MVA - Convex Optimization
DM3

Ariane ALIX

November 25th, 2019

---

1. We consider the LASSO problem, defined by :

$$\min_{w} \quad \frac{1}{2}\|Xw - y\|_2^2 + \lambda\|w\|_1$$

Where $X = (x_1^T, ..., x_n^T) \in \mathbb{R}^{nxd}$, $y = (y_1, ..., y_n) \in \mathbb{R}^n$, $\lambda > 0$ is a regularization problem, and we aim to minimize with regard to $w \in \mathbb{R}^d$.

Introducing a dummy variable $v = w$, we can write the following equivalent problem :

$$\begin{aligned} \min_{v,w} \quad & \frac{1}{2}\|Xw - y\|_2^2 + \lambda\|v\|_1 \\ \text{s.t} \quad & w = v \end{aligned}$$

The Lagrangian is written as :

$$\begin{aligned} L((w, v), \mu) &= \frac{1}{2}\|Xw - y\|_2^2 + \lambda\|v\|_1 + \mu^T(v - w) \\ &= \frac{1}{2}(Xw - y)^T(Xw - y) + \lambda\|v\|_1 + \mu^T(v - w) \end{aligned}$$

And the dual function is :

$$g(\mu) = \inf_{w,v} \frac{1}{2}\|Xw - y\|_2^2 + \lambda\|v\|_1 + \mu^T(v - w)$$

Since L is the sum of two independant parts depending respectively on w and w, we can compute they minimum separately.

For w.
We consider $\frac{1}{2}\|Xw - y\|_2^2 - \mu^T w$. Since it is convex, we look at the gradient to minimize it.

$$
\begin{aligned}
\nabla_w L((w,v),\mu) &= \frac{1}{2}(2X^TXw - 2X^Ty) - \mu \\
&= X^TXw - X^Ty - \mu
\end{aligned}
$$

Which is equal to 0 for $w = (X^TX)^{-1}(X^Ty + \mu)$.

Hence,

$$
\begin{aligned}
\min_w \frac{1}{2}\|Xw - y\|_2^2 - \mu^T w &= \min_w \frac{1}{2}(Xw - y)^T(Xw - y) - \mu^T w \\
&= \min_w \frac{1}{2}w^TX^TXw - \frac{1}{2}w^TX^Ty - \frac{1}{2}y^TXw + \frac{1}{2}y^Ty - \mu^T w \\
&= \min_w \frac{1}{2}w^T(X^TXw - X^Ty) - \frac{1}{2}y^TXw + \frac{1}{2}y^Ty - \mu^T w \\
&= -\frac{1}{2}\mu^T(X^TX)^{-1}(X^Ty + \mu) - \frac{1}{2}y^TX(X^TX)^{-1}(X^Ty + \mu) + \frac{1}{2}y^Ty
\end{aligned}
$$

For v.

We consider $\lambda\|v\|_1 + \mu^T v$. We notice that it corresponds to the opposite of the conjugate of the norm L_1 :

$$
\begin{aligned}
\inf_v \lambda\|v\|_1 + \mu^T v &= -\lambda \sup_v -\|v\|_1 - \frac{\mu^T}{\lambda} v \\
&= \begin{cases} 0, & \text{if } -1 \preceq \frac{\mu}{\lambda} \preceq 1 \\ -\infty, & \text{otherwise} \end{cases} \\
&= \begin{cases} 0, & \text{if } -\lambda \preceq \mu \preceq \lambda \\ -\infty, & \text{otherwise} \end{cases}
\end{aligned}
$$

Plugging these two results in g, we get :

$$
g(\mu) = \begin{cases} -\frac{1}{2}\mu^T(X^TX)^{-1}(X^Ty + \mu) - \frac{1}{2}y^TX(X^TX)^{-1}(X^Ty + \mu) + \frac{1}{2}y^Ty, & \text{if } -\lambda \preceq \mu \preceq \lambda \\ -\infty, & \text{otherwise} \end{cases}
$$

The dual problem is therefore :

$$
\begin{aligned}
\max_\mu \quad & -\frac{1}{2}\mu^T(X^TX)^{-1}(X^Ty + \mu) - \frac{1}{2}y^TX(X^TX)^{-1}(X^Ty + \mu) + \frac{1}{2}y^Ty \\
\text{s.t} \quad & \lambda \preceq \mu \preceq \lambda
\end{aligned}
$$

Which can be simplified by deleting the constant terms to :

$$
\begin{aligned}
\max_\mu \quad & -\frac{1}{2}\mu^T(X^TX)^{-1}\mu - y^TX(X^TX)^{-1}\mu \\
\text{s.t} \quad & \lambda \preceq \mu \preceq \lambda
\end{aligned}
$$

And, after inverting the sign and writing the box condtion like a linear one, we get the quadratic problem :

$$\min_{\mu} \quad \mu^T Q \mu + p^T \mu$$
$$\text{s.t} \quad A\mu \preceq b$$

Where :
$Q = \frac{1}{2}(X^T X)^{-1}$,
$p = (X^T X)^{-1} X^T y$,
$A = [-I_d, I_d]^T$,
$b = \lambda \mathbf{1}_{2d}$

2.

Centering step

The function we need to minimize is $tf_0 + \Phi = t(v^T Qv + p^T v) - \sum_{i=1}^{2d} \log(b[i] - A[i]v)$, where Φ is the log-barrier.

At each step, we compute the gradient and the hessian of this function (let's call it g) :

$$\nabla g = t(2Qv + p) + \sum_{i=1}^{2d} \frac{A[i]}{b[i] - A[i]v}$$

$$\nabla^2 g = 2tQ + \sum_{i=1}^{2d} \frac{A[i]A[i]^T}{(b[i] - A[i]v)^2}$$

The backtracking line search compares the loss for v plus a step in the diection of the gradient descent :

$$\text{loss}(v - \text{step} * \nabla^2 g^{-1} * \nabla g)$$

to the loss of v plus a value :

$$\text{loss}(v) - \alpha * \text{step} * \nabla g^T \nabla^2 g^{-1} * \nabla g$$

Where the **loss** function computes $t(v^T Qv + p^T v) - \sum_{i=1}^{2d} \log(b[i] - A[i]v)$. We also check during this loop that the conditions on v described by A and b are respected.

The iteration in the centering step stops when the precision goes below ϵ (in our case the criterion chosen is the distance between two successive values for v).

The calculation is detailed in the code which is commented at every step.

Barrier method

The function *barr_method* loops while $\frac{m}{t} > \epsilon$ (with $m = 2d$ the number of constraints). At each iteration, it computes a new v thanks to the centering step, and updates the factor t by $t = \mu t$.

3. The code was tested with X and Y randomly generated, with $X \in [-100, 100]^{100\text{x}2}$, and $Y \in [-100, 100]^{100}$. With those same values, we tried the barrier method with $\mu = 2, 15, 50, 100$, and we observed this evolution of v:

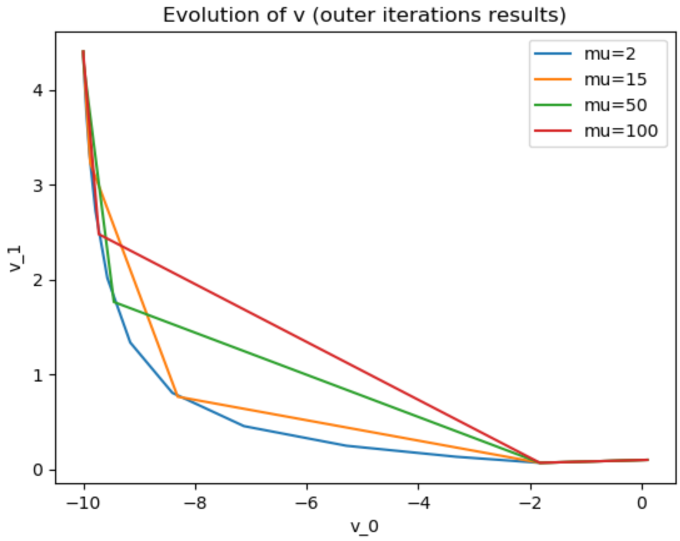

Figure 1 – v found after each outer iteration

And this evolution of the precision criterion $\frac{m}{t}$ and of the loss gap $\|f(v_t) - f^*\|$:

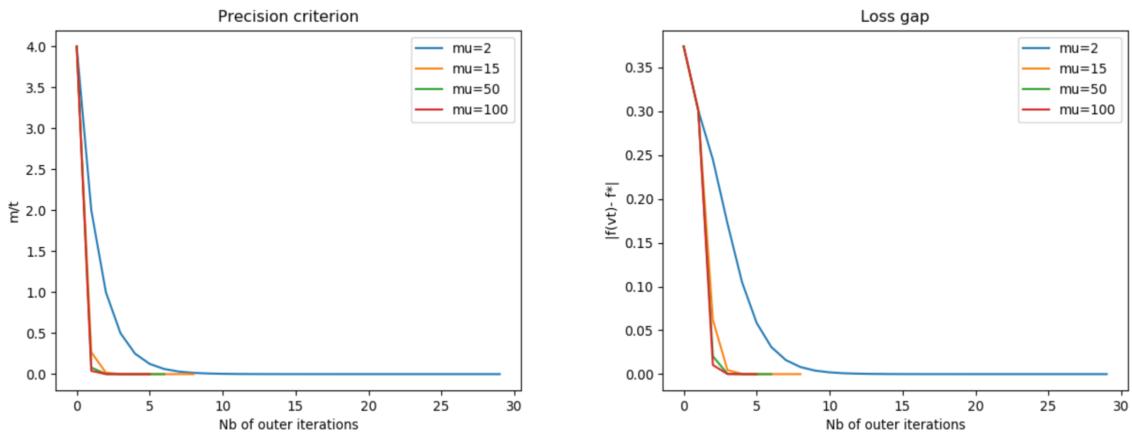

Figure 2 – Precision and Loss gap after each outer iteration

For all the parameters μ, the method converged toward the same solution.

We can see in the **Loss gap** that $\mu = 50$ and $\mu = 100$ both converged in 3 outer iterations, but the stopping condition on $\frac{m}{t}$ makes it possible in the case of $\mu = 50$ to have one more iteration to optimize v. Therefore $\mu = 50$ would be an appropriate choice.