

The project will be realized in groups of 2 (“binômes”). At the end of the project, you will submit, for each part independently, a folder containing the MATLAB codes corresponding to the respective questions. Each folder must contain a file `main_demo.m` that should display the results upon execution. The codes must be accompanied by a short report (10 pages maximum, including figures) that details the calculations, summarizes the proposed algorithms and discusses the results.

## Part 1: Simulation of a Gaussian law truncated to positive values

In this part, we are interested in simulating the truncated Normal law defined by

$$f(x) = \frac{1}{K(m, \sigma^2)} \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right) \mathbf{1}_{\mathbb{R}^+}(x) \quad \text{avec} \quad K(m, \sigma^2) = \sqrt{\frac{\pi\sigma^2}{2}} \left[1 + \operatorname{erf}\left(\frac{m}{\sqrt{2\sigma^2}}\right)\right]$$

where the cumulative distribution function (CDF) is given by

$$F(t) = P[X < t] = \frac{\operatorname{erf}\left(\frac{t-m}{\sqrt{2\sigma^2}}\right) + \operatorname{erf}\left(\frac{m}{\sqrt{2\sigma^2}}\right)}{1 + \operatorname{erf}\left(\frac{m}{\sqrt{2\sigma^2}}\right)}$$

and where the function  $\operatorname{erf}(x)$  is defined as  $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2) dt$  (function `erf` in MATLAB).

1. Propose a first simulation algorithm based on the method of CDF inversion.  
Write a function `X = randnt_inversion(m,sigma2,N)` that returns a vector `X` of size  $N \times 1$  with elements following a normal law truncated to positive values, with parameters `m` and `sigma2`.

*Hint: one can use the function `erfinv` that returns the inverse of the function  $\operatorname{erf}(\cdot)$ .*

2. Compare the empirical histogram from  $N$  realizations with the theoretical probability density function.

The drawback of this first method is that one needs to know the functions  $\operatorname{erf}(\cdot)$  and  $\operatorname{erf}^{-1}(\cdot)$ . In order to simulate samples from a truncated normal law, we therefore propose the use of an accept-reject method. Recall that the accept-reject method relies on the use of a candidate law  $q(x)$  such that  $f(x) \leq Mq(x)$  ( $\forall x \in \mathbb{R}$ ) where  $M$  is a constant that determines the acceptance ratio.

3. First, choose the normal law  $\mathcal{N}(m, \sigma^2)$  as the candidate law.
  - (a) Compute the constant  $M$  that maximizes the acceptance ratio.
  - (b) From this, develop an algorithm that enables the generation of random variables that are distributed according to the truncated normal law. Write a function `X = randnt_ar1(m,sigma2,N)` that returns a vector `X` of size  $N \times 1$  has elements distributed according to the normal law truncated to positive values, with parameters `m` and `sigma2`.
4. Now choose an exponential distribution as the candidate law, defined by

$$q(x) = \lambda \exp(-\lambda x) \mathbf{1}_{\mathbb{R}^+}(x) \quad \text{with} \quad \lambda = \left(\sqrt{m^2 + 4\sigma^2} - m\right) / 2\sigma^2.$$

Answer question 3 with this new candidate law. The function `X = randnt_ar2(m,sigma2,N)` returns a vector `X` of size  $N \times 1$  has elements distributed according to the normal law truncated to positive values, with parameters `m` and `sigma2`.

5. Compare these two simulation methods in terms of acceptance ratio for different values of  $m$  and  $\sigma^2$ . Propose a strategy for efficiently simulating samples from the truncated normal law by carefully combining these two algorithms.

## Part 2: Descent Methods

We want to determine the minimum of the function  $f_0 : \mathbb{R}^2 \mapsto \mathbb{R}$  given by

$$f_0(x) = a \arctan(dx_1^2 + x_2^2) + b(x_1 - x_1^{(0)})^2 + c(x_2 - x_2^{(0)})^2$$

with  $x = (x_1, x_2)^T$ ,  $a = 0.5$ ,  $b = 8$ ,  $c = 0.55$ ,  $d = 2$  et  $x^{(0)} = (x_1^{(0)}, x_2^{(0)})^T = (0.5, 1)^T$ . The parameters are chosen such that  $f_0$  is convex.

We are first going to solve the problem without constraints

$$\min_x f_0(x)$$

using descent methods or order 1 and 2.

We will consider having reached the minimum at iteration  $k$  if the absolute deviation of the new and the old value of  $f_0$  is smaller than  $\varepsilon$ ,  $|f_0(x(k)) - f_0(x(k-1))| < \varepsilon$ , for two values  $\varepsilon_1 = 10^{-3}$  and  $\varepsilon_2 = 10^{-6}$ .

We use the point  $x' = (-0.8, -2.2)^T$  for initializing the descent.

1. Determine the gradient and the Hessian of  $f_0$  *Remark: you can use dedicated software for finding the expressions.*
2. Gradient descent
  - (a) Program a gradient descent algorithm with fixed step size  $\alpha(k)$ . Find fixed step sizes  $\alpha(k) = \alpha$  for which the algorithm converges for  $\varepsilon = \varepsilon_1$  and  $\varepsilon = \varepsilon_2$ .
  - (b) Include an “optimal” step size  $\alpha(k)$  in the gradient descent method. For the simple problem considered, we choose the “optimal” step size at each iteration  $k$  as the value  $\alpha'$  from a finite discrete set (for instance, `alpha=logspace(-3,1,1000)`) which minimizes  $f_0(x(k-1) + \alpha'd(k))$ .
3. Newton Method
  - (a) Program the Newton method and use it to find the solution for  $\varepsilon = \varepsilon_1$  et  $\varepsilon = \varepsilon_2$ .
  - (b) Include an “optimal” step size in the Newton method in the same way as for the gradient descent method.
4. How many iterations are needed for convergence for  $\varepsilon = \varepsilon_1$  and  $\varepsilon = \varepsilon_2$  for the different descent algorithms and different choices of step size? Illustrate, compare et interpret the behaviour and performance of the algorithms.

## Part 3: Deconvolution of sparse signals

In this part, we consider different methods for finding the solution to the inverse problem defined by

$$y = Hx + n$$

where  $n$  is a zero-mean Gaussian white noise term with known covariance matrix  $\sigma^2 I$  ( $\sigma^2$  known),  $H$  is a convolution matrix and  $x \in \mathbb{R}_+^N$  is a sparse positive signal ( $x_i \geq 0$ ). Finding a solution to this problem consists in estimating the signal  $x$  from a vector of observations  $y$ . We will solve the inverse problem using a convex optimization algorithm, and using a Monte Carlo Markov Chain algorithm.

An example for generating the signal and solving the problem by means of *Orthogonal Matching Pursuit* is given by the file `main_deconvolution_OMP.m`. You will use this file to answer the following questions and study and compare the methods and their performance, notably as a function of the problem parameters ( $\sigma^2$  and  $\lambda$ ).

1. The equivalent convex optimisation problem to be solved is

$$\min_x \|y - Hx\|_2^2 + \lambda \|x\|_1 \quad \text{such that } x_i \geq 0, \forall i,$$

i.e.,  $f_0(x) = \|y - Hx\|_2^2 + \lambda \|x\|_1 = f(x) + g(x)$  with  $f(x)$  differentiable  $g(x)$  non-smooth.

- (a) Program a proximal gradient -descent algorithm for this problem *without* positivity constraints, i.e.,  $\min_x \|y - Hx\|_2^2 + \lambda \|x\|_1$ . In order to find the step size  $\alpha^k$  at each iteration, one can use the rule

$$\alpha^k = \alpha, \quad 0 \ll \beta < 1, \quad \text{while } \|r^k\|_2^2 > \|r^{k-1}\|_2^2 - \nabla f(x^{k-1} - x^k) + \frac{1}{2\alpha^k} \|x^{k-1} - x^k\|_2^2 \quad \text{do} \quad \alpha^k = \alpha^k \beta$$

where  $r^k = y - Hx^k$  is the residual for  $\alpha^k$ .

- (b) Include the positivity constraint by using the projected gradient approach, i.e.,  $x_C^k = Proj_C(x^k)$  with  $C = \{x : x_i \geq 0\}$ .
2. In a Bayesian setting, this problem can be solved by choosing an exponential prior distribution for each element  $x_i$  of  $x$ , which are assumed to be independent:

$$f(x_i) = \lambda \exp(-\lambda x_i) \mathbf{1}_{\mathbb{R}^+}(x_i). \quad (1)$$

- (a) Calculate the conditional posterior law of each element of the vector  $x$  conditioned on all other elements:  $f(x_i | x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N, y)$ .
- (b) Based on this conditional posterior law, deduce a Gibbs sampling algorithm that enables to solve the problem. You can use one or several of the algorithms developed in Part 1.