the additional information in the interaction term.

Let the new covariance be: $\sigma_{y,3} = cov(y, x_1x_2) = \mu_1\sigma_{y,2} + \mu_2\sigma_{y,1} + \gamma$

$$\text{Then: } \boldsymbol{\beta} = \frac{1}{d}\begin{bmatrix} (\sigma_{y,1}\sigma_2^2 - \sigma_{y,2}\sigma_{1,2})(\sigma_1^2\sigma_2^2 + \sigma_{1,2}^2) + \gamma\mu_2(\sigma_1^2\sigma_2^2 - \sigma_{1,2}^2) \\ (\sigma_{y,2}\sigma_1^2 - \sigma_{y,1}\sigma_{1,2})(\sigma_1^2\sigma_2^2 + \sigma_{1,2}^2) + \gamma\mu_1(\sigma_1^2\sigma_2^2 - \sigma_{1,2}^2) \\ \gamma(\sigma_1^2\sigma_2^2 - \sigma_{1,2}^2) \end{bmatrix} \quad (3.10)$$

$$\text{with } \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} \text{ and } cov\left((y, x_1x_2) \middle| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = \gamma$$

The calculations and formulas in this section contain all the necessary information for constructing data with a desired level of noise and information. In the following section a "recipe" and an example will be presented.

Simulation recipe

This section presents a method for constructing a response depending on two explanatory variables and an important interaction term.

1. Decide upon an expectation vector and a covariance matrix for x_1 and x_2 and draw a number of observations from the multivariate normal distribution with that covariance matrix (and with a given expectation vector).

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim N\left(\boldsymbol{\mu_{1,2}} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \boldsymbol{\Sigma_{1,2}} = \begin{bmatrix} \sigma_1^2 & \sigma_{1,2} \\ \sigma_{1,2} & \sigma_2^2 \end{bmatrix} \right)$$

2. Choose values of β_0 and $\boldsymbol{\beta}$, and construct the response $\mathbf{y}$ with those parameters.

$$y = \beta_0 + \beta_1x_1 + \beta_2x_2 + \beta_3x_1x_2 \quad (3.11)$$

3. Compute the 3×3 covariance matrix by using the formulas (3.6), (3.7), (3.8):

$$\mathbf{\Sigma_{XX}} = \begin{bmatrix} \sigma_1^2 & \sigma_{1,2} & \sigma_{1,3} \\ \sigma_{1,2} & \sigma_2^2 & \sigma_{2,3} \\ \sigma_{1,3} & \sigma_{2,3} & \sigma_3^2 \end{bmatrix}$$

$$\sigma_3^2 = \sigma_{1,2}^2 + \sigma_1^2\sigma_2^2 + \mu_2^2\sigma_1^2 + 2\mu_1\mu_2\sigma_{1,2} + \mu_1^2\sigma_2^2$$

$$\sigma_{1,3} = \mu_1\sigma_{1,2} + \mu_2\sigma_1^2$$

$$\sigma_{2,3} = \mu_1\sigma_2^2 + \mu_2\sigma_{1,2}$$

4. Compute the covariance between y and x_1, and y and x_2 using the rule for covariance between linear combinations of variables (Miller et al., 2004).

$$\sigma_{y,1} = cov(y, x_1) = cov(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2, x_1) = \beta_1\sigma_1^2 + \beta_2\sigma_{1,2} + \beta_3\sigma_{1,3}$$

$$\sigma_{y,2} = cov(y, x_1) = \beta_1\sigma_{1,2} + \beta_2\sigma_2^2 + \beta_3\sigma_{2,3}$$

5. Compute the γ parameter using the formula (3.10), and then find the covariance between y and the interaction,

$$\gamma = \frac{\beta_3 d}{\sigma_1^2\sigma_2^2 - \sigma_{1,2}^2}$$

$$\sigma_{y,3} = cov(y, x_1x_2) = \mu_1\sigma_{y,2} + \mu_2\sigma_{y,1} + \gamma$$

where d is the determinant of the covariance matrix $\mathbf{\Sigma_{XX}}$.

6. Now we have the vector $\sigma_{\mathbf{XY}}$ and the matrix $\mathbf{\Sigma_{XX}}$, and we can verify the vector of true beta-values by the formula (3.2). We can check that

the values in the vector are equal to the beta-values we specified in (3.11).

7. Decide upon a desired information level, R^2, and calculate the corresponding total amount of noise ($\sigma_{y,new}$).

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \epsilon \qquad \epsilon \sim N(0, \sigma)$$

$$\sigma^2_{y,new} = \frac{\boldsymbol{\sigma_{XY}^T \Sigma_{XX}^{-1} \sigma_{XY}}}{R^2}$$

8. In order to find the noise parameter σ^2 we need the to know the variance of the response σ_y^2 (without added noise). To compute this we use the rule for the variance of linear combinations of variables (Miller et al., 2004).

$$\begin{aligned}\sigma_y^2 = var(y) = \beta_1^2\sigma_1^2 + \beta_2^2\sigma_2^2 + \beta_3^3\sigma_3^2 \\ + 2\beta_1\beta_2\sigma_{1,2} + 2\beta_1\beta_3\sigma_{1,3} + 2\beta_2\beta_3\sigma_{2,3}\end{aligned}$$

$$\sigma^2 = \sigma^2_{y,new} - \sigma_y^2$$

Example 2. *Following the recipe above with* $\boldsymbol{\Sigma_{XX}} = \left[\begin{smallmatrix} 4 & 2.5 \\ 2.5 & 9 \end{smallmatrix}\right]$, $\mu_1 = 1$, $\mu_2 = 2$, $\beta_0 = 0$, $\beta_1 = \beta_2 = 1$, $\beta_3 = 10$ *and* $R^2 = 0.9$. *The table below gives the theoretical results according to the calculations above, and the results from a linear model with 1 000 000 observations.*

	Theoretical	*Simulated*
σ_3^2	*77.25*	*77.23*
$\sigma_{1,3}$	*10.5*	*10.47*
$\sigma_{2,3}$	*14*	*14*
$\sigma_{y,1}$	*111.5*	*111.24*
$\sigma_{y,2}$	*151.5*	*151.50*
$\sigma_{y,3}$	*797*	*796.74*
β_1	*1*	*1.10*
β_2	*1*	*0.91*
β_3	*10*	*9.99*
R^2	*0.9*	*0.90*
σ^2	*915*	*914.46*

The simulated results with 1 000 000 observations are very close to the theoretical results. The recipe presented above seems to provide an accurate method for specifying the amount of noise in a dataset with an interaction term.

3.3 Simulation studies

3.3.1 Goal

The goal of the simulation studies is to study the performance of the three methods in Section 3.1 in a systematic way. Their performance is defined as their ability to find *few* relevant explanatory variables among a large number of noise variables. The methods are initially studied separately, and are applied to different kinds of datasets and with different choices of method arguments. In the following section the protocol for the simulation studies will be presented: the dataset parameters, the method arguments and the performance measure.

3.3.2 Procedures

The simulations are run in R. All experiments are run on fully independent datasets, *i.e.* new data is randomly generated for every experiment, for every new scenario and method. In order to reduce the variability from the data, 10 replicates for every combination of scenario and method were used.

3.3.3 Dataset generation

The datasets consist of a $n \times 1$ response vector $\mathbf{y}$, and a matrix $\mathbf{X}$ with n observations of p variables. Only two among these p variables are generated with a relationship to the response, these two variables are considered the relevant variables and the methods are evaluated according to their ability to find these two variables (henceforth called x_1 and x_2).

The response and the two relevant explanatory variables are generated ac-

cording to the recipe in 3.2.2. The parameters are chosen as $\mathbf{\Sigma_{XX}} = \left[\begin{smallmatrix} 1 & 0.5 \\ 0.5 & 1 \end{smallmatrix}\right]$, $\mu_1 = 0$, $\mu_2 = 0$, $\beta_0 = 0$ and $\beta_1 = \beta_2 = 1$. The two last dataset parameters that need to be specified, the interaction coefficient β_3 and the true information content R^2, are allowed to vary and are further discussed in the subsection below.

The noise variables are drawn independently from a normal distribution with $\mu = 0$ and $\sigma = 1$.

3.3.4 Scenarios

The performance of the methods is evaluated on 16 different scenarios, or kinds of datasets. The scenarios are defined by four factors with two levels each (Figure 3.1).

	Levels	
n	50	100
p	100	1000
β_3	1	10
R^2	0.2	0.9

Table 3.1: Factors defining 16 different scenarios.

The methods are studied on datasets with few observations ($n = 50$) and with more observations ($n = 100$), and on datasets with relatively few variables ($p = 100$) and with many variables ($p = 1000$). In addition, the scenarios are defined by two parameters controlling the relationship between the response and the relevant explanatory variables: the size of the interaction term and the relative amount of information (and noise) in the relationship

between the response and the relevant explanatory variables (Figure 3.2).

		Information level	
		Low	High
Interaction size	Small	$\beta_3 = 1, R^2 = 0.2$	$\beta_3 = 1, R^2 = 0.9$
	Large	$\beta_3 = 10, R^2 = 0.2$	$\beta_3 = 10, R^2 = 0.9$

Table 3.2: The four scenarios defined by the relationship between the response and the relevant variables. These scenarios will henceforth be called "small interaction, low information", "small interaction, high information", "large interaction, low information" and "large interaction, high information".

3.3.5 Methods to be evaluated

Each of the three methods introduced in 3.1 have some arguments that can be adjusted. These arguments will be factors in the analysis of the performances.

PR

As noted in Section 3.1.1, the number of components in the kernel PLS model could influence the performance of the PR method. The components number 1, 4 and 8 are tested. The experiments are used to assess whether the number of components used to calculate the distance, have any effect on the ranking of the relevant explanatory variables in different situations.

Factor associated with the PR method:

- Number of components (3 levels)

IR

In this method, a PLS model is fitted on the extended **Q** matrix. The number of components is a factor in the analysis, three levels are chosen: 1 component, 4 components and 8 components.

In addition two different scoring methods can be used in the IR method. One which scores the variables with the $\hat{\beta}$ vector from the PLS model, the other scores with the T-statistic after jackknifing.

The IR method can be used in combination with the PR method (useful when there are many variables), and then the level of filtering from the PR method can influence the outcome. This will be discussed in Chapter 4.

Factors associated with the IR method:

- Number of components (3 levels)
- Scoring method (2 levels)

RIR

The method arguments in RIR are the number of variables per iteration, the number of iterations, the scoring methods ($\hat{\beta}$s or T-statistics from jackknifing) and potentially the filter threshold. However the filter threshold is not a factor in the simulations, and is kept constant and equal to 0.1.

In order to simplify the analysis, on can let the number of iterations be a function of the probability of drawing the relevant variables together ($prob$), the number of variables (p) and the number of variables per iteration (v). It

can be shown that the number of iterations g is equal to:

$$g = \frac{\log_{10}(1 - prob)}{\log_{10}(1 - \frac{v(v-1)}{p(p-1)})} \tag{3.12}$$

Where *prob* is the probability of drawing the two relevant variables together (ones, twice or more times) from a total of p variables with v variables in each iteration.

This allows us to run the experiments with a fixed probability of drawing the right variables together. For example if you have 100 variables and want to be 99.9 % certain that the relevant variables are drawn together at least once, you need to run 40, 176 or 756 iterations depending on the number of variables per iteration (40, 20 or 10). With 1000 variables the corresponding number are 4 420, 18 157 and 76 673.

Factors associated with the RIR method:

- Number of variables per iteration (2 levels*)
- Probability of drawing the relevant variables together (3 levels*)
- Scoring method (2 levels)

RIR with T-scoring is slower than with β-scoring, and the running time of RIR with T-scoring is also more sensitive to the number of variables per iterations. So while RIR with β-scoring is tested with the probabilities 0.9, 0.995 and 0.999 and with 10 and 40 variables per iteration, RIR with T-scoring is only tested with the probabilities 0.9 and 0.995 and with 10 and 20 variables per iteration.

3.3.6 Performance criteria and analysis

All three methods produce a ranking of the explanatory variables where the variables considered most important are given a high rank. The chosen performance criteria for all the methods is the probability of ranking both relevant variables amongst the ten most important variables. The probability $\pi = P(Z = 1)$ is defined by:

$$Z = 0 \iff \textit{one or both relevant variables are not ranked among the 10 highest ranked variables}$$

$$Z = 1 \iff \textit{both relevant variables are ranked among the 10 highest ranked variables}$$

Other performance measures could have been chosen, the justification for choosing this measure is that it clearly captures whether the relevant variables are given high scores, and therefore will be analysed further, or not.

Each of the simulation studies produce a vector with zeros and ones, which is analysed by logistic regression (see Section 2.5). All explanatory variables in this meta-model are considered categorical, as factors. I first fitted the most complex logistic regression model possible: the model with interactions between all the factors present (5th or 6th order interactions). I then applied a backward elimination procedure to simplify the model as much as possible. That is, I sequentially eliminated the least significant terms in the model, starting with the interactions of highest order. I used the LR tests (see Section 2.5) for this purpose. Interactions of lower order could only be eliminated if they were not part of any significant interactions of higher order (same with main effects).

Another important measure is the running time for the methods. The running times for the three methods are compared to the running time of an exhaustive search. An exhaustive search considers all possible pairs of explanatory variables, fits a linear model with the two-way interaction and main effects for all pairs, and scores them according to their predictive performance on an independent validation test. The running time for exhaustive search are shown in Table 3.3.

$p = 100$	7.4
$p = 1000$	920

Table 3.3: Running times (in seconds) for exhaustive search with few and many variables.

The methods introduced in this thesis are supposed to have shorter running times than the exhaustive search. Running times are collected with the R function system.time().

Chapter 4

Results

In this chapter the results from the simulation studies and the analysis on real data are presented. The sections concerning the simulation studies (4.1 to 4.5) describe the results from the analysis by logistic regression. Only significant effects are commented upon, *i.e.* likelihood-ratio tests with p-value below 0.05.

4.1 PR

Time

The Pseudoloadings Ranking is a very fast method. The Table 4.1 gives the running times (in seconds) for four different combinations of p and n. As expected, larger $\mathbf{X}$ matrices require longer time. More components in the distance calculations also requires longer time.

$p = 100, n = 50$	0.003
$p = 100, n = 100$	0.004
$p = 1000, n = 50$	0.006
$p = 1000, n = 100$	0.011

Table 4.1: Running times for PR (in seconds) for four combinations of p and n (averaged over 40 experiments each)

Performance

As expected the ranking of the relevant variables is significantly improved by an increased information content in the data (Figure 4.1). Increasing the number of observations also has a positive effect. Another expected effect is that increasing the number of noise variables has an adverse effect (Figure 4.1).

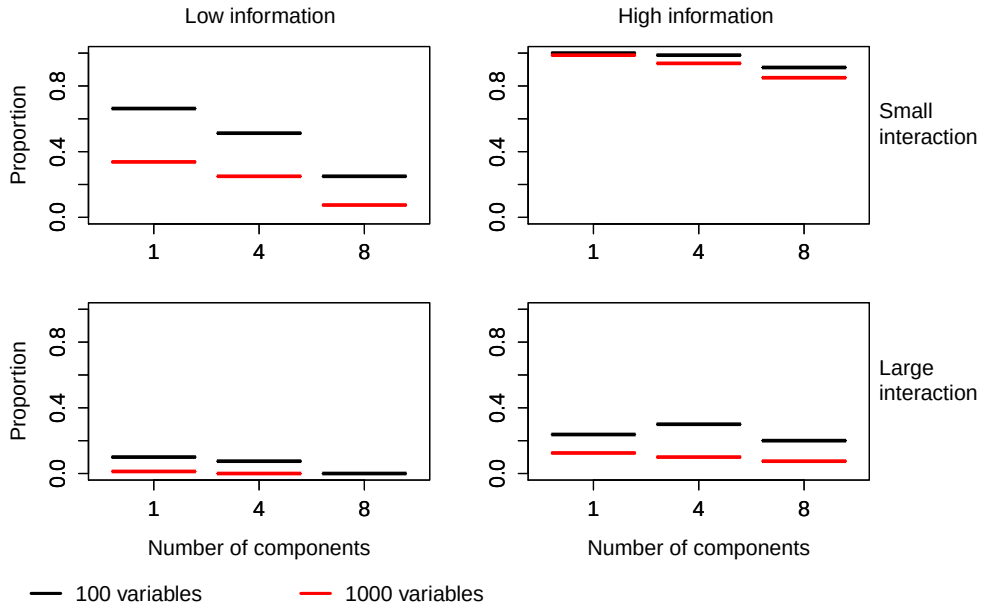


Figure 4.1: Proportion of experiments where the PR method ranked the relevant variables among the 10 most important variables (1920 experiments). Effect of number of explanatory variables and number of components.

The method performs quite well in situations with a small interaction compared to the main effects, but the performance is significantly reduced when the interaction is large (Figure 4.1). Increasing the number of components has a overall negative effect, however when the interaction term is large the negative effect is less pronounced or even reversed (Figure 4.1).

4.2 IR

4.2.1 IR with β-scoring

Time

The IR method with β-scoring is slower than PR, but still quite fast (Table 4.2). However the use of the IR method can be restricted by memory limitations (this is discussed in Chapter 5).

$p = 100$	0.34
$p = 1000$	42.06

Table 4.2: Running times (in seconds) for IR with β-scoring for few and many variables (average of 240 experiments each)

Performance

The performance of IR with β-scoring is not significantly related to the number of components in the analysis. An increased number of observations has a positive effect and more noise variables has a negative effect, just as for PR (and as expected) (Figure 4.2). While the PR method performed worse for scenarios with large interaction, the opposite is the case for the IR method: the performance is significantly better when the interaction term is large.

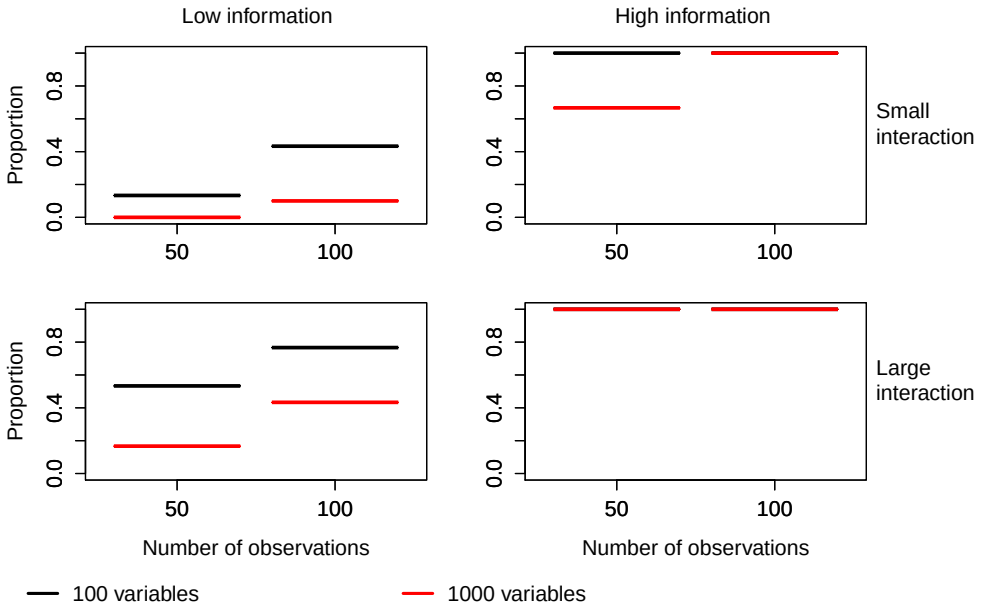


Figure 4.2: Proportion of experiments where the IR method with β-scoring ranked the relevant variables among the 10 most important variables (480 experiments). Effect of number of explanatory variables and number of observations.

4.2.2 IR with T-scoring

Time

Because of the cross-validation, IR with T-scoring is a much more time consuming method than IR with β-scoring (Table 4.3).

Due to the increased running times, IR with T-scoring could not be used on the entire $\mathbf{X}$ matrix when $p = 1000$ (this would take up to 4000 seconds per experiment). For those scenarios, IR was run only on the 250 highest scored variables from PR. In some experiments the relevant explanatory variables were not among the top 250 after PR-filtration, and the experiment was then stopped. The results in Table 4.4 seem consistent with the analysis of PR

$p = 100, n = 50$	7.38
$p = 100, n = 100$	23.78
$p = 250, n = 50$	46.91
$p = 250, n = 100$	165.4

Table 4.3: Running times (in seconds) for IR with T-scoring for few and many variables (the two first are based on 120 experiments each, the third is based on 79 and the last on 83)

(Section 4.1). In the subsequent analysis, the experiments without scoring with IR are disregarded (as this is in fact an effect from PR).

Small interaction, low information	0.08
Small interaction, high information	0
Large interaction, low information	0.75
Large interaction, high information	0.47

Table 4.4: Proportion of experiments where the relevant variables were not scored with the IR method (because they were discarded by PR).

Performance

Just as for the other methods, the performance of IR with T-scoring is positively affected by increasing the number of observation and the information content, whereas increasing the number of noise variables is negative for the performance. Just like IR with β-scoring, IR with T-scoring performs better when the interaction term is large. The effect of increasing the number of components in the analysis is in general negative, but is not so clear (Figure 4.3).

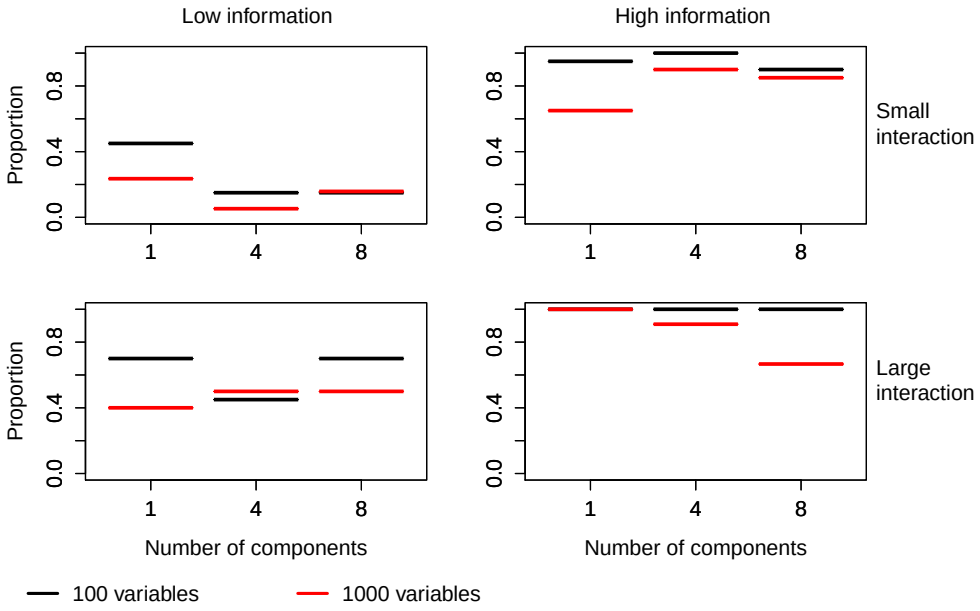


Figure 4.3: Proportion of experiments where the IR method with T-scoring ranked the relevant variables among the 10 most important variables (480 experiments). Effect of number of explanatory variables and number of components.

In Figure 4.4, one can observe that the negative effect of increasing the number of noise variables can almost be eliminated by increasing the number of observations.

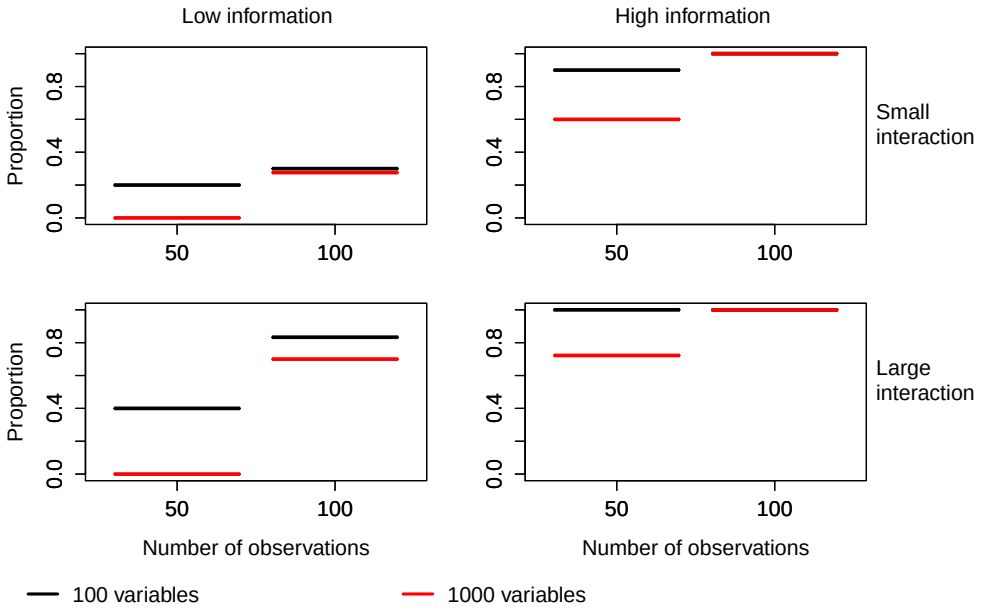


Figure 4.4: Proportion of experiments where the IR method with T-scoring ranked the relevant variables among the 10 most important variables (480 experiments). Effect of number of explanatory variables and number of observations.

4.2.3 Comparing the scoring methods

Comparing scoring methods when there are few variables

There is no significant difference between the methods when comparing their performance for a small number of noise variables ($p = 100$) (Figure 4.5).

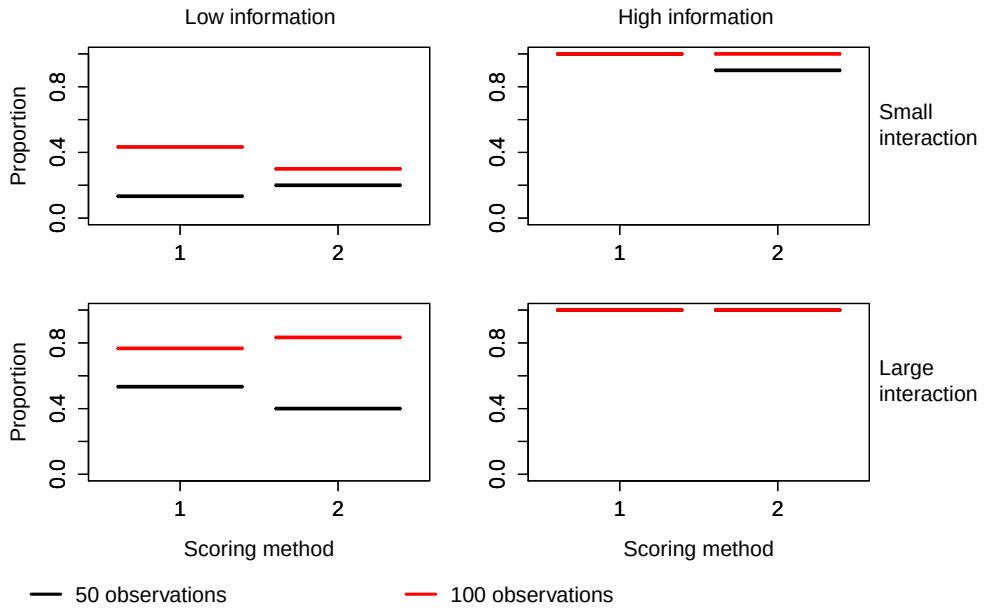


Figure 4.5: Proportion of experiments where the relevant variables were ranked among the 10 most important variables (480 experiments). Scoring method 1 is IR with β-scoring, method 2 is IR with T-scoring.

Comparing scoring methods in general

In order to do a fair comparison between the scoring methods, IR with β-scoring is run again with PR-filtration first (the number of not-scored experiments are similar to with T-scoring, as expected). The analysis that follows is based on 960 experiments (where 799 are actually scored with IR).

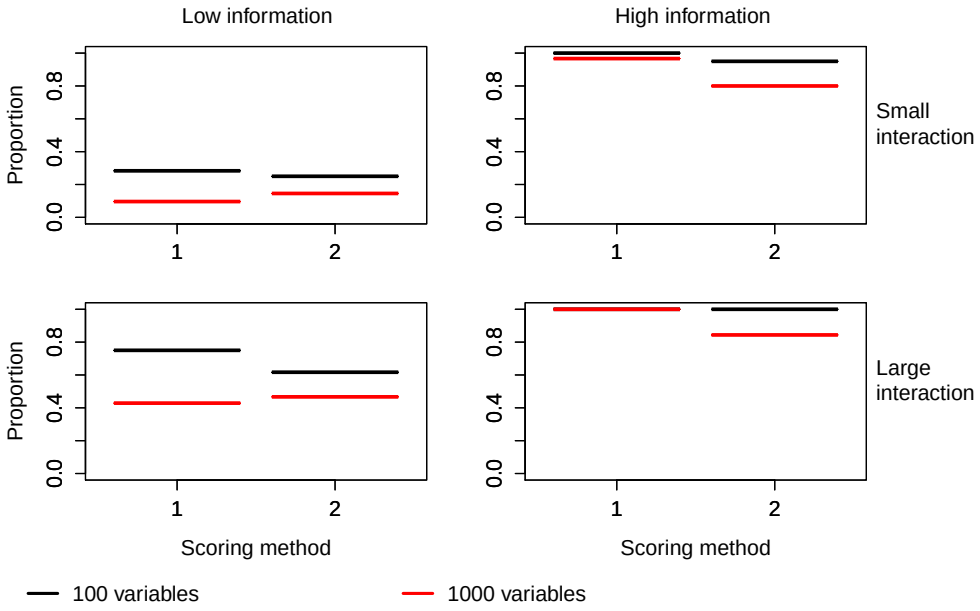


Figure 4.6: Proportion of experiments where the relevant variables were ranked among the 10 most important variables (799 experiments). Scoring method 1 is IR with β-scoring, method 2 is IR with T-scoring.

However this analysis reveals no general significant difference between the two scoring methods. Scoring with T-scores seems to perform worse than scoring with βs when there is a high information content, especially when there are many noise variables (Figure 4.6).

4.3 RIR

4.3.1 RIR with β-scoring

Time

RIR is a much slower method than PR and IR. The time increases notably with the number of variables, and also increases with the number of observations and the probability of drawing the relevant variables together (Table 4.5). However, the time decreases when the number of variables per iteration increases (v).

	$prob = 0.9$	$prob = 0.995$	$prob = 0.999$
$p = 100, n = 50, l = 10$	1.7	3.7	4.7
$p = 100, n = 50, l = 40$	1.2	2.5	3.4
$p = 100, n = 100, l = 10$	2.3	5.1	6.1
$p = 100, n = 100, l = 40$	2.1	4.0	5.3
$p = 1000, n = 50, l = 10$	155.3	366.7	482.7
$p = 1000, n = 50, l = 40$	112.4	249.9	328.4
$p = 1000, n = 100, l = 10$	193.6	460.8	608.1
$p = 1000, n = 100, l = 40$	157.8	344.7	452.2

Table 4.5: Running times (in seconds) for RIR with β-scoring for different scenarios (based on 10 experiments each for prob=0.9, on 40 or a little less for prob=0.995)

On average, 84% of the iterations are skipped because of the filter threshold in RIR (see 3.1.3). In some experiments when there are few explanatory variables, all iterations are skipped because none of the random variables set had more predictive power than the entire dataset. Those experiments were

removed from this analysis.

Performance

Figure 4.7 illustrates how the performance of RIR with β- scoring is positively affected by increased information content. Increasing the number of noise variables has no effect (even slightly positive) when the interaction term is small, but when the interaction is large the performance is greatly reduced when the number of noise variables increases. This is especially evident when there is low information.

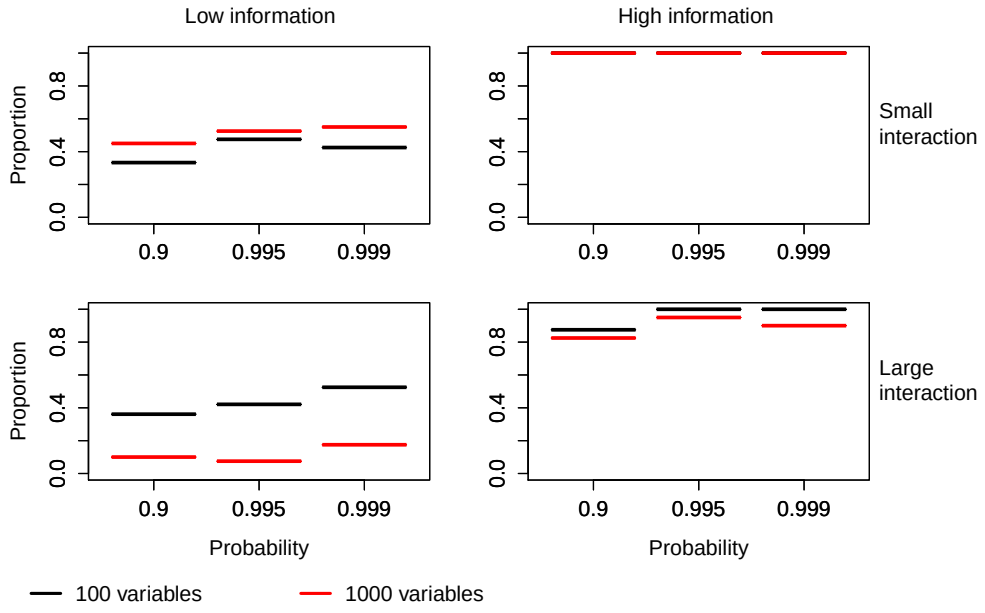


Figure 4.7: Proportion of experiments where the RIR method with β-scoring ranked the relevant variables among the 10 most important variables (950 experiments). Effect of number of explanatory variables and probability of drawing the relevant variables together.

The performance is positively affect by an increased probability of drawing the relevant variables together (significant difference between 0.9 and 0.995/0.999). This is not surprising as the increased probability means a larger number of iterations. The difference is not very large however, and the figure suggests that increasing the probability may be most useful in the most difficult scenario (low information, large interaction).

A large number of observations is as expected positive for the performance. A smaller number of variables per iteration is beneficial for the result, and this is especially apparent when the number of observations is large (Figure 4.8).

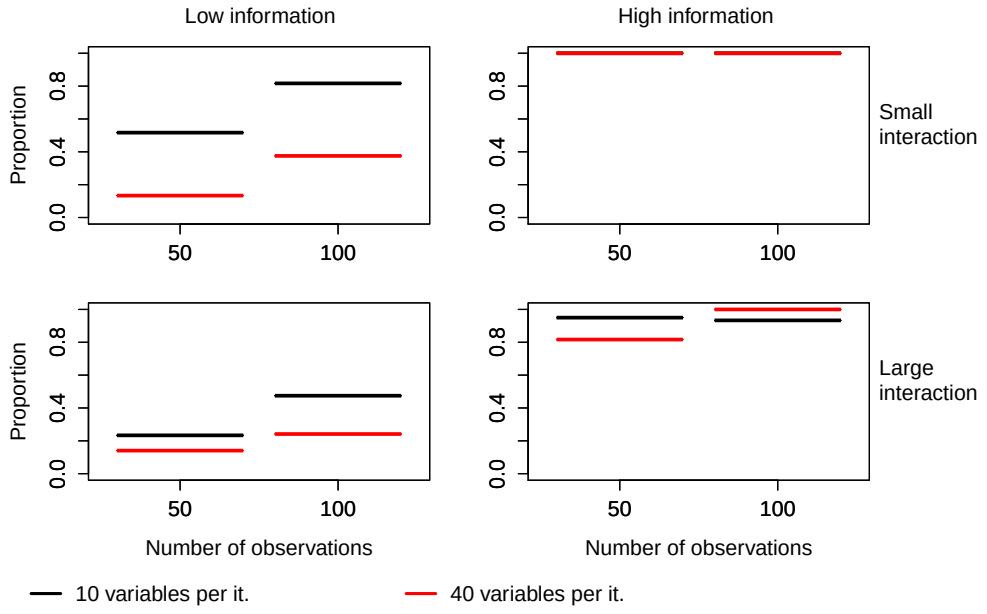


Figure 4.8: Proportion of experiments where the RIR method with β-scoring ranked the relevant variables among the 10 most important variables (950 experiments). Effect of number of variables per iteration and number of observations.

4.3.2 RIR with T-scoring

Time

Because of cross-validation RIR with T-scoring is in general somewhat slower than RIR with β-scoring. As with the related method, the time increases with p, n and $prob$ and decreases when v, the number of variables per iteration, increases from 10 to 20 (Table 4.6).

	$prob = 0.9$	$prob = 0.995$
$p = 100, n = 50, l = 10$	1.0	7.4
$p = 100, n = 50, l = 20$	0.9	5.4
$p = 100, n = 100, l = 10$	1.5	9.5
$p = 100, n = 100, l = 20$	1.3	8.1
$p = 1000, n = 50, l = 10$	224.7	696.5
$p = 1000, n = 50, l = 20$	243.6	530.8
$p = 1000, n = 100, l = 10$	378.7	733.3
$p = 1000, n = 100, l = 20$	285.9	651.2

Table 4.6: Running times (in seconds) for RIR with T-scoring for different scenarios (based on 40 experiments each or a little less).

The filtration step in the RIR algorithm assures that a large proportion of the variable sets are not scored (because they have less predictive power than the entire dataset). On average 88% of the variables sets are skipped. As for RIR with β-scoring, a few of the experiments skipped all the iterations and no variables were scored. These were removed from this analysis.

Performance

High information is (as always) beneficial for the performance of the method (Figure 4.9). Increasing the number of noise variables is only detrimental in the scenarios with large interaction terms. Contrary to expectations, an increase in the probability of drawing the relevant variables together does not lead to an large increase in the performance, except when p is large and especially in the high information, large interaction scenario. Drawing 10 variables per iteration is significantly better than drawing 20 variables, but this effect is attenuated when the number of observations is large (Figure 4.10). A large number of observation is generally positive.

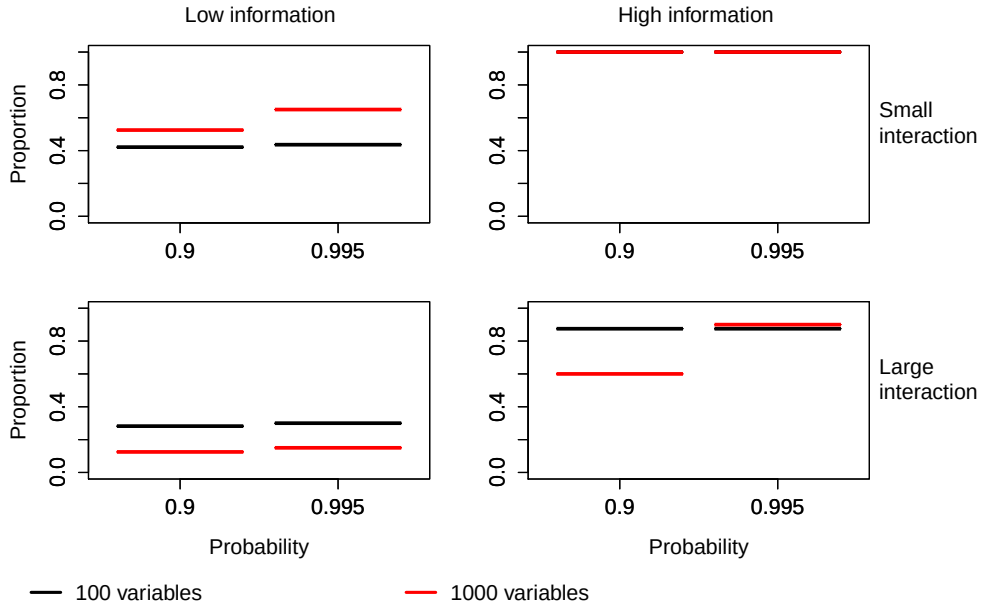


Figure 4.9: Proportion of experiments where the RIR method with T-scoring ranked the relevant variables among the 10 most important variables (636 experiments). Effect of number of explanatory variables and probability of drawing the relevant variables together.

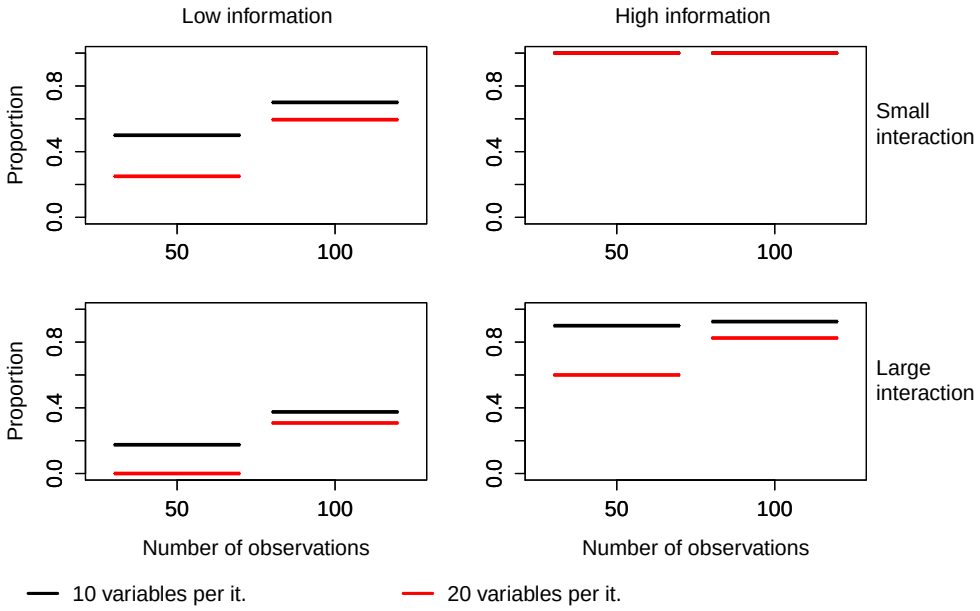


Figure 4.10: Proportion of experiments where the RIR method with T-scoring ranked the relevant variables among the 10 most important variables (636 experiments). Effect of number of variables per iteration and number of observations.

4.3.3 Comparing the scoring methods

Is the slower scoring method (T-scoring) better than the faster? When the methods are compared with equal settings (both with $prob = 0.995$ and $v = 10$) there is no significant difference between them (Figure 4.11).

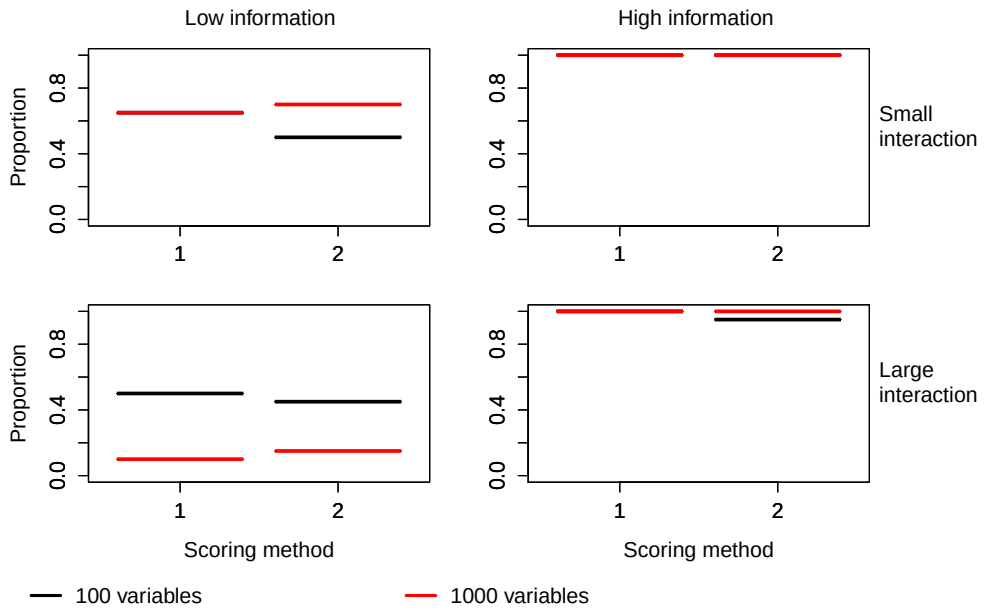


Figure 4.11: Proportion of experiments where the relevant variables were ranked among the 10 most important variables (320 experiments). Scoring method 1 is RIR with β-scoring, method 2 is RIR with T-scoring.

4.4 Comparing IR and RIR

The methods IR and RIR are also compared, each with their best settings. For RIR: β-scoring, $v = 10$, $prob = 0.999$; and for IR: β-scoring, without PR-filtrering, 1 component. Each method has 160 experiments. The RIR method performs significantly better than the IR method in general. But as Figure 4.12 demonstrates the difference between the methods, originates mainly from the small interaction scenarios (and especially when p is large).

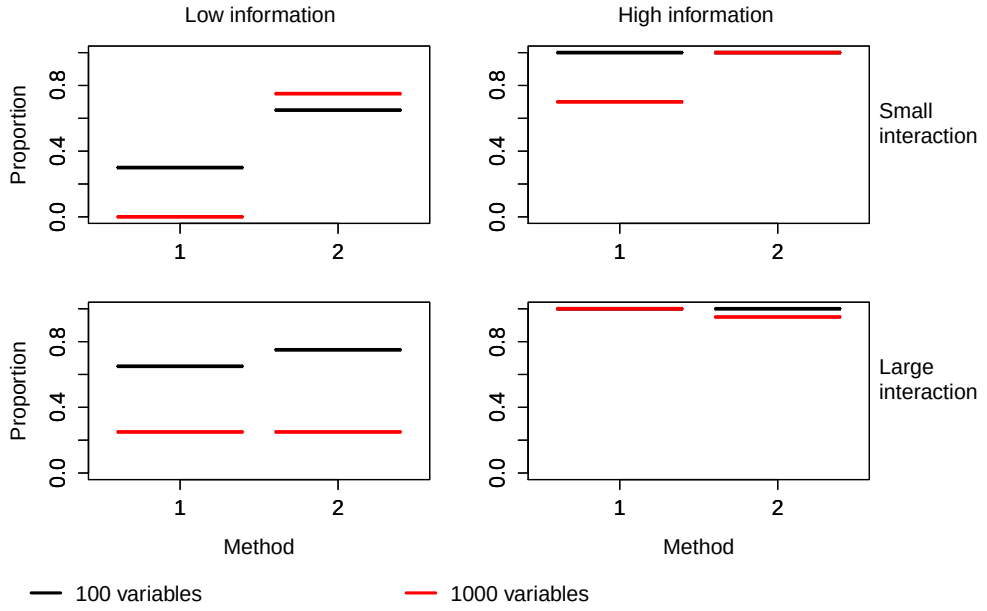


Figure 4.12: Proportion of experiments where the relevant variables were ranked among the 10 most important variables (based on 320 experiments), Method 1 is IR with β-scoring, method 2 is RIR with β-scoring.

4.5 SIRI

4.5.1 Time and Performance

SIRI was run with 2 slices, and with 2 iterations. The SIRI procedure from Jiang and Liu (2013) is very fast (Table 4.7). The running time clearly increases more slowly with p than it does for the IR or RIR methods.

$p = 100$	0.38
$p = 1000$	2.72

Table 4.7: Running times (in seconds) for SIRI for few and many variables (average of 240 experiments each)

The performance of the SIRI procedure is greatly reduced when the information content is small compared with when it is large. The performance also decreases when the number of noise variables is large (Figure 4.13).

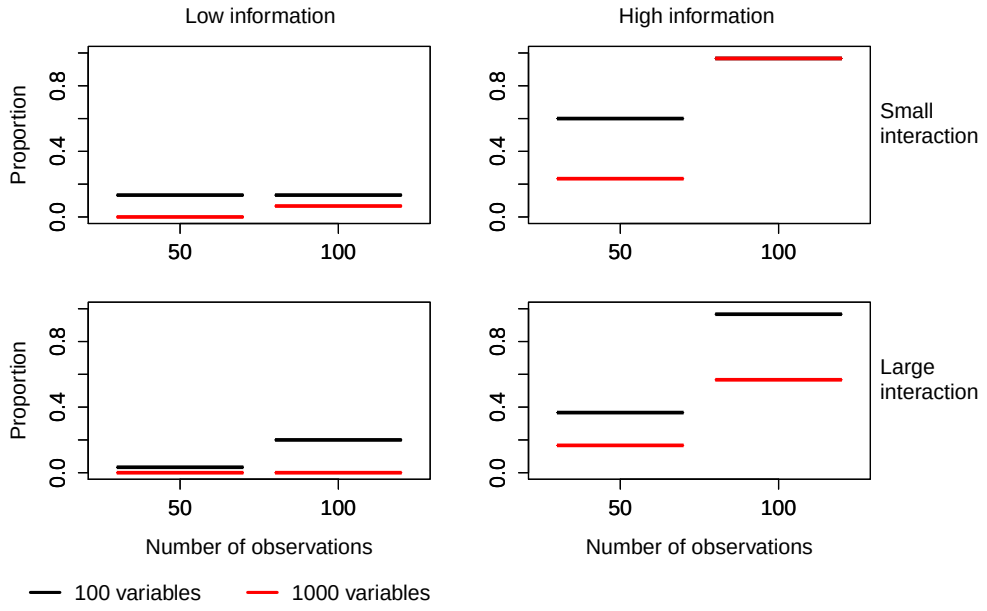


Figure 4.13: Proportion of experiments where the SIRI procedure ranked the relevant variables among the 10 most important variables (480 experiments). Effect of number of variables per iteration and number of observations.

4.5.2 Comparing with IR and RIR

The performance of SIRI is significantly worse than the performance of both IR with β-scoring and RIR with β-scoring (same settings as in 4.4) (Figure 4.14).

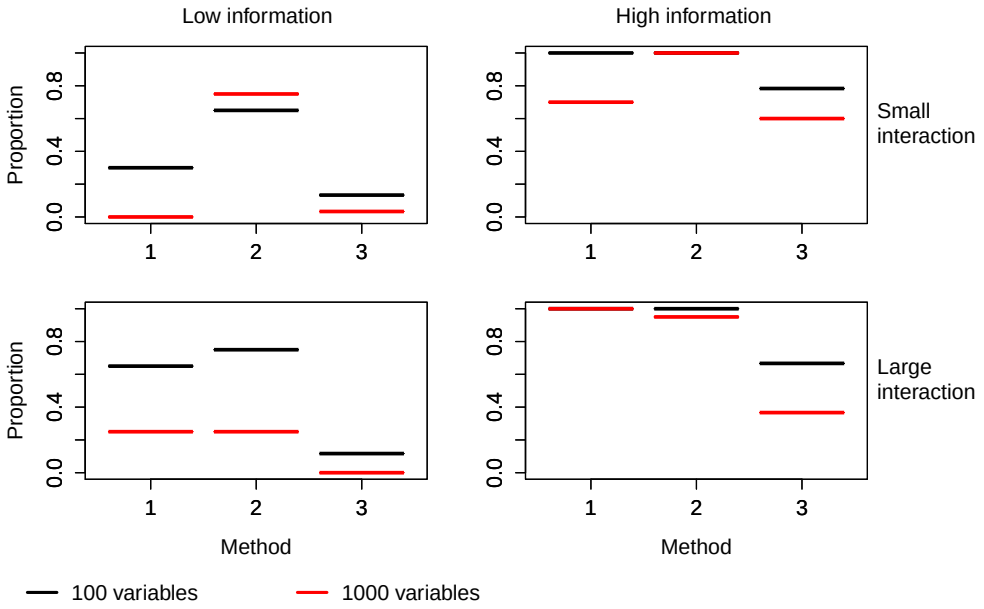


Figure 4.14: Proportion of experiments where the relevant variables were ranked among the 10 most important variables (based on 800 experiments), Method 1 is IR with β-scoring, method 2 is RIR with β-scoring and method 3 is SIRI.

4.6 Gene expression data

The methods IR and RIR were used to identify the relevant explanatory variables of ten genes (the responses variables). The explanatory variables are the expression levels of the 856 putative transcription factors (TFs).

4.6.1 IR

The data are analysed with IR with β-scoring and 1 component. The average running time of IR is 30 seconds for each gene. IR outputs a ranked list of all TFs and pairs of TFs. However, the 45 highest ranked explanatory variables are almost exclusively pairs. Many TFs appeared several times among the top 45, and the top pairs consist of 33 to 46 unique TFs.

Genes	$R^2_{prediction}$ all	$R^2_{prediction}$ top 10	p-value
POPTR_0001s10400	0.94	0.93	0.045
POPTR_0012s14430	0.99	0.99	0.010
POPTR_0017s12100	0.96	0.91	0.074
POPTR_0007s00290	0.94	0.91	0.021
POPTR_0008s02680	0.93	0.90	0.293
POPTR_0006s28140	0.92	0.92	0.007
POPTR_0009s15840	0.96	0.97	0.008
POPTR_0013s11170	0.97	0.96	0.025
POPTR_0001s15590	0.95	0.95	0.126
POPTR_0018s11390	0.85	0.78	0.556

Table 4.8: $R^2_{prediction}$ for the entire dataset and for the ten best TFs from IR, with corresponding p-values.

The results are first evaluated with method 1 from 3.2.1. The predictive performance of the ten unique TFs belonging to the highest ranked pairs was compared with the predictive performance of random sets of ten TFs. For six among the ten genes, the top ranked set has a significantly higher predictive performance than random sets (Table 4.8). However, the predictive performance with the ten best TFs is generally identical or slightly lower than the performance of the entire dataset.

Another evaluation method is described in Section 3.2.1, concerning the ability of the method to discover "important interactions", *i.e.* interactions that significantly improve the prediction. The 45 highest ranked pairs of TFs are evaluated according to that method (for a few of the genes only 44 pairs, because a single TF was ranked in the top 45) . The results are shown in Table 4.9. For several of the genes a significant number of important interactions is identified.

Genes	Important interactions	p-value
POPTR_0001s10400	4	0.930
POPTR_0012s14430	35	0.000
POPTR_0017s12100	9	0.035
POPTR_0007s00290	7	0.159
POPTR_0008s02680	23	0.000
POPTR_0006s28140	3	0.759
POPTR_0009s15840	42	0.000
POPTR_0013s11170	24	0.000
POPTR_0001s15590	19	0.000
POPTR_0018s11390	2	0.967

Table 4.9: Number of important interactions and corresponding p-values for the 45 highest ranked pairs of TFs from IR.

4.6.2 RIR

The data are analysed with RIR with β-scoring, 10 000 iterations and 20 variables per iterations. The average running time of RIR is 302 seconds per gene. The skipping threshold is set to 0 (see 3.1.3). The proportion of sets that are skipped varies from 19 % to 97 %, with an average value of 58 %. As expected, the time increases when the proportion of skipped set decreases.

RIR outputs a ranked list of all TFs. The ten highest ranked TFs are evaluated with method 1 from 3.2.1. The predictive performance of this set is compared with the predictive performance of random sets of ten TFs. Four genes have top ranked TFs with a significantly larger predictive performance (Table 4.10).

Genes	$R^2_{prediction}$ all	$R^2_{prediction}$ top 10	p-value
POPTR_0001s10400	0.94	0.91	0.152
POPTR_0012s14430	0.99	0.99	0.038
POPTR_0017s12100	0.96	0.93	0.014
POPTR_0007s00290	0.94	0.91	0.024
POPTR_0008s02680	0.93	0.91	0.179
POPTR_0006s28140	0.92	0.86	0.290
POPTR_0009s15840	0.96	0.97	0.005
POPTR_0013s11170	0.97	0.93	0.297
POPTR_0001s15590	0.95	0.96	0.074
POPTR_0018s11390	0.85	0.86	0.128

Table 4.10: $R^2_{prediction}$ for the entire dataset and for the 10 best TFs from IR, with corresponding p-values for the 10 highest ranked TFs for each gene.

The ten top ranked TFs can form 45 different interaction pairs. The number of "important interactions" among these 45 pairs is used as an evaluation criteria according to method 2 in 3.2.1. Five genes have top ranked TFs with a larger number of important interactions than would have been expected by chance (Table 4.11).

Genes	Important interactions	p-value
POPTR_0001s10400	11	0.271
POPTR_0012s14430	18	0.002
POPTR_0017s12100	5	0.404
POPTR_0007s00290	17	0.003
POPTR_0008s02680	17	0.012
POPTR_0006s28140	8	0.111
POPTR_0009s15840	14	0.012
POPTR_0013s11170	14	0.017
POPTR_0001s15590	8	0.255
POPTR_0018s11390	4	0.710

Table 4.11: Number of important interactions and corresponding p-values for the 45 pairs from the 10 highest ranked TFs from RIR for each gene

Unlike IR, RIR is a randomized method which produces different results for every run. The consistency of the method is therefore studied. The method is run twice (with the same parameters) on the same genes and the results are compared in Table 4.12. The two runs find somewhat similar TFs, and have similar conclusions on the tests from method 2 for most of the genes.

Genes	Top 42	Similar p-value
POPTR_0001s10400	26	yes
POPTR_0012s14430	13	yes
POPTR_0017s12100	24	no
POPTR_0007s00290	26	yes
POPTR_0008s02680	20	yes
POPTR_0006s28140	27	yes
POPTR_0009s15840	24	yes
POPTR_0013s11170	12	almost
POPTR_0001s15590	25	almost
POPTR_0018s11390	21	yes

Table 4.12: Number of similar TFs among the top 5% (42) TFs for two runs of RIR, also noted if the two runs had similar p-values (below or over 0.05) for the number of important interactions.

4.6.3 Comparing IR and RIR

The RIR and IR methods have similar goals and should therefore identify similar TFs when run for the same genes. The 42 highest ranked TFs from (the first run of) RIR and the 42 TFs constituting the highest ranked pairs from IR are compared (Table 4.13). The results overlap to a certain degree but not completely. The p-values from the test on the number of important interactions (method 2) are also compared, for eight out of the ten genes the two methods conclude in similar manner, *i.e.* both methods either find a significant number of important interactions or do not. For genes number 2,5,7 and 8 both methods report a significant number of important interactions.

Genes	Top 42	Similar p-value
POPTR_0001s10400	18	yes
POPTR_0012s14430	12	yes
POPTR_0017s12100	18	yes
POPTR_0007s00290	7	no
POPTR_0008s02680	7	yes
POPTR_0006s28140	8	yes
POPTR_0009s15840	8	yes
POPTR_0013s11170	5	yes
POPTR_0001s15590	6	no
POPTR_0018s11390	10	yes

Table 4.13: Number of similar TFs among the top 5% (42) TFs for RIR and IR, also noted if the two runs had similar p-values (below or over 0.05) for the number of important interactions.

For gene number 2 (POPTR_0012s14430) the pair of TFs found by RIR with the most significant difference between the performance of models with and without interaction term was "POPTR_0014s03590 POPTR_0012s03650". The model with interaction term has an $R^2_{prediction}$ of 0.98, while the model without has an $R^2_{prediction}$ of 0.91. Figure 4.15 plots the gene expression of POPTR_0012s14430 for two trees with the expression of this TF pair.

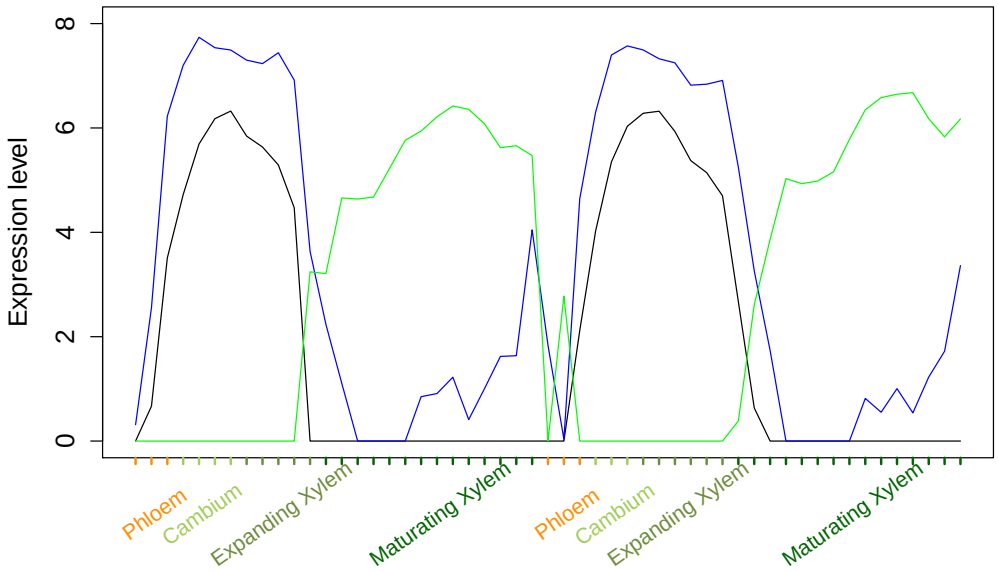


Figure 4.15: Expression level of the gene POPTR_0012s14430 (in black) and two "best" TFs found by RIR (in blue and green).

For gene number 5 (POPTR_0008s02680) the pair of TFs found by IR with the most significant difference between the performance of models with and without interaction term was "POPTR_0014s02530 POPTR_0002s04440". The model with interaction term has an $R^2_{prediction}$ of 0.88, while the model without has an $R^2_{prediction}$ of 0.75. Figure 4.16 plots the gene expression of POPTR_0008s02680 for two trees with the expression of this TF pair.

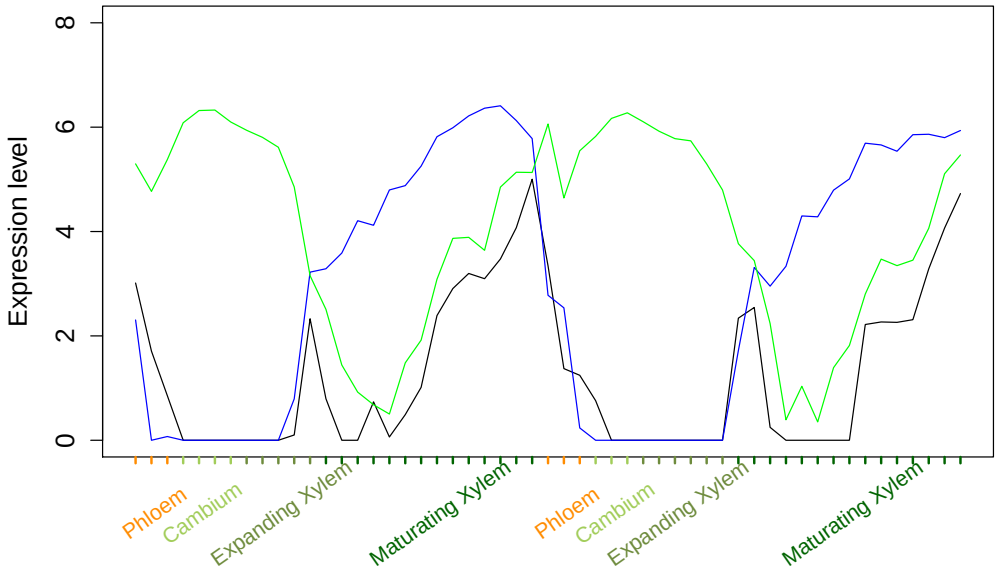


Figure 4.16: Expression level of the gene POPTR_0008s02680 (in black) and two "best" TFs found by IR (in blue and green).

Chapter 5

Discussion

Identifying the most relevant variables in a high-dimensional problem can be useful both for subsequent model construction and for further biological investigation. The methods proposed in this thesis seek to address this problem in situations where one assumes two-way interactions between the explanatory variables. The need to include interactions in statistical models can arise in many different fields, notably in biology, for example when considering gene expression. The expression of genes in living cells is controlled by transcription factors (TFs), which usually function in groups rather than individually (Chen et al., 2012).

Three methods for interaction detection are proposed. The methods produce a ranked list of all explanatory variables, or pairs of explanatory variables in the case of IR. The first method, PR, uses the score vectors from a polynomial kernel PLS model of second order to compute *pseudoloadings*. The explanatory variables are then scored with a distance measure based on these pseudoloadings. The IR method is an extension of common filtering methods for PLS (see Section 2.3.1), but applies them to an extended $\mathbf{X}$ ma-

trix with all the candidate two-way interaction terms in addition to the main effects. Unlike the two first methods, the third method, RIR, is randomized and scores the variables by repeatedly selecting subsets of variables and fitting a PLS model on the extended matrix of all main effects and two-way interaction terms for the subset.

The first section of this chapter describes some other methods related to the goals of this thesis. The next section discusses the findings from the simulation studies and some problems concerning the PR, IR and RIR methods. In the following section, the findings for the gene expression data are briefly discussed. In addition the methods used for evaluation are examined and some alternatives are proposed. The last section is a conclusion.

5.1 Related research

There exists a large literature on variable selection, but there are relatively few papers on variable selection involving interactions (Radchenko and James, 2010). As mentioned in the introduction (2.1), I distinguish between methods for variable ranking and variable selection. The methods treated in this thesis are variable ranking methods, and some alternative methods with similar goals will be described first. Then a few variable selection methods that could be used in combination with my methods will be mentioned.

Section 2.6 already describes SIS and SIRI. Variants of SIS are used to rank explanatory variables according to relevance and thus have the same goal as my methods. All SIS-based methods score the variables individually without explicitly considering the interaction terms, but if the ranking

statistic is well chosen, interaction are supposed to be included (for example in SIRI). In addition to ISIS (iterative SIS) and SIRI, DC-SIS is a SIS-based method that is able to select variables that are important through interactions. DC-SIS uses distance correlation to rank the explanatory variables (Li et al., 2012). Both SIRI and SIS are applicable to a broader set of non-linear situations than my methods. According to Jiang and Liu (2013), SIRI does not require any assumptions on the relationship between the response and the explanatory variables, and this relationship can for example be a rational model.

After using an effective variable ranking method to reduce the number of variables, a variable selection method can be applied to construct a model. An important group of variable selection methods are based on penalized regression, like LASSO (Tibshirani, 1996) or SCAD (Fan and Li, 2001). A few of these methods allow for interactions, for example hierNet (Bien et al., 2013) and VANISH (Radchenko and James, 2010). Both hierNet and VANISH are regularized regression methods with penalty functions that include interaction terms. Both methods also assume that the interactions belong to non-zero main effects (Radchenko and James, 2010). Other types of variable selection methods are forward-addition/backward-deletion methods like the aforementioned variable selection part of SIRI. There are also several variable selection methods for PLS (some are mentioned in Section 2.3 and in Mehmood et al. (2012)).

5.2 Simulation studies

All three methods (and SIRI) are investigated with simulations on the same scenarios. As expected all methods perform better in situations with fewer noise variables, more observations and higher information content (true R^2). The methods differ in their general performance and in their ability to find the relevant variables when the interaction term is large compared with the main effects.

The simulations demonstrate that the PR method is not suitable for identifying the relevant explanatory variables when the interaction between them is large and the main effects are small (see Figure 4.1). Contrary to expectations variables with important interaction terms did not get large pseudoloadings. The reason is that when the interaction term is dominant compared to the two main effects, the score vector $\mathbf{t}$ is nearly proportional to the interaction term. In that case it can be shown that the pseudoloadings of the relevant variables are approximately proportional to the covariances between the relevant variables and their interaction term

$$\mathbf{p}^* = \mathbf{X_0^T t} = \begin{bmatrix} \mathbf{x}_1^T\mathbf{t} \\ \mathbf{x}_2^T\mathbf{t} \end{bmatrix} \underset{\sim}{\propto} \begin{bmatrix} cov(x_1, x_1x_2) \\ cov(x_2, x_1x_2) \end{bmatrix}$$

According to the formulas (3.7) and (3.8) the covariances between the relevant variables and the interaction term are equal to 0 when the expectations of the explanatory variables is 0. This explains the low performance of PR in the simulation scenarios with large interaction, because the variables were constructed with expectations equal to 0. When the simulations are repeated with variables generated with non-zero expectations, the performance in the large-interaction scenarios is improved and the results are similar for the scenarios with small and large interactions. However this does not solve the

problem with the PR method, its ability to find large interactions is dependent on the relationship between the relevant variables and their interaction.

Unlike PR, the IR method performs better in scenarios with large interaction terms than in scenarios with small interactions (see Figure 4.12). In general the IR method has a good performance in the simulations, but the use of IR is limited by memory requirements: when $n = 100$ and $p = 1000$ the method requires handling of a 100×500500 matrix and the number columns increase rapidly with p. This fact and the running times in Table 4.2 reveal that although the IR method is much faster than the exhaustive search, the running time of the IR method still increases in a quadratic way according to p. This means that if the number of variables is multiplied by 10, the running time is multiplied by $10^2 = 100$. Because of the increasing time and memory requirements, the use of the IR method is dependent on effective filter methods for use in very high-dimensions.

The RIR method performs well for situations with both small and large interactions (Figure 4.12). The simulations indicate that the performance is dependent on three elements (see below). Examining these three elements, may clarify how the choice of arguments in the RIR method affect the performance. The performance depends on:

- The number of times the relevant variables are drawn together
- The number of times the variables are drawn together in an informative subset (of variables), *i.e.* that has a predictive performance greater than the performance of all the variables (see Formula (3.1))
- The number of times the relevant variables are given the highest score

among the variables in the subset, when they are drawn together in an informative subset.

The first element is a function of the total number of variables (p), the number of iterations (g) and the number of variables per iteration (v), as I demonstrated for the two-relevant-variables case in (3.12).

The second element is dependent on the information content (the relation between the response and the relevant variables), the number of variables per iteration and the filter threshold (see Section 3.1.3). If the predictive performance of all the variables is considerably larger than zero, the filter threshold will greatly affect the proportion of subsets that are scored (*i.e.* not discarded). A higher threshold will allow more subsets to be scored. With the constructed datasets the predictive performance of all the variables is often close to zero, so a subset of variables just has to have a predictive performance greater than zero in order to be scored (*i.e.* to avoid being filtered out). In these situations the filter threshold will not have much effect. Surprisingly, subsets containing both relevant variables can have a relatively low probability of getting an $R^2_{prediction}$ larger than zero. This probability decreases with lower information content, as expected, and with higher number of variables per iterations (Figure 5.1). Only the two scenarios with low information are represented in the figure, because this is not an issue when the information content is high.

Thus, a larger number of variables per iterations increases the risk of discarding subsets containing the relevant variables. This indicates that one should avoid choosing to many variables per iteration, but this may be dependent on the type of dataset.

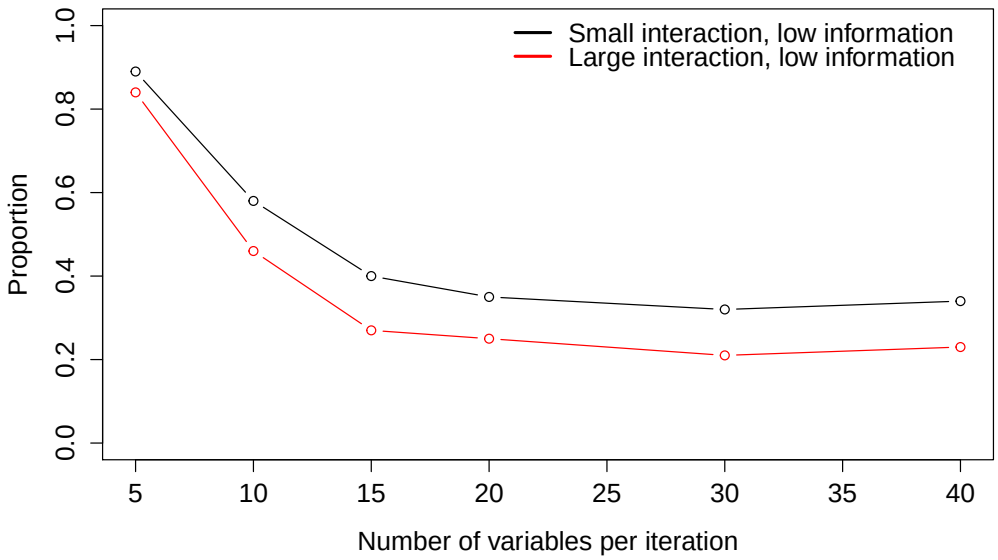


Figure 5.1: Proportion of experiments where subsets containing the relevant variables have a predictive performance larger than 0, evaluated for different subset sizes (number of variables per iteration). Only for scenarios with low information (1000 experiments for each scenario).

The last element also depends on the relationship between the the response and the relevant variables and the number of variables per iteration, and in addition the scoring method. The probability of the relevant variables being given the highest scores increases with increasing information content, larger interaction and smaller number of variables per iterations.

To sum up, more variables per iteration allows us to explore the variable-space faster, but increases the risk of hiding the relevant variables in noise when they are drawn together. The optimal choice of number of variables per iteration probably depends on the type of data, but a number close to

10 seems reasonable for many situations. The choice of filter threshold is not very important if the predictive performance of all the variables is close to zero. However, if a model with all the explanatory variables has a high predictive performance, as is the case with the gene expression dataset used in this thesis, the threshold should be chosen with care. A larger threshold leads to scoring of more variable subsets and increased running time, whereas a small threshold in combination with a large number of variables per iteration may result in the discarding of *all* variable subsets.

Like IR, RIR is faster than an exhaustive search, but the running time can be considered to increase in a quadratic way. The running time for RIR is dependent on the number of iterations, but when keeping a constant probability of drawing the relevant variables together, the number of iterations has to increase with p according to (3.12). This function is limited by a quadratic function, and the RIR method can therefore be considered to run in quadratic time.

Both the IR and RIR methods performs better than SIRI in these simulation studies. SIRI seems to perform worse than expected compared with the simulations in Jiang and Liu (2013). This can be explained by the relative low number of observations used in my simulation studies. SIRI scores the variables by evaluating their conditional variance within different slices of the response (see Section 2.6). The performance increases with a larger number of slices, but only if there are 40 or more observations in each slice (Jiang and Liu, 2013). With only 50 and 100 observations, a large number of slices with many observations in each slice can not be obtained, and this probably reduces the performance. Unlike IR and RIR, the running time of SIRI in-

creases linearly with the number of variables (Jiang and Liu, 2013). This is a very attractive feature for a variable ranking method for high-dimensions.

5.3 Gene expression data

The two most successful methods from the simulation studies, IR and RIR, were applied to 10 genes from the gene expression dataset of *Populus tremula* (see Section 4.6). Both methods were used with simple β-scoring, as T-scoring is slower and did not significantly improve the results in the simulations.

Both methods identified transcription factors (TFs) with important interactions, for example in the Figures 4.15 and 4.16. Figure 4.15 show how the gene expression follows the expression of the first TF, but is repressed when the other TF is expressed. In Figure 4.16 the gene is only expressed when both TFs also are. Two runs of RIR produced quite similar results, indicating that the ranking is quite consistent from run to run. Although there was some overlap between the top ranked TF from IR and RIR, the methods did not identify the same "important" interactions (as defined in Section 3.2.1). Since the truth is not known for real data applications, two evaluation methods were proposed in Section 3.2.1.

The first evaluation method compares the predictive performance of the ten highest ranked TFs with the predictive performance of random subsets of TFs. IR obtained a significant p-value for six genes and RIR for only four. However this evaluation method might not be suitable considering the goal of the methods. As stated several times, the methods only rank all the

variables according to relevance, and do not claim to identify the subset of variables that predicts the response in the best possible way.

The second evaluation method assesses whether the methods find more "important" two-way interaction than than would be expected by chance. IR had six genes with a significant number of important interactions, while RIR had five. Both the usefulness of this evaluation method and the definition of "important" interactions can be discussed. In Section 3.2.1 important interactions were defined as interactions that made the predictive performance of the model significantly better compared to the model with only the two main effects:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 \qquad \text{compared to}$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Alternatively I could have defined it as interactions were the model with two main effects and interactions is significantly better than the best model with only one main effect:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 \qquad \text{compared to}$$

$$\text{The best of: } y = \beta_0 + \beta_1 x_1 \quad \text{or} \quad y = \beta_0 + \beta_2 x_2$$

Yet another possibility would have been to define "important" interactions as significant interaction (in the linear model). These alternatives would probably have yielded slightly different results.

More importantly, the usefulness of this evaluation method may also be questioned. Is the identification of a large number of "important interactions" really an advantage for the method? Some of the genes may only have a few really relevant TFs. A better solution might be to compare the TFs

found with the IR and RIR methods, with the TFs identified with an exhaustive search. At least the notion of important interactions should be combined with a sufficiently large predictive performance: pairs of TF with important interaction, but low predictive performance are probably not very interesting.

Independent biological evidence can be used to examine whether the top ranked TFs really are involved in the regulation of the specific gene. Several different relevant experiments exists, but there is not much such data for *Populus*. Two examples, knockout experiments and protein-protein interactions, will be described here. In knockout experiments one observes whether the expression of a specific gene is significantly changed when TFs or pairs of TFs are knocked out (Awad and Chen, 2014). If there really exists an interaction between the TFs, the expression of the gene should be significantly different when only one of the TFs is knocked out compared to when both are.

In some cases interacting TFs bind on different parts of the genes promotor region and do not physically interact. While in other cases TF interactions are direct, physical protein-protein interactions where the two TFs for example form a complex which binds to the TF binding site of the gene (Chen et al., 2012). The last kind of TFs, can be validated by experiments that detect protein-protein interactions.

There exists several specialized methods for analysing gene expression data and uncover TF interactions. Several methods use a network approach, one recent example is the mTRIM algorithm which uses the concept of activation time-points and logical roles in order to model interactions between TFs (Chen et al., 2012). Rather than assuming that the expression of impor-

tant TFs correlates with the gene expression over the entire time course (or developmental gradient) as I do, the authors assert that one should match TFs with genes only at specific activation time-points. Unlike the methods of this thesis, mTRIM also infers the logical roles of the TFs, whether they are activators or repressors and whether the TF is necessary, sufficient or necessary and sufficient for the gene (Chen et al., 2012).

5.4 Conclusion

Three methods for variable ranking and interaction detection are investigated in this thesis. The PR method is based on a kernel PLS model and was unsuccessful in finding important interactions. It remains an open question whether the information from kernel PLS can be used for variable ranking in such situations. The two other methods are based on the regular PLS algorithm, and they both outperform a competing method (SIRI) in the simulation studies. However use of the IR method in high-dimensions is limited because of memory requirements, and would require an efficient pre-filtering method. SIRI is a possible choice, but is as mentioned unreliable in situations with relatively few observations.

The RIR method seems the most promising method, despite its rather long and quadratically increasing running time. The method can be extended to higher order interactions, but this would probably increase the running time a lot. The method could also be combined with other methods. Both with pre-filtering methods (like SIRI) or with variable selection methods if the goal is model construction. In future work, the methods should be applied to datasets where there are possibilities for biological validation of the findings, see Section 5.3 for some suggestions.

Bibliography

A. Agresti. *An introduction to categorical data analysis*, volume 423. John Wiley & Sons, 2007.

S. Awad and J. Chen. Inferring transcription factor collaborations in gene regulatory networks. *BMC Systems Biology*, 8(Suppl 1):S1, 2014.

R. Bhalerao, O. Nilsson, and G. Sandberg. Out of the woods: forest biotechnology enters the genomic era. *Current Opinion in Biotechnology*, 14(2): 206–213, 2003.

P. J. Bickel and K. A. Doksum. *Mathematical statistics, Volume 1: Basic and selected topics (Second (updated printing 2007) of the Holden-Day 1976 ed.)*. Pearson Prentice Hall, Pearson Education, Inc., Upper Saddle River, NJ, 2001.

J. Bien, J. Taylor, R. Tibshirani, et al. A lasso for hierarchical interactions. *The Annals of Statistics*, 41(3):1111–1141, 2013.

M.-J. M. Chen, L.-C. Chou, T.-T. Hsieh, D.-D. Lee, K.-W. Liu, C.-Y. Yu, Y.-J. Oyang, H.-K. Tsai, and C.-Y. Chen. De novo motif discovery facilitates identification of interactions between transcription factors in saccharomyces cerevisiae. *Bioinformatics*, 28(5):701–708, 2012.

H. Chun and S. Keleş. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):3–25, 2010.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96 (456):1348–1360, 2001.

J. Fan and J. Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

I. S. Helland. On the structure of partial least squares regression. *Communications in Statistics - Simulation and Computation*, 17(2):581–607, 1988.

M. Hertzberg, H. Aspeborg, J. Schrader, A. Andersson, R. Erlandsson, K. Blomqvist, R. Bhalerao, M. Uhlén, T. T. Teeri, J. Lundeberg, et al. A transcriptional roadmap to wood formation. *Proceedings of the National Academy of Sciences*, 98(25):14732–14737, 2001.

G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*. Springer, 2013.

B. Jiang and J. S. Liu. Sliced inverse regression with variable selection and interaction detection. *arXiv preprint arXiv:1304.4056*, 2013.

İ. Karaman, E. M. Qannari, H. Martens, M. S. Hedemann, K. E. B. Knudsen, and A. Kohler. Comparison of sparse and jack-knife partial least squares

regression methods for variable selection. *Chemometrics and Intelligent Laboratory Systems*, 122:65–77, 2013.

R. Li, W. Zhong, and L. Zhu. Feature screening via distance correlation learning. *Journal of the American Statistical Association*, 107(499):1129–1139, 2012.

F. Lindgren, P. Geladi, and S. Wold. The kernel algorithm for pls. *Journal of Chemometrics*, 7(1):45–59, 1993.

T. Mehmood, K. H. Liland, L. Snipen, and S. Sæbø. A review of variable selection methods in partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 2012.

I. Miller, M. Miller, and E. John. *Freund's mathematical statistics with applications*. Pearson Prentice Hall, Pearson Education, Inc., Upper Saddle River, NJ, 2004.

P. Radchenko and G. M. James. Variable selection using adaptive nonlinear interaction structures in high dimensions. *Journal of the American Statistical Association*, 105(492):1541–1553, 2010.

S. Rännar, F. Lindgren, P. Geladi, and S. Wold. A pls kernel algorithm for data sets with many variables and fewer objects. part 1: Theory and algorithm. *Journal of Chemometrics*, 8(2):111–125, 1994.

P. H. Raven, R. F. Evert, and S. E. Eichhorn. *Biology of plants*. Macmillan, 2005.

A. C. Rencher and G. B. Schaalje. *Linear models in statistics*. John Wiley & Sons, 2008.

R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. In C. Saunders, M. Grobelnik, S. Gunn, and J. Shawe-Taylor, editors, *Subspace, Latent Structure and Feature Selection*, volume 3940 of *Lecture Notes in Computer Science*, pages 34–51. Springer Berlin Heidelberg, 2006.

R. Rosipal and L. J. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *The Journal of Machine Learning Research*, 2:97–123, 2001.

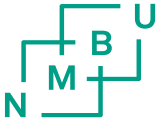
M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 111–147, 1974.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

J. W. Tukey. Bias and confidence in not-quite large samples. In *Annals of Mathematical Statistics*, volume 29, pages 614–614, 1958.

E. J. Williams. The comparison of regression variables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 396–399, 1959.

H. Wold. Soft modeling by latent variables; the nonlinear iterative partial least squares approach. In J. Gani, editor, *Perspectives in Probability and Statistics: papers in honour of M. S. Bartlett on the occasion of his sixty-fifth birthday*, pages 520–540. Academic Press, 1975.



Norwegian University
of Life Sciences

Postboks 5003
NO-1432 Ås, Norway
+47 67 23 00 00
www.nmbu.no