Simulated Data for Linear Regression with Structured and Sparse Penalties

Tommy Lofstedt, Vincent Guillemot, Vincent Frouin, Edouard Duchesnay, Fouad Hadj-Selem

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.

Abstract

The integration of structure in Machine Learning methods is a very active field in Bioinformatics. Methods recently developed claim that they allow at the same time to link the computed model to the graphical structure of the data set and to select a handful of important features in the analysis.

We nevertheless lack simulated data for which we can separate the three properties that the method claim achieving:

(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.

1 Introduction

Simulated data are broadly used for the assessment of optimization methods bacause of their ability to evaluate certain aspects of the studied methods that are impossible to look into when using real data sets. In the context of convex optimization, it is never possible to know the exact solution of the minimization problem with real data and it prooves to be a difficult problem even with simulated data. We propose to adapt an algorithm originally proposed by Nesterov for the LASSO regression to a borader family of penalised regressions.

We would like to generate simulated data for which we know the exact solution of the optimized function.

As inputs, there are the number of individuals n, the number of variables p, the number of non-null variables p^*, the covariance structure Σ, two regularization parameters κ and γ and the expression of the function $f(\beta)$ to minimize.

The general procedure of this approach is as follows:

(i) Generate a first version, $X^{(0)}$, of the X matrix. Let e.g. $X^{(0)} \sim N(\mathbb{1}, \Sigma)$.

(ii) Generate an error vector e with the desired noise distribution, e.g. $e \sim N(0, 1)$.

(iii) Generate the regression vector β^* so that it adhers to the constraints associated with the desired final loss function.

We obtain as outputs X and y such that

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

*tommy.lofstedt@cea.fr

2 Background

In this Section, we present the context of linear regression, with complex penalties and a first algorithm presented by Nesterov [**?**]. 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 normally distributed noise, β represents the unknown vector of length p containing 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.

In the following paragraphs, we provide mathematical definitions and notations that will frequently be used throughout this paper. First, we note by $\|\cdot\|_q$ the standard ℓ_q-norm defined on $\mathbb{R}^p$ by $\|x\|_q := \left(\sum_{j=1}^p x_j^q\right)^{\frac{1}{q}}$. For a smooth real function f on $\mathbb{R}^p$, we denote by $\nabla f(\beta)$ the gradient vector $(\partial_1 f(\beta), \ldots, \partial_p f(\beta))$; the function f has a Lipschitz continuous gradient on a convex set K with Lipschitz constant $L(\nabla(f)) > 0$ if for all $\beta_1, \beta_2 \in K$

$$|\nabla f(\beta_1) - \nabla f(\beta_2)| \leq L(\nabla(f))\|\beta_1 - \beta_2\|_2.$$

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 to naturally extend the notion of gradient 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: $dom(f) \subset \mathbb{R}^p \to \mathbb{R}$ at β if*

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

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

We conclude this section by recalling the definition of the proximal operator, which is a very important concept in the non-differentiable optimisation framework.

Definition 2.2. *Let $h : \mathbb{R}^p \to \mathbb{R}$ be a closed proper (i.e. $h(\beta) \neq +\infty$) convex function [**?**], then the proximal mapping (or proximal operator) $\text{prox}_h(x) : \mathbb{R}^p \to \mathbb{R}^p$ is defined by*

$$\text{prox}_h(\beta) := \arg\min_{u\in\mathbb{R}^p} \left\{ h(u) + \frac{1}{2}\|u - \beta\|^2 \right\}, \tag{2}$$

Note that we will often encounter the proximal operator of the scaled function th, where $t > 0$, which can be expressed as

$$\text{prox}_{th}(\beta) := \arg\min_{u\in\mathbb{R}^p} \left\{ h(u) + \frac{1}{2t}\|u - \beta\|^2 \right\}, \tag{3}$$

and will be referred to as the proximal operator of h with parameter t.

2.2 LASSO

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

This case is addressed by Nesterov [1], and we will therefore not go into details. Instead we will in this section simply adapt it to our notation and explain some steps that are not obvious.

The principle behind Nesterov's idea is as follows: First, generate the error, $\varepsilon = X\beta - y$, between $X\beta$ and y, independent from β^*. Then, select acceptable values for β^* such that 0 belongs to the sub-gradient of f at point β^*, with the subgradient being

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

We select β^* is such that

$$0 \in X^\top\varepsilon + \kappa\partial\|\beta^*\|_1, \tag{4}$$

and we stress again that that $X^\top\varepsilon$ does not depend on β^*.

We suppose that the $p - p^*$ last values of β are equal to 0 and distinguish two cases:

First case: We consider a variable $i \leq p^*$

The following constraints apply to β_i^*, the ith element of β^*:

(i) $\beta_i^* \neq 0$,

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

From constraint (i) it follows that $\partial|\beta_i^*| = \mathrm{sign}(\beta_i^*)$, and thus constraint (ii) follows because of Eq. (4) with X_i the ith column of X.

Second case: We consider i, with $p^* < i \leq p$

The constraint in this case is that $\beta_i^* = 0$. We note that the subgradient of $|\beta_i^*|$ when $\beta_i^* = 0$ is

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

and thus from Eq. (4) we see that

$$\frac{X_i^\top\varepsilon}{\kappa} \in [-1, 1]. \tag{5}$$

Solution

As mentioned earlier, the first step is to generate ε $(n \times 1)$, e.g. such that $\varepsilon_j \sim \mathcal{N}(0, 1)$ for all $1 \leq j \leq n$.

The next step is to generate an $n \times p$ matrix $X^{(0)}$ with a known covariance structure Σ such that $X^{(0)} \sim \mathcal{N}(0, \Sigma)$. It will serve as a first un-scaled version of X and in fact we have such that $X_i = \alpha_i X_i^{(0)}$, for $1 \leq i \leq p$.

If $i \leq p^*$, then $X_i^\top\varepsilon + \kappa\,\mathrm{sign}(\beta_i^*) = 0$ and thus

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

and since $X_i = \alpha_i X_i^{(0)}$ we have

$$\alpha_i = \frac{-\kappa\,\mathrm{sign}(\beta_i^*)}{X_i^{(0)\top}\varepsilon}.$$

We note that selecting $\beta_i \neq 0$ we obtain the required gradient in the case when $i \leq p^*$, i.e.

$$\partial|\beta_i^*| = \frac{X_i^\top \varepsilon}{\kappa} = \frac{\alpha_i X_i^{(0)\top} \varepsilon}{\kappa} = \frac{-\kappa \operatorname{sign}(\beta_i^*) X_i^{(0)\top} \varepsilon}{\kappa X_i^{(0)\top} \varepsilon} = -\operatorname{sign}(\beta_i^*) = \pm 1.$$

Thus, we may for instance let $\beta_i^* \sim \mathcal{U}^+(-1,1)$ in the case when $i \leq p^*$, where $\mathcal{U}^+(-1,1)$ means the uniform distribution on $[-1,1]$ except for 0.

If $i > p^*$, i.e. we have $\beta_i^* = 0$, we use Eq. (5) and have

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

Thus, with $X_i = \alpha_i X_i^{(0)}$ we obtain

$$\frac{\alpha_i X_i^{(0)\top} \varepsilon}{\kappa} \in [-1,1],$$

or equivalently

$$\alpha_i \sim \frac{\kappa \mathcal{U}(-1,1)}{X_i^{(0)\top} \varepsilon},$$

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

2.3 SNR and correlation

We use the same definition of signal-to-noise ratio as [**?**], 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

$$\text{SNR}(a) = \frac{\|X(\beta a)\beta a\|_2}{\|e\|_2}. \tag{6}$$

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 2.1. *Using the definition of simulated data described above, and with the definition of signal-to-noise ratio in Eq.* (6) *there exists an $a > 0$ such that*

$$\mathit{SNR}(a) = \sigma, \tag{7}$$

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

$$\begin{aligned}\|X(\beta a)\beta a\|_2^2 &= \left(\sum_{i=1}^{p} X_i\beta_i a\right)^\top \left(\sum_{i=1}^{p} X_i\beta_i a\right) \\ &= \sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^2 X_i^\top X_j \beta_i\beta_j + \sum_{i=1}^{p} a^2 X_i^\top X_i \beta_i^2 \\ &= \sum_{i=1}^{p}\sum_{j=1,j\neq i}^{p} a^2 X_{0,i}^\top X_{0,j} \beta_i\beta_j\omega_i\omega_j + \sum_{i=1}^{p} a^2 X_i^\top X_i \beta_i^2\omega_i^2.\end{aligned} \tag{8}$$

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. (8) 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

$$Aa^4 + Ba^3 + Ca^2 - s = 0. \tag{9}$$

This fourth order polynomial has a minimum of $-s$ at $a = 0$, also Eq. (6) 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. (9) 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 $\text{SNR}(a) = \sigma$.

We control the correlation structure of $X^{(0)}$ by e.g. letting $X^{(0)} \sim \mathcal{N}(0, \Sigma)$. Also, we let $X_i = \alpha_i X_i^{(0)}$ for all $1 \leq i \leq p$. It then follows that $\text{cor}(X_l, X_m) = \text{cor}(\alpha_l X_l^{(0)}, \alpha_m X_m^{(0)})$.

3 Method

The objective is to generate X, y and β^* such that

$$\beta^* = \arg\min_\beta f(\beta) + P(\beta),$$

where f is a function we already know how to simulate (e.g. OLS + Elastic net), and P is a penalty that can be expressed on the form

$$P(\beta) = \sum_{g=1}^{G} \|A_g \beta\|_p \tag{10}$$

where $\|\cdot\|_p$ is the p-norm with dual norm $\|\cdot\|_{p'}$. We will in this work only be interested in the case when $p = p' = 2$, i.e. the Euclidean norm.

This is the case when P is e.g. the Total variation constraint [**?**] or Group lasso [**?**].

3.1 Generalisation of Nesterov's Algorithm to any Number of Penalties

[Explain here that the scaling parameter has a very general form:

$$\alpha_i = \frac{\sum_{\pi \in \Pi} -\kappa_\pi r_\pi(\beta^*)}{\langle X^{(0)\top} | e \rangle},$$

where π and Π are a very general notation to represent the fact that we have many different penalties; κ_π is the regularisation parameter of penalty π and $r_\pi(\beta^*)$ is a candidate for the subgradient of penalty π at β^*.]

3.2 Subgradient of Complex Penalties

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

Lemma 3.1 (Sub-gradient 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 [**?**]. □

Lemma 3.2 (Sub-gradient 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 [**?**]. □

These lemmas play a central role in the following theorem that details the structure of the subgradient of P.

Theorem 3.3 (Sub-gradient of P)**.** *If P has the form given in Eq.* (10)*, then*

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

[How to write this with p-norm and dual p'-norm?]

Proof.

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

□

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

$$\partial\|x\|_2 = \begin{cases} \frac{x}{\|x\|_2} & \text{if } \|x\|_2 > 0, \\ \{y \mid \|y\|_2 \leq 1\} & \text{if } \|x\|_2 = 0. \end{cases} \tag{11}$$

3.3 Algorithm

[Here explain every step of the algorithm: SNR, correlation structure, compute the sub-gradients, compute the scaling parameters.]

4 Application

4.1 Total Variation

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

$$\mathrm{TV}\,(\beta) = \sum_{i=1}^{p} \|\nabla\beta_i\|_2, \tag{12}$$

where $\nabla\beta_i$ is the gradient at point β_i.

Since β is not continuous, the TV constraint needs to be approximated ***[necessary to go into this kind of details?]***. 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 $\|x\| = \sqrt{x^2} = |x|$, 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. (11) and obtain that

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

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 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 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

$$\partial TV(\beta) = A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_{24}\beta\|_2 \end{bmatrix}. \tag{13}$$

We use Eq. (11) and obtain

$$\partial\|A_i\beta\|_2 = \begin{cases} \frac{A_i\beta}{\|A_i\beta\|_2} & \text{if } \|A_i\beta\|_2 > 0, \\ \frac{\alpha u}{\|u\|_2},\ \alpha \sim \mathcal{U}(0,1), u \sim \mathcal{U}(-1,1)^3 & \text{if } \|A_i\beta\|_2 = 0. \end{cases} \tag{14}$$

[A little conclusion, maybe a remark on the fact that A is a very sparse matrix, which is important for the implementation.]

4.2 Linear regression with Elastic Net and Total Variation penalties

The subgradient is in this case

$$0 \in X^\top\varepsilon + (1-\kappa)\beta + \kappa\partial\|\beta^*\|_1 + \gamma\partial\,\mathrm{TV}(\beta^*). \tag{15}$$

The only difference with the previous derivation for LASSO arises when $i \le p^*$. In this case we have to first generate β^* such that $\beta_i^* \neq 0$. The we notice that

$$X_i^\top\varepsilon = -(1-\kappa)\beta_i^* - \kappa\,\mathrm{sign}(\beta_i^*)$$

and further, since $X_i = \alpha_i X_i^{(0)}$, that

$$\alpha_i = \frac{-(1-\kappa)\beta_i^* - \kappa \operatorname{sign}(\beta_i^*)}{X_i^{(0)\top}\varepsilon}$$

When $i > p^*$, adding a (smooth) Ridge constraint to the LASSO has no effect since $\beta_i^* = 0$. If we combine TV with OLS we obtain the function

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

The subdifferential of f at β^* must contain zero. Thus at β^* we have

$$0 \in \partial f(\beta) = X^\top \varepsilon + \gamma A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_G\beta\|_2 \end{bmatrix}.$$

With the subgradient of TV defined using Eq. (13) and Eq. (14), we obtain

$$\alpha_i = \frac{-\gamma \left(A^\top \begin{bmatrix} \partial\|A_1\beta\|_2 \\ \vdots \\ \partial\|A_G\beta\|_2 \end{bmatrix} \right)_i}{X_i^{(0)\top}\varepsilon},$$

for each variable $i = 1, \ldots, p$ and with $X_i = \alpha_i X_i^{(0)}$. With $(\cdot)_i$ denoting the ith variable of the vector within parentheses.

References

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