Simulated Data for Linear Regression with Structured and Sparse Penalties

Tommy Löfstedt*[1,†], Vincent Guillemot[1,†], Vincent Frouin[1], Edouard Duchesnay[1], and Fouad Hadj-Selem[1,†]

[1]Brainomics Team, Neurospin, CEA Saclay, 91190 Gif sur Yvette – France.
[†]These authors contributed equally to this work.

Abstract

A very active field of research in Bioinformatics is to integrate structure in Machine Learning methods. Methods recently developed claim that they allow simultaneously to link the computed model to the graphical structure of the data set and to select a handful of important features in the analysis.

However, there is still no way to simulate data for which we can separate the three properties that such method claim to achieve. These properties are:

(i) the sparsity of the solution, *i.e.*, the fact the the model is based on a few features of the data;

(ii) the structure of the model;

(iii) the relation between the structure of the model and the graphical model behind the generation of the data.

We propose a framework to simulate data for linear regression in which we control: The signal-to-noise ratio, the internal correlation structure of the data and the optimisation problem that they are a solution of. Also, we make no statistical assumptions on the distribution of the data set.

1 Introduction

Simulated data are widely used to assess optimisation methods. This is because of their ability to evaluate certain aspects of the methods under study, that are impossible to look into when using real data sets. In the context of convex optimisation, it is never possible to know the exact solution of the minimisation problem with real data and it proves to be a difficult problem even with simulated data. We propose to generalise an approach originally proposed by Nesterov [4], for LASSO regression, to a broader family of penalised regressions.

We would thus like to generate simulated data for which we know the exact solution of the optimised function. The inputs are: The minimiser β^*, a candidate data set X_0 $(n \times p)$, residual vector ε, regularisation parameters (in our case they are two: κ and γ), the signal-to-noise ratio σ, and the expression of the function $f(\beta)$ to minimize.

The candidate version of the dataset may for instance be $X_0 \sim N(0, \Sigma)$, and the residual vector may be $e \sim N(0, 1)$.

*lofstedt.tommy@gmail.com

The proposed procedure outputs X and y such that

$$\beta^* = \arg\min_{\beta} f(\beta),$$

with f a convex function depending on the data set X and the outcome y.

2 Background

In this section we will present the context of linear regression, with complex penalties and a first algorithm presented by Nesterov [3]. We finish by introducing the properties of the simulated data that a user would like to control.

2.1 Linear Regression

We place ourselves in the context of linear regression models. Let $X \in \mathbb{R}^{n\times p}$ be a matrix of n samples, where each sample lies in a p-dimensional space; and let $y \in \mathbb{R}^n$ denote the n-dimensional response vector. In the linear regression model $y = X\beta + e$, where e is an additive noise vector and β represents the unknown p-vector that contains the regression coefficients. This statistical model explains the variability in the dependent variable, y, as a function of the independent variables, X. The model parameters are calculated so as to minimise the classical least squares loss. The value of β that minimises the sum of squared residuals, that is $\frac{1}{2}\|X\beta - y\|_2^2$, is called the Ordinary Least Squares (OLS) estimator for β.

We provide in the following paragraphs the mathematical definitions and notations that will be used throughout this paper. First, we denote by $\|\cdot\|_q$ the standard ℓ_q-norm defined on $\mathbb{R}^p$ by

$$\|x\|_q := \left(\sum_{j=1}^{p} x_j^q\right)^{\frac{1}{q}},$$

with dual norm $\|\cdot\|_{q'}$.

For a smooth real function f on $\mathbb{R}^p$, we denote by $\nabla f(\beta)$ the gradient vector $(\partial_1 f(\beta), \ldots, \partial_p f(\beta))$. However, many functions that arise in practice may be non-differentiable at certain points. A common example is the ℓ_1-norm. In that case, the generalisation of the gradient for a non-differentiable convex functions leads naturally to the following definition of the subgradient.

Definition 2.1. *A vector $v \in \mathbb{R}^p$ is a subgradient of a convex function f: $\mathrm{dom}(f) \subseteq \mathbb{R}^p \to \mathbb{R}$ at β if*

$$f(y) \geq f(\beta) + \langle v | y - \beta \rangle, \tag{1}$$

for all $y \in \mathrm{dom}(f)$. The set of all subgradients at β is called the subdifferential, and is denoted by $\partial f(\beta)$.

2.2 LASSO

The function

$$f(\beta) = \frac{1}{2}\|X\beta - y\|_2^2 + \kappa\|\beta\|_1$$

is known as the LASSO problem.

Nesterov [4] addressed how to simulate data for this case, and we will therefore not go into details. Instead we will simply adapt it to our notation and explain some steps that are not obvious.

The principle behind Nesterov's idea is as follows: First, define the error to be $\varepsilon = X\beta - y$, in the model between $X\beta$ and y, such that it is independent from β^*. Then, select acceptable values for the columns of X such that zero belongs to the sub-gradient of f at point β^*, with the subgradient

$$\begin{aligned}\partial f(\beta) &= X^\top (X\beta - y) + \kappa \partial \|\beta\|_1 \\ &= X^\top \varepsilon + \kappa \partial \|\beta\|_1.\end{aligned}$$

At β^* we have such that

$$0 \in X^\top \varepsilon + \kappa \partial \|\beta^*\|_1, \tag{2}$$

and we stress again that that $X^\top \varepsilon$ does not depend on β^*. We distinguish two cases:

First case: We consider a variable $\beta_i^* \neq 0$, the ith element of β^*. With $\beta_i^* \neq 0$ it follows that $\partial |\beta_i^*| = \text{sign}(\beta_i^*)$, and thus that

$$0 = X_i^\top \varepsilon + \kappa \,\text{sign}(\beta_i^*)$$

follows because of Eq. (2), with X_i the ith column of X.

Second case: We consider the case when $\beta_i^* = 0$. We note that the subgradient of $|\beta_i^*|$ when $\beta_i^* = 0$ is

$$\partial |\beta_i^*| \in [-1, 1],$$

and thus from Eq. (2) we see that

$$0 \in X_i^\top \varepsilon + \kappa[-1, 1]. \tag{3}$$

Solution

The candidate matrix X_0 will serve as a first un-scaled version of X and in fact we have such that $X_i = \omega_i X_{0,i}$, for all $1 \leq i \leq p$.

If $\beta_i^* \neq 0$, then $X_i^\top \varepsilon + \kappa \,\text{sign}(\beta_i^*) = 0$ and thus

$$X_i^\top \varepsilon = -\kappa \,\text{sign}(\beta_i^*)$$

and since $X_i = \omega_i X_{0,i}$ we have

$$\omega_i = \frac{-\kappa \,\text{sign}(\beta_i^*)}{X_{0,i}^\top \varepsilon}.$$

If $\beta_i^* = 0$, we use Eq. (3) and have

$$X_i^\top \varepsilon \in \kappa[-1, 1].$$

Thus, with $X_i = \omega_i X_{0,i}$ we obtain

$$\omega_i X_{0,i}^\top \varepsilon \in \kappa[-1, 1],$$

or equivalently

$$\omega_i \sim \frac{\kappa \mathcal{U}(-1, 1)}{X_{0,i}^\top \varepsilon},$$

Once X is generated, we let $y = X\beta^* - \varepsilon$.

3 Method

The objective is to generate X and y such that

$$\beta^* = \arg\min_{\beta} \frac{1}{2}\|X\beta - y\|_2^2 + P(\beta),$$

where P is a penalty that can be expressed on the form

$$P(\beta) = \sum_{\pi \in \Pi} \kappa_\pi \pi(\beta),$$

in which Π is the set of all penalties, π. This is a general notation to represent the fact that we have many different penalties; κ_π is the regularisation parameter of penalty π, and $\partial\pi(\beta^*)$ is the subgradient of penalty π at β^*.

Thus, if you know the subgradient of each penalty, the general solution can be written on the form

$$\omega_i = \frac{\sum_{\pi \in \Pi} -\kappa_\pi \partial\pi(\beta^*)}{X_0^\top \varepsilon}.$$

The penalties that we consider in this work are

$$P(\beta) = \frac{\kappa_{\text{Ridge}}}{2}\|\beta\|_2^2 + \kappa_{\ell_1}\|\beta\|_1 + \kappa_{TV}\text{TV}(\beta),$$

and, as we show below, we know their subgradients.

3.1 Subgradient of Complex Penalties

The complex penalties that we consider in this work can be written on the form

$$\pi(\beta) = \sum_{g=1}^{G} \|A_g\beta\|_q. \tag{4}$$

We will in this work only be interested in the case when $q = q' = 2$, i.e. the Euclidean norm. This is the case when π is e.g. the Total Variation constraint [5] or Group LASSO [6].

We need the following two Lemmas in order to derive the subgradient of this complex penalty.

Lemma 3.1 (Subgradient of the sum)**.** *If f_1 and f_2 are convex functions, then*

$$\partial(f_1 + f_2) = \partial f_1 + \partial f_2.$$

Proof. See [2]. □

Lemma 3.2 (Subgradient of the composition)**.** *If f and $g(x) = Ax$ are linear functions, then*

$$\partial(g \circ f)(\beta) = A^\top \partial f(A\beta).$$

Proof. See [2]. □

These Lemmas play a central role in the following theorem that details the structure of the subgradient of π.

Theorem 3.3 (Subgradient of π)**.** *If π has the form given in Eq.* (4)*, then*

$$\partial\pi(\beta) = A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_G\beta\|_2 \end{bmatrix}.$$

Proof.

$$\begin{aligned}
\partial\pi(\beta) &= \partial\left(\sum_{g=1}^{G}\|A_g\beta\|_2\right) \\
&= \sum_{g=1}^{G}\partial\|A_g\beta\|_2 && \text{(Using Lemma 3.1)} \\
&= \sum_{g=1}^{G}A_g^\top\partial\|A_g\beta\|_2 && \text{(Using Lemma 3.2)} \\
&= A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_G\beta\|_2 \end{bmatrix}.
\end{aligned}$$

□

Before we show the application to some actual penalties we will mention that the subgradient of the 2-norm is

$$\partial\|x\|_2 = \begin{cases} \frac{x}{\|x\|_2} & \text{if } \|x\|_2 > 0, \\ \{y \mid \|y\|_2 \leq 1\} & \text{if } \|x\|_2 = 0. \end{cases} \tag{5}$$

3.2 Algorithm

In this section, we detail the algorithm that generates a simulated dataset that is the solutions to a complex optimisation problem.

Algorithm 1 Simulate dataset

Require: β^*, X_0, ε
Ensure: X, y and β^* such that $\beta^* = \arg\min f(\beta)$
1: **for** $\pi \in \Pi$ **do**
2: Generate $r_\pi \in \partial\pi(\beta^*)$
3: **end for**
4: **for** $i = 1, \ldots, p$ **do**
5: $\omega_i = \frac{-\sum_{\pi\in\Pi}\kappa_\pi r_{\pi,i}}{X_{0,i}^\top e}$
6: $X_i = \omega_i X_{0,i}$
7: **end for**
8: $y = X\beta^* - e$

4 Application

We apply the aforementioned algorithm to generate a data set and associate it to the exact solution of a linear regression problem with Elastic Net and Total variation, TV, penalties. Since Algorithm 1 requires the computation of an element of the subgradient for each penalty, we first focus on detailing the subgradient of TV.

4.1 Total Variation

The TV constraint, for a discrete β, is defined as

$$\mathrm{TV}(\beta) = \sum_{i=1}^{p} \|\nabla\beta_i\|_2, \tag{6}$$

where $\nabla\beta_i$ is the gradient at point β_i. Since β is not continuous, the TV constraint needs to be approximated. It is usually approximated using the forward difference, i.e. such that

$$\begin{aligned}\mathrm{TV}(\beta) &= \sum_{i=1}^{p} \|\nabla\beta_i\|_2 \\ &\approx \sum_{i_1=1}^{p_1-1} \cdots \sum_{i_D=1}^{p_D-1} \sqrt{(\beta_{i_1+1,\ldots,i_D} - \beta_{i_1,\ldots,i_D})^2 + \cdots + (\beta_{i_1,\ldots,i_D+1} - \beta_{i_1,\ldots,i_D})^2},\end{aligned}$$

where p_i is the number of variables in the ith dimension, for $i = 1, \ldots, D$, with D dimensions.

We will first illustrate this in the 1-dimensional case. In this case $\|x\|_2 = \sqrt{x^2} = |x|$, since $x \in \mathbb{R}$. Thus we have

$$\mathrm{TV}(\beta) \approx \sum_{i=1}^{p-1} |\beta_{i+1} - \beta_i|,$$

We note that if we define

$$A = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & & \ddots & \ddots & & \vdots \\ 0 & 0 & \cdots & 0 & -1 & 1 \\ 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix},$$

then

$$\mathrm{TV}(\beta) \approx \sum_{i=1}^{p-1} |\beta_{i+1} - \beta_i| = \sum_{g=1}^{G} \|A_g\beta\|_2,$$

where $G = p$ and A_g is the gth row of A.

Thus, we use Theorem 3.3 and obtain

$$\partial TV(\beta) = A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_G\beta\|_2 \end{bmatrix} = A^\top \begin{bmatrix} \partial\,|\beta_2 - \beta_1| \\ \vdots \\ \partial\,|\beta_G - \beta_{G-1}| \end{bmatrix},$$

in which we use Eq. (5) and obtain that

$$\partial|x| = \begin{cases} \mathrm{sign}(x) & \text{if } |x| > 0, \\ [-1, 1] & \text{if } |x| = 0. \end{cases}$$

The general case will be illustrated with a small example using a 3-dimensional image. A 24-dimensional regression vector β is generated, that represents a $2 \times 3 \times 4$ image. The image, with linear indices indicated, is

1st layer:

1	2	3	4
5	6	7	8
9	10	11	12

2nd layer:

13	14	15	16
17	18	19	20
21	22	23	24

We note, when using the linear indices, that β_1 and β_2 are neighbours in the 1st dimension, that β_1 and β_5 are neighbours in the 2nd dimension and that β_1 and β_{13} are neighbours in the 3rd dimension. Using 3-dimensional indices, i.e. such that $\beta_{i,j,k}$, the penalty becomes

$$TV(\beta) \approx \sum_{i=1}^{p_1-1}\sum_{j=1}^{p_2-1}\sum_{k=1}^{p_3-1}\sqrt{(\beta_{i+1,j,k}-\beta_{i,j,k})^2+(\beta_{i,j+1,k}-\beta_{i,j,k})^2+(\beta_{i,j,k+1}-\beta_{i,j,k})^2},$$

in which $p_1 = 4$, $p_2 = 3$ and $p_3 = 2$.

We thus construct the A matrix to reflect this penalty. The first group will be

$$A_1 = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

Thus, for group A_i we will have a -1 in the ith column in all dimensions, a 1 in the $i+1$th column for the 1st dimension, a 1 in the (p_1+i)th column for the 2nd dimension and a 1 in the $(p_1 \cdot p_2 + i)$th column for the 3rd dimension. Note that when these indices fall outside of the A matrix (i.e., the indices are greater than p_1, p_2 or p_3, respectively) then the whole row (but not the group!) must be set to zero (or handled in some other way not specified here).

We thus obtain

$$\partial TV(\beta) = A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_{24}\beta\|_2 \end{bmatrix}. \tag{7}$$

We use Eq. (5) and obtain

$$\partial\|A_i\beta\|_2 = \begin{cases} \frac{A_i\beta}{\|A_i\beta\|_2} & \text{if } \|A_i\beta\|_2 > 0, \\ \frac{\alpha u}{\|u\|_2},\ \alpha \sim \mathcal{U}(0,1), u \sim \mathcal{U}(-1,1)^3 & \text{if } \|A_i\beta\|_2 = 0. \end{cases} \tag{8}$$

We note that A is very sparse, which greatly helps to speed up the implementation.

4.2 Linear regression with Elastic Net and Total Variation penalties

We will here give an example with Elastic Net and Total Variation penalties. The function we are working with is

$$f(\beta) = \frac{1}{2}\|X\beta - y\|_2^2 + \frac{1-\kappa}{2}\|\beta^*\|_2^2 + \kappa\|\beta^*\|_1 + \gamma\,\mathrm{TV}(\beta).$$

The subgradient in this case is

$$0 \in X^\top\varepsilon + (1-\kappa)\beta + \kappa\partial\|\beta^*\|_1 + \gamma\partial\,\mathrm{TV}(\beta^*). \tag{9}$$

We rearrange like for the LASSO and note that we seek

$$X_i^\top\varepsilon = -(1-\kappa)\beta_i^* - \kappa\partial|\beta_i^*| - \gamma(\partial\,\mathrm{TV}(\beta))_i$$

and further, since $X_i = \omega_i X_{0,_i}$, that

$$\omega_i = \frac{-(1-\kappa)\beta_i^* - \kappa\partial|\beta_i^*| - \gamma(\partial\,\mathrm{TV}(\beta))_i}{X_{0,i}^\top\varepsilon}$$

We note that in the case when $\beta_i^* = 0$, adding the smooth Ridge constraint to the LASSO has no effect.

We use Theorem 3.3 and obtain a subdifferential that contains zero

$$0 \in X^\top \varepsilon + (1-\kappa)\beta + \kappa \partial \|\beta^*\|_1 + \gamma A^\top \begin{bmatrix} \partial \|A_1 \beta\|_2 \\ \vdots \\ \partial \|A_G \beta\|_2 \end{bmatrix}.$$

With the subgradient of TV defined using Eq. (7) and Eq. (8), we obtain

$$\omega_i = \frac{-(1-\kappa)\beta_i^* - \kappa \partial |\beta_i^*| - \gamma \left(A^\top \begin{bmatrix} \partial \|A_1 \beta\|_2 \\ \vdots \\ \partial \|A_G \beta\|_2 \end{bmatrix} \right)_i}{X_{0,i}^\top \varepsilon},$$

for each variable $i = 1, \ldots, p$ and with $X_i = \omega_i X_{0,_i}$. By $(\cdot)_i$, we denote the ith variable of the vector within parentheses. We also remember from Section 2.2 that $\partial|x|$ is $\text{sign}(x)$ if $x \neq 0$ and $x \in [-1, 1]$ if $x = 0$; thus, if $x = 0$, we may choose $x \sim \mathcal{U}(-1, 1)$.

4.3 SNR and correlation

We use the same definition of signal-to-noise ratio as in [1], namely

$$\text{SNR} = \frac{\|X(\beta)\beta\|_2}{\|e\|_2},$$

where $X(\beta)$ is the data generated from β when using the simulation process described above.

With this definition of signal-to-noise ratio, and with the definition of the simulated data given above we may scale the regression vector such that

$$\text{SNR}(a) = \frac{\|X(\beta a)\beta a\|_2}{\|e\|_2}. \tag{10}$$

If the user provides a desired signal-to-noise ratio, σ, it is reasonable to ask if we are able to find an a such that $\text{SNR}(a) = \sigma$. We have the following theorem.

Theorem 4.1. *Using the definition of simulated data described above, and with the definition of signal-to-noise ratio in Eq.* (10) *there exists an $a > 0$ such that*

$$SNR(a) = \sigma, \tag{11}$$

for $\sigma > 0$.

Proof. We rephrase the signal-to-noise ratio as

$$\|X(\beta a)\beta a\|_2 = \sigma \|e\|_2,$$

and square both sides to get

$$\|X(\beta a)\beta a\|_2^2 = \sigma^2 \|e\|_2^2 = s.$$

We let X_i be the ith column of X, we remember that $X_i = \omega_i X_{0,i}$, and let β_i be the ith element of β. The left-hand side is written

$$\begin{aligned}\|X(\beta a)\beta a\|_2^2 &= \left(\sum_{i=1}^{p} X_i\beta_i a\right)^\top \left(\sum_{i=1}^{p} X_i\beta_i a\right) \\ &= \sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^2 X_i^\top X_j \beta_i\beta_j + \sum_{i=1}^{p} a^2 X_i^\top X_i \beta_i^2 \\ &= \sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^2 X_{0,i}^\top X_{0,j} \beta_i\beta_j\omega_i\omega_j + \sum_{i=1}^{p} a^2 X_i^\top X_i \beta_i^2\omega_i^2.\end{aligned} \tag{12}$$

If we add all the constraint rescribed above, i.e. ℓ_1, ℓ_2 and TV, we have that

$$\omega_i = \frac{-\lambda\partial|\beta_i| - a\kappa\beta_i - \gamma\left(A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_G\beta\|_2 \end{bmatrix}\right)_i}{X_{0,i}^\top\varepsilon},$$

and we may thus write

$$\omega_i = k_i a + m_i,$$

with

$$k_i = \frac{-\kappa\beta_i}{X_{0,i}^\top\varepsilon}$$

and

$$m_i = \frac{-\lambda\partial|\beta_i| - \gamma\left(A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_G\beta\|_2 \end{bmatrix}\right)_i}{X_{0,i}^\top\varepsilon}.$$

We continue to expand Eq. (12) and get

$$\begin{aligned}&\sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^2 X_{0,i}^\top X_{0,j}\beta_i\beta_j\omega_i\omega_j + \sum_{i=1}^{p} a^2 X_i^\top X_i\beta_i^2\omega_i^2 \\ &\quad= \sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^2 X_{0,i}^\top X_{0,j}\beta_i\beta_j(k_i a + m_i)(k_j a + m_j) + \sum_{i=1}^{p} a^2 X_i^\top X_i\beta_i^2(k_i a + m_i)^2 \\ &\quad= \sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^2 \underbrace{X_{0,i}^\top X_{0,j}\beta_i\beta_j}_{d_{i,j}}(a^2 k_i k_j + a k_i m_j + a m_i k_j + m_i m_j) \\ &\qquad+ \sum_{i=1}^{p} a^2 \underbrace{X_i^\top X_i\beta_i^2}_{d_{i,i}}(a^2 k_i^2 + a 2 k_i m_i + m_i^2). \\ &\quad= \sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^4 d_{i,j} k_i k_j + a^3 d_{i,j}(k_i m_j + m_i k_j) + a^2 d_{i,j} m_i m_j \\ &\qquad+ \sum_{i=1}^{p} a^4 d_{i,i} k_i^2 + a^3 2 d_{i,i} k_i m_i + a^2 d_{i,i} m_i^2.\end{aligned}$$

We note that this is a fourth order polynomial and write it on the generic form

$$\|X(\beta a)\beta a\|_2^2 = Aa^4 + Ba^3 + Ca^2.$$

Now, since we seek a solution $a > 0$ such that $\|X(\beta a)\beta a\|_2^2 = s$, we seek positive roots of the quartic equation

$$Aa^4 + Ba^3 + Ca^2 - s = 0. \tag{13}$$

This fourth order polynomial has a minimum of $-s$ at $a = 0$, also Eq. (10) is positive for all values of a, and tends to infinity when a tends to infinity. Thus, by the intermediate value theorem there is a value of a for which $\|X(\beta a)\beta a\|_2^2 - s = 0$ and thus also that $\text{SNR}(a) = \sigma$. □

We may use Eq. (13) above to find the roots of this fourth order polynomial analytically. This may, however, be tedious because of the many terms of the function. Instead, because of the above theorem, we know that we can sucessfully apply the Bisection method to find a root of this function. The authors have tested this sucessfully, even with larger datasets. Also, we may use either root, if there are more than one, since they all give $\text{SNR}(a) = \sigma$.

Thus, to control the signal-to-noise ratio, we would encapsulate Algorithm 1 in a Bisection loop in order to control the signal-to-noise ratio.

Also, we control the correlation structure of X_0 by e.g. letting $X_0 \sim \mathcal{N}(0, \Sigma)$. Also, we let $X_i = \omega_i X_{0,i}$ for all $1 \leq i \leq p$. It then follows that $\text{cor}(X_l, X_m) = \text{cor}(\omega_l X_{0,l}, \omega_m X_{0,m})$.

5 Example

The main benefit of generating the data like this is that we know everything about our data. In particular, we know the true minimiser, β^*, and we know the Lagrange multipliers κ and γ. There is no need to use *e.g.* cross-validation to find any parameters, and we know directly if the $\beta^{(k)}$ that our minimising algorithm found is close to the true β^* or not.

We illustrate this main point by a small simulation, in which we vary κ and γ in an interval around their "true" values and compute $f(\beta^{(k)}) - f(\beta^*)$ for each of these values. The result is shown in Figure 1, and we see that the solution that gives the smallest function value is at precisely the true values of κ and γ.

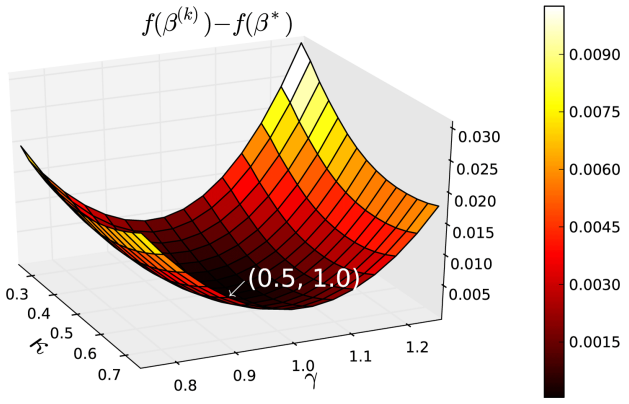

Figure 1: An illustration of the benefit of using the simulated data described in this section. The minimum solution is found when using the regularisation parameters used in the construction of the simulated data. These (25×36) data had no correlation between variables and the following characteristics: 50 % sparsity, signal-to-noise ratio 100, $\kappa = 0.5$, $\gamma = 1.0$.

References

[1] Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Convex optimization with sparsity-inducing norms. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.

[2] J. Frédéric Bonnans, Jean Charles Gilbert, and Claude Lemarechal. *Numerical Optimization: Theoretical and Practical Aspects*. Springer-Verlag Berlin and Heidelberg GmbH & Co. K, 2nd edition, 2006.

[3] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, December 2004.

[4] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[5] L Rudin, S Osher, E Fatemi, and Santa Monica. Nonlinear total variation based noise removal algorithms * f u = f Uo. *Physica D: Nonlinear Phenomena*, 60(1-4):259–268, 1992.

[6] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, February 2006.