Coordinate descent methods

Contents

1 Exact coordinate descent

The idea of coordinate descent is to decompose a large optimisation problem into a sequence of one-dimensional optimisation problems. The algorithm was first described for the minimization of quadratic functions by Gauss and Seidel in [Seidel, 1874]. Coordinate descent methods have become unavoidable in machine learning because they are very efficient for key problems, namely Lasso, logistic regression and support vector machines. Moreover, the decomposition into small subproblems means that only a small part of the data is processed at each iteration and this makes coordinate descent easily scalable to high dimensions.

We first decompose the space of optimisation variables X into blocks $X_1 \times \ldots \times X_n = X$. A classical choice when $X = \mathbb{R}^n$ is to choose $X_1 = \ldots = X_n = \mathbb{R}$. We will denote U_i the canonical injection from X_i to X, that is U_i is such that for all $h \in X_i$,

$$U_i h = (\underbrace{0, \ldots, 0}_{i-1 \text{ zeros}}, h^\top, \underbrace{0, \ldots, 0}_{n-i \text{ zeros}})^\top \in X.$$

For a function $f : X_1 \times \ldots \times X_n \to \mathbb{R}$, we define the following algorithm.

Algorithm 1: Exact coordinate descent

Start at $x_0 \in X$.

At iteration k, choose $l = (k \mod n) + 1$ (cyclic rule) and define $x_{k+1} \in X$ by

$$\begin{cases} x_{k+1}^{(i)} = \arg\min_{z \in X_l} f(x_k^{(1)}, \ldots, x_k^{(l-1)}, z, x_k^{(l+1)}, \ldots, x_k^{(n)}) & \text{if } i = l \\ x_{k+1}^{(i)} = x_k^{(i)} & \text{if } i \neq l \end{cases}$$

Proposition 1 ([Warga, 1963])**.** *If f is continuously differentiable and strictly convex and there exists $x_* = \arg\min_{x \in X} f(x)$, then the exact coordinate descent method (Alg. 1) converges to x_*.*

Exercise 1 (least squares)**.** $f(x) = \frac{1}{2}\|Ax - b\|_2^2 = \frac{1}{2}\sum_{j=1}^m (a_j^\top x - b_j)^2$

At each iteration, we need to solve in z the 1D equation

$$\frac{\partial f}{\partial x^{(l)}}(x_k^{(1)}, \ldots, x_k^{(l-1)}, z, x_k^{(l+1)}, \ldots, x_k^{(n)}) = 0$$

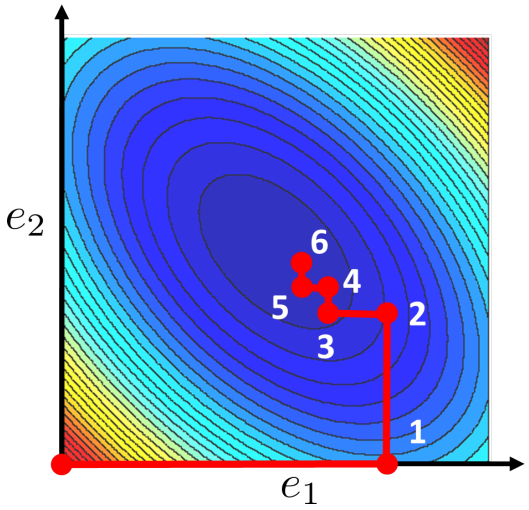

Figure 1: The successive iterates of the coordinate descent method on a 2D example. The function we are minimising is represented by its level sets: the bluer is the circle, the lower is the function values.

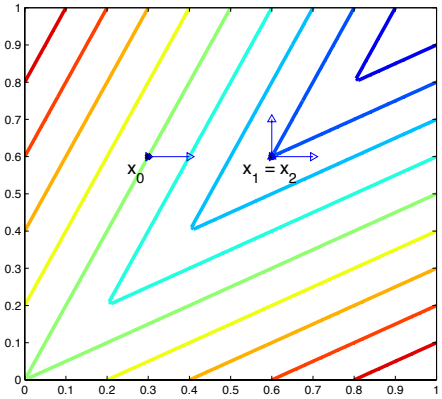

Figure 2: The function in Example 2

For all $x \in \mathbb{R}^n$,

$$\frac{\partial f}{\partial x^{(l)}}(x) = a_l^\top (Ax - b) = a_l^\top a_l x^{(l)} + a_l^\top (\sum_{j \neq l} a_j x^{(j)}) - a_l^\top b$$

so we get to be exact, the differential equals 0

$$z^* = x_{k+1}^{(l)} = \frac{1}{\|a_l\|_2^2}\Big(- a_l^\top (\sum_{j \neq l} a_j x_k^{(j)}) + a_l^\top b\Big) = x_k^{(l)} - \frac{1}{\|a_l\|_2^2}\Big(a_l^\top (\sum_{j=1}^{n} a_j x_k^{(j)}) - a_l^\top b\Big)$$

Exercise 2 (non-differentiable function)**.** $f(x^{(1)}, x^{(2)}) = |x^{(1)} - x^{(2)}| - \min(x^{(1)}, x^{(2)}) + \mathbb{I}_{[0,1]^2}(x)$

f is convex but not differentiable. If we nevertheless try to run exact coordinate descent, the algorithm proceeds as $x_1^{(1)} = \arg\min_z f(z, x_0^{(2)}) = x_0^{(2)}$, $x_2^{(2)} = \arg\min_z f(x_1^{(1)}, z) = x_0^{(2)}$, *and so on. Thus exact coordinate descent converges in two iterations to* $(x_0^{(2)}, x_0^{(2)})$*: the algorithm is stuck on the non-differentiability point on the line* $\{x^{(1)} = x^{(2)}\}$ *and does not reach the minimiser* $(1, 1)$.

Exercise 3 (non-convex differentiable function)**.**
$f(x^{(1)}, x^{(2)}, x^{(3)}) = -(x^{(1)}x^{(2)} + x^{(2)}x^{(3)} + x^{(3)}x^{(1)}) + \sum_{i=1}^{3} \max(0, |x^{(i)}| - 1)^2$

As shown by [Powell, 1976], exact coordinate descent on this function started at the initial point $x^{(0)} = (-1-\epsilon, 1+\epsilon/2, -1-\epsilon/4)$ *has a limit cycle around the 6 corners of the cube that are not minimisers and avoids the 2 corners that are minimisers.*

Exercise: *Show Powell's result.*

Exercise 4 (Adaboost)**.** *The Adaboost algorithm [Collins et al., 2002] was designed to minimise the exponential loss given by*

$$f(x) = \sum_{j=1}^{m} \exp(-y_j h_j^\top x).$$

At each iteration, we select the variable l *such that* $l = \arg\max_i |\nabla_i f(x)|$ *and we perform an exact coordinate descent step along this coordinate.*

This variable selection rule is called the greedy or Gauss-Southwell rule. Like the cyclic rule, it leads to a converging algorithm but requires to compute the full gradient at each iteration. Greedy coordinate descent is interesting in the case of the exponential loss because the gradient of the function has a few very large coefficients and many negligible coefficients.

Exercise: *Suppose that* $y_j \in \{-1, 1\}$ *and* $h_{j,i} \in \{-1, 0, 1\}$ *for all* j, i*. Give the explicit formulas of* $\nabla_i f(x)$ *and of the next update* x_{k+1} *knowing* x_k*.*

2 Coordinate gradient descent

Solving a one-dimensional optimisation problems is generally easy and the solution can be approximated very well by algorithms like the bisection method. However, for the exact coordinate descent method, one needs to solve a huge number of one-dimensional problems and the expense quickly becomes prohibitive. Moreover, why should we solve to high accuracy the 1-dimensional problem and destroy this solution at the next iteration?

The idea of coordinate gradient descent is to perform one iteration of gradient descent in the 1-dimensional problem $\min_{z \in X_l} f(x_k^{(1)}, \dots, x_k^{(l-1)}, z, x_k^{(l+1)}, \dots, x_k^{(n)})$ instead of solving it completely. In general, this reduces drastically the cost of each iteration while keeping the same convergence behaviour.

Algorithm 2: Coordinate gradient descent

Start at x_0.

At iteration k, choose $i_{k+1} \in \{1, \dots, n\}$ and define x_{k+1} by

$$\begin{cases} x_{k+1}^{(i)} = x_k^{(i)} - \gamma_i \nabla_i f(x_k) & \text{if } i = i_{k+1} \\ x_{k+1}^{(i)} = x_k^{(i)} & \text{if } i \neq i_{k+1} \end{cases}$$

When choosing the cyclic or greedy rule, the algorithm does converge for any convex function f that has a Lipschitz-continuous gradient and such that $\arg\min_x f(x) \neq \emptyset$.

In fact we will assume that we actually know the *coordinate-wise* Lipschitz constants of the gradient of f, namely the Lipschitz constants of the functions

$$\begin{aligned} g_{i,x} : \; & X_i \to \mathbb{R} \\ & h \mapsto f(x + U_i h) = f(x^{(1)}, \dots, x^{(i-1)}, x^{(i)} + h, x^{(i+1)}, \dots, x^{(n)}) \end{aligned}$$

We will denote $L_i = L(\nabla g_{i,x})$ this Lipschitz constant. Written in terms of f, this means that

$$\forall x \in X, \forall i \in \{1, \dots, n\}, \forall h \in X_i, \quad \|\nabla f(x + U_i h) - \nabla f(x)\|_2 \leq L_i \|U_i h\|_2.$$

Lemma 2. *If* f *has a coordinate-wise Lipschitz gradient with constants* $L_1, \dots, L_n$*, then* $\forall x \in X$*,* $\forall i \in \{1, \dots, n\}, \forall h \in X_i$*,*

$$f(x + U_i h) \leq f(x) + \langle \nabla_i f(x), h \rangle + \frac{L_i}{2} \|h\|^2$$

PROOF. This is Taylor's inequality applied to $g_{i,x}$. Note that we do not require the function to be twice differentiable.

Proposition 3 ([Beck and Tetruashvili, 2013]). *Assume that f is convex, ∇f is Lipschitz continuous and $\arg\min_{x\in X} f(x) \neq \emptyset$. If i_{k+1} is chosen with the cyclic rule $i_{k+1} = (k \mod n) + 1$ and $\forall i$, $\gamma_i = \frac{1}{L_i}$, then the coordinate gradient descent method (Alg. 2) satisfies*

$$f(x_{k+1}) - f(x_*) \leq 4L_{\max}(1 + n^3 L_{\max}^2/L_{\min}^2)\frac{R^2(x_0)}{k + 8/n}$$

where $R^2(x_0) = \max_{x,y\in X}\{\|x - y\| : f(y) \leq f(x) \leq f(x_0)\}$, $L_{\max} = \max_i L_i$ and $L_{\min} = \min_i L_i$.

The proof of this result is quite technical and in fact the bound is much more pessimistic than what is observed in practice (n^3 is very large if n is large). This is due to the fact that the cyclic rule behaves particularly bad on some extreme examples. To avoid such traps, it has been suggested to randomise the coordinate selection process.

Proposition 4 ([Nesterov, 2012]). *Assume that f is convex, ∇f is Lipschitz continuous and $\arg\min_{x\in X} f(x) \neq \emptyset$. If i_{k+1} is randomly generated, independently of $i_1, \ldots, i_k$ and $\forall i \in \{1, \ldots, n\}$, $\mathbb{P}(i_{k+1} = i) = \frac{1}{n}$ and $\gamma_i = \frac{1}{L_i}$, then the coordinate gradient descent method (Alg. 2) satisfies for all $x_* \in \arg\min_x f(x)$*

$$\mathbb{E}[f(x_{k+1}) - f(x_*)] \leq \frac{n}{k+n}\Big((1 - \frac{1}{n})(f(x_0) - f(x_*)) + \frac{1}{2}\|x_* - x_0\|_L^2\Big)$$

where $\|x\|_L^2 = \sum_{i=1}^n L_i \|x^{(i)}\|_2^2$.

PROOF. This is a particular case of the method developed in the next section.

Comparison with gradient descent The iteration complexity of the gradient descent method is

$$f(x_{k+1}) - f(x_*) \leq \frac{L(\nabla f)}{2(k+1)}\|x_* - x_0\|_2^2$$

This means that to get an ϵ-solution (*i.e.,* such that $f(x_k) - f(x_*) \leq \epsilon$), we need at most $\frac{L(\nabla f)}{2\epsilon}\|x_* - x_0\|_2^2$ iterations. What is most expensive in gradient descent is the evaluation of the gradient $\nabla f(x)$ with a cost C, so the total cost of the method is

$$C_{\text{grad}} = C\frac{L(\nabla f)}{2\epsilon}\|x_* - x_0\|_2^2$$

Neglecting the effect of randomisation, we usually have an ϵ-solution with coordinate descent in $\frac{n}{\epsilon}\big((1 - \frac{1}{n})(f(x_0) - f(x_*)) + \frac{1}{2}\|x_* - x_0\|_L^2\big)$ iterations. The cost of one iteration of coordinate descent is of the order of the cost of evaluation one partial derivative $\nabla_i f(x)$, with a cost c, so the total cost of the method is

$$C_{\text{cd}} = c\frac{n}{\epsilon}\big((1 - \frac{1}{n})(f(x_0) - f(x_*)) + \frac{1}{2}\|x_* - x_0\|_L^2\big)$$

How do these two quantities compare?
Let us consider the case where $f(x) = \frac{1}{2}\|Ax - b\|_2^2$.

- Computing $\nabla f(x) = A^\top(Ax - b)$ amounts to updating the residuals $r = Ax - b$ (one matrix vector product and a sum) and computing one matrix vector product. We thus have $C = O(\text{nnz}(A))$.
- Computing $\nabla_i f(x) = e_i^\top A^\top(Ax - b)$ amounts to
  1. updating the residuals $r = Ax - b$: one scalar-vector product and a sum since we have $r_{k+1} = r_k + (x_{k+1}^{(i_{k+1})} - x_k^{(i_{k+1})})Ae_{i_{k+1}}$,
  2. computing one vector-vector product (the i^{th} column of A versus the residuals).

  Thus $c = O(\text{nnz}(Ae_{i_{k+1}})) = O(\text{nnz}(A)/n) = C/n$ if the columns of A are equally sparse.
- $f(x_0) - f(x_*) \leq \frac{L(\nabla f)}{2}\|x_0 - x_*\|_2^2$ and it may happen that $f(x_0) - f(x_*) \ll \frac{L(\nabla f)}{2}\|x_0 - x_*\|_2^2$
- $L(\nabla f) = \lambda_{\max}(A^\top A)$ and $L_i = a_i^\top a_i$ with $a_i = Ae_i$. We always have $L_i \leq L(\nabla f)$ and it may happen that $L_i = O(L(\nabla f)/n)$.

To conclude, in the quadratic case, $C_{\text{cd}} \leq C_{\text{grad}}$ and we may have $C_{\text{cd}} = O(C_{\text{grad}}/n)$.

3 Proximal coordinate descent

We are often interested in solving problems of the type

$$\min_{x\in X} F(x) = \min_{x\in X} f(x) + g(x) \tag{1}$$

where f and g are convex so that $F = f + g$ is convex, f has a Lipschitz continuous gradient and g may be nonsmooth but is separable. This means that for all $x \in X = X_1 \times \dots X_n$,

$$g(x) = \sum_{i=1}^{n} g_i(x^{(i)}).$$

We can solve this kind of problems with the proximal coordinate descent method (Alg. 3, [Tseng, 2001]), which is also using the coordinate-wise Lipschitz constant.

Algorithm 3: Proximal coordinate descent

Start at $x_0 \in X$.
At iteration k, choose $i_{k+1} \in \{1, \dots, n\}$ and define $x_{k+1} \in X$ by

$$\begin{cases} x_{k+1}^{(i)} = \arg\min_{x\in X_i} g_i(x) + f(x_k) + \langle \nabla_i f(x_k), x - x_k^{(i)}\rangle + \frac{L_i}{2}\|x - x_k^{(i)}\|^2 & \text{if } i = i_{k+1} \\ x_{k+1}^{(i)} = x_k^{(i)} & \text{if } i \neq i_{k+1} \end{cases}$$

For this algorithm to be practical, we need to be able to compute efficiently

$$\text{prox}_{\gamma,g}(y) = \arg\min_{x\in X} g(x) + \frac{1}{2}\|x - y\|^2_{\gamma^{-1}},$$

the proximal operator of g (remember that $\|x\|^2_{\gamma^{-1}} = \sum_{i=1}^n \frac{1}{\gamma_i}\|x^{(i)}\|_2^2$).

Exercise 5 (Simple proximal operators)**.**

- *Indicator of a box: if* $g(x) = \mathbb{I}_{[a,b]}(x)$*, then* $\text{prox}_{\gamma,g}(y) = \max(a, \min(x, b))$*. This is the projection on* $[a, b]$ *(it does not depend on* γ*).*
- *Absolute value: if* $g(x) = \lambda|x|$*, then* $\text{prox}_{\gamma,g}(y) = \text{sign}(y)\max(0, |y| - \gamma\lambda)$*. This is the soft-thresholding operator.*

We define 将模展开即可

$$\bar{x}_{k+1} = \text{prox}_{L^{-1},g}(x_k - L^{-1}\nabla f(x_k)) = \arg\min_{x\in X} g(x) + f(x_k) + \langle \nabla f(x_k), x - x_k\rangle + \frac{1}{2}\|x - x_k\|_L^2,$$

so that

$$x_{k+1}^{(i)} = \begin{cases} \bar{x}_{k+1}^{(i)} & \text{if } i = i_{k+1} \\ x_k^{(i)} & \text{if } i \neq i_{k+1} \end{cases}$$

Lemma 5. *For all* $\gamma \in \mathbb{R}^n_{+*}$ *and* $x \in X$

$$g(\bar{x}_{k+1}) + \langle \nabla f(x_k), x_{k+1} - x_k\rangle + \frac{1}{2}\|x_{k+1} - x_k\|^2_{\gamma^{-1}} \leq g(x) + \langle \nabla f(x_k), x - x_k\rangle + \frac{1}{2}\|x - x_k\|^2_{\gamma^{-1}} - \frac{1}{2}\|x_{k+1} - x\|^2_{\gamma^{-1}}$$

PROOF. The function $\psi : x \mapsto g(x) + \langle \nabla f(x_k), x - x_k\rangle + \frac{1}{2}\|x - x_k\|^2_{\gamma^{-1}}$ is strongly convex and its minimiser is $\bar{x}_{k+1}$. The inequality is just the strong convexity inequality of this function with respect to the norm $\|\cdot\|^2_{\gamma^{-1}}$ and applied at x and $\bar{x}_{k+1}$.

Theorem 6 ([Richtárik and Takáč, 2014])**.** *The proximal coordinate descent method (Alg. 3) with the random selection rule applied to Problem 1 satisfies for all* $x_* \in \arg\min_x F(x)$

$$\mathbb{E}[F(x_{k+1}) - F(x_*)] \leq \frac{n}{k+n}\Big((1 - \frac{1}{n})(F(x_0) - F(x_*)) + \frac{1}{2}\|x_* - x_0\|_L^2\Big)$$

where $\|x\|_L^2 = \sum_{i=1}^n L_i\|x^{(i)}\|_2^2$.

proof. By definition of the algorithm, $x_{k+1} - x_k = U_{i_{k+1}}(x_{k+1}^{(i_{k+1})} - x_k^{(i_{k+1})})$, so by Lemma 2,

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) + \langle \nabla_{i_{k+1}} f(x_k), x_{k+1}^{(i_{k+1})} - x_k^{(i_{k+1})} \rangle + \frac{L_{i_{k+1}}}{2} \|x_{k+1}^{(i_{k+1})} - x_k^{(i_{k+1})}\|^2 \\ &= f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + \frac{1}{2}\|x_{k+1} - x_k\|_L^2 \end{aligned} \tag{2}$$

Using the notation $\bar{x}_{k+1} = \arg\min_{x \in X} g(x) + f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2}\|x - x_k\|_L^2$, we have

$$x_{k+1}^{(i)} = \begin{cases} \bar{x}_{k+1}^{(i)} & \text{if } i = i_{k+1} \\ x_k^{(i)} & \text{if } i \neq i_{k+1} \end{cases}$$

Using the conditional expectation knowing $\mathcal{F}_k = (i_1, \dots i_k)$, we get

$$\mathbb{E}[x_{k+1}^{(i)} | \mathcal{F}_k] = \mathbb{P}(i_{k+1} = i)\bar{x}_{k+1}^{(i)} + \mathbb{P}(i_{k+1} \neq i)x_k^{(i)} = \frac{1}{n}\bar{x}_{k+1}^{(i)} + (1 - \frac{1}{n})x_k^{(i)}$$

$$\begin{aligned} \mathbb{E}[\langle \nabla_i f(x_k), x_{k+1}^{(i)} - x_k^{(i)} \rangle | \mathcal{F}_k] &= \mathbb{P}(i_{k+1} = i)\langle \nabla_i f(x_k), \bar{x}_{k+1}^{(i)} - x_k^{(i)} \rangle + \mathbb{P}(i_{k+1} \neq i)\langle \nabla_i f(x_k), x_k^{(i)} - x_k^{(i)} \rangle \\ &= \frac{1}{n}\langle \nabla_i f(x_k), \bar{x}_{k+1}^{(i)} - x_k^{(i)} \rangle \end{aligned}$$

$$\mathbb{E}[\langle \nabla f(x_k), x_{k+1} - x_k \rangle | \mathcal{F}_k] = \sum_{i=1}^n \mathbb{E}[\langle \nabla_i f(x_k), x_{k+1}^{(i)} - x_k^{(i)} \rangle | \mathcal{F}_k] = \frac{1}{n}\langle \nabla f(x_k), \bar{x}_{k+1} - x_k \rangle \tag{3}$$

$$\mathbb{E}[\frac{1}{2}\|x_{k+1} - x_k\|_L^2 | \mathcal{F}_k] = \sum_{i=1}^n \mathbb{E}[\frac{L_i}{2}\|x_{k+1}^{(i)} - x_k^{(i)}\|^2 | \mathcal{F}_k] = \frac{1}{2n}\|\bar{x}_{k+1} - x_k\|_L^2 \tag{4}$$

$$\mathbb{E}[g(x_{k+1}) - g(x_k) | \mathcal{F}_k] = \sum_{i=1}^n \mathbb{E}[g_i(x_{k+1}^{(i)}) - g_i(x_k^{(i)}) | \mathcal{F}_k] = \frac{1}{n}(g(\bar{x}_{k+1}) - g(x_k)) \tag{5}$$

Combining (3), (4) and (5) with (2), we get

$$\begin{aligned} \mathbb{E}[g(x_{k+1}) + f(x_{k+1}) | \mathcal{F}_k] &\leq \mathbb{E}[g(x_{k+1}) | \mathcal{F}_k] + \mathbb{E}\Big[f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + \frac{1}{2}\|x_{k+1} - x_k\|_L^2 \Big| \mathcal{F}_k\Big] \\ &= (1 - \frac{1}{n})g(x_k) + \frac{1}{n}g(\bar{x}_{k+1}) + f(x_k) + \frac{1}{n}\langle \nabla f(x_k), \bar{x}_{k+1} - x_k \rangle + \frac{1}{2n}\|\bar{x}_{k+1} - x_k\|_L^2 \end{aligned}$$

Using Lemma 5 with $x = x_k$, we get

$$\begin{aligned} \mathbb{E}[F(x_{k+1}) | \mathcal{F}_k] = \mathbb{E}[g(x_{k+1}) + f(x_{k+1}) | \mathcal{F}_k] &\leq g(x_k) + f(x_k) - \frac{1}{n}\|\bar{x}_{k+1} - x_k\|_2^2 \\ &\leq g(x_k) + f(x_k) = F(x_k) \end{aligned} \tag{6}$$

Then, using Lemma 5 again, with $x = x_*$, we get

$$\begin{aligned} &\mathbb{E}[F(x_{k+1}) | \mathcal{F}_k] = \mathbb{E}[g(x_{k+1}) + f(x_{k+1}) | \mathcal{F}_k] \\ &\leq (1 - \frac{1}{n})g(x_k) + f(x_k) + \frac{1}{n}g(x_*) + \frac{1}{n}\langle \nabla f(x_k), x_* - x_k \rangle + \frac{1}{2n}\|x_* - x_k\|_L^2 - \frac{1}{2n}\|x_* - \bar{x}_{k+1}\|_L^2 \end{aligned}$$

We remark that

$$\mathbb{E}[\frac{1}{2}\|x_* - x_k\|_L^2 - \frac{1}{2}\|x_* - x_{k+1}\|_L^2 | \mathcal{F}_k] = \frac{1}{2n}\|x_* - x_k\|_L^2 - \frac{1}{2n}\|x_* - \bar{x}_{k+1}\|_L^2,$$

so that

$$\begin{aligned} \mathbb{E}[F(x_{k+1}) | \mathcal{F}_k] \leq (1 - \frac{1}{n})g(x_k) + f(x_k) + \frac{1}{n}g(x_*) + \frac{1}{n}\langle \nabla f(x_k), x_* - x_k \rangle \\ + \frac{1}{2}\|x_* - x_k\|_L^2 - \frac{1}{2}\mathbb{E}[\|x_* - x_{k+1}\|_L^2 | \mathcal{F}_k]. \end{aligned}$$

We use the convexity of f:

$$\mathbb{E}[F(x_{k+1}) | \mathcal{F}_k] \leq (1 - \frac{1}{n})(g(x_k) + f(x_k)) + \frac{1}{n}(g(x_*) + f(x_*)) + \frac{1}{2}\|x_* - x_k\|_L^2 - \frac{1}{2}\mathbb{E}[\|x_* - x_{k+1}\|_L^2 | \mathcal{F}_k].$$

We rearrange and we apply total expectation:

$$\mathbb{E}[F(x_{k+1}) - F(x_*) + \frac{1}{2}\|x_* - x_{k+1}\|_L^2] \leq \mathbb{E}[(1 - \frac{1}{n})(F(x_k) - F(x_*)) + \frac{1}{2}\|x_* - x_k\|_L^2].$$

Summing for k from 0 to K-1 yields

$$\mathbb{E}[F(x_{K+1}) - F(x_*)] + \frac{1}{2}\mathbb{E}[\|x_* - x_{K+1}\|_L^2] + \sum_{k=1}^{K-1} \mathbb{E}[\frac{1}{n}(F(x_k) - F(x_*)] \leq (1 - \frac{1}{n})(F(x_0) - F(x_*)) + \frac{1}{2}\|x_* - x_0\|_L^2].$$

Using (6) and the fact that $\mathbb{E}[\|x_* - x_{K+1}\|_L^2] \geq 0$,

$$(1 + \frac{k}{n})\mathbb{E}[F(x_{K+1}) - F(x_*)] \leq (1 - \frac{1}{n})(F(x_0) - F(x_*)) + \frac{1}{2}\|x_* - x_0\|_L^2]$$

We just need to divide by $\frac{n}{k+1}$ to conclude.

Exercise 6 (Lasso). *Proximal coordinate descent is widely used to solve the Lasso problem given by*

$$\min_{x \in \mathbb{R}^p} \frac{1}{2}\|y - Zx\|_2^2 + \lambda\|x\|_1$$

Here, $f(x) = \frac{1}{2}\|y - Zx\|_2^2$ is differentiable while $g(x) = \lambda\|x\|_1$ is a non-differentiable function whose proximal operator is the soft-thresholding operator.

Exercise 7 (Multi-task Lasso). *In the multi-task framework, the Lasso problem can be generalised as*

$$\min_{x \in \mathbb{R}^{p \times q}} \frac{1}{2}\|Y - Zx\|_F^2 + \lambda \sum_{j=1}^{p} \|x_{j,:}\|_2.$$

Here, the optimisation variable is a $p \times q$ matrix. One can see that the nonsmooth part of the objective is $g(X) = \lambda \sum_{j=1}^{p} \|x_{j,:}\|_2$. This function is not separable when we consider the entries of x one by one but it is separable if we group these entries column-wise. Hence, we can consider block coordinate descent with p blocks of size q for the resolution of the multi-task Lasso problem.

Exercise 8 (ℓ_1/ℓ_2-regularised multinomial logistic regression). *Logistic regression is famous for classification problems. One observes for each $i \in [n]$ a class label $c_i \in \{1, \dots, q\}$ and a vector of features $z_i \in \mathbb{R}^p$. This information can be recast into a matrix $Y \in \mathbb{R}^{n \times q}$ filled by 0's and 1's: $Y_{i,k} = \mathbf{1}_{\{c_i = k\}}$. A matrix $B \in \mathbb{R}^{p \times q}$ is formed by q vectors encoding the hyperplanes for the linear classification. The multinomial ℓ_1/ℓ_2 regularized regression reads:*

$$\min_{B \in \mathbb{R}^{p \times q}} \sum_{i=1}^{n} \left(\sum_{k=1}^{q} -Y_{i,k} z_i^\top B_{:,k} + \log \left(\sum_{k=1}^{q} \exp\left(z_i^\top B_{:,k}\right) \right) \right) + \lambda \sum_{j=1}^{p} \|B_{j,:}\|_2,$$

Like the multi-task Lasso problem, this problem can be solved with proximal coordinate descent as long as we consider blocks of variables corresponding to the columns of B rather than single variables.

Exercise:

1. *Find the proximal operator of the non-smooth function $g(B) = \lambda \sum_{j=1}^{p} \|B_{j,:}\|_2$.*
2. *Give the expression of the partial derivatives of the smooth function*

$$f(B) = \sum_{i=1}^{n} \left(\sum_{k=1}^{q} -Y_{i,k} z_i^\top B_{:,k} + \log \left(\sum_{k=1}^{q} \exp\left(z_i^\top B_{:,k}\right) \right) \right)$$

3. *Give an estimate of the p block-wise Lipschitz constants of ∇f.*
4. *Write the proximal coordinate descent method for ℓ_1/ℓ_2-regularised multinomial logistic regression.*

4 Stochastic dual coordinate ascent for support vector machines

In this section, we focus on the linear Support Vector Machines (SVM) problem

$$\min_{w\in\mathbb{R}^p} C\sum_{i=1}^n \max(0, 1 - y_i z_i^\top w) + \frac{1}{2}\|w\|_2^2$$

where C is a positive real number, $y \in \mathbb{R}^n$ and $\forall i,\ z_i \in \mathbb{R}^p$. Note that we consider the formulation without intercept. The objective function contains a non-smooth and non-separable term so we cannot apply coordinate descent to it.

However, a dual formulation of the SVM problem is given by

$$\max_{\alpha\in\mathbb{R}^n} -\frac{1}{2}\sum_{j=1}^p \big(\sum_{i=1}^n Z_{i,j} y_i \alpha^{(i)}\big)^2 + \sum_{i=1}^n \alpha^{(i)} - \mathbb{I}_{[0,C]^n}(\alpha).$$

The objective function of this problem does decompose into a differentiable concave function $f(\alpha) = -\frac{1}{2}\sum_{j=1}^p \big(\sum_{i=1}^n Z_{i,j} y_i \alpha^{(i)}\big)^2 + \sum_{i=1}^n \alpha^{(i)}$ and a nonsmooth concave and separable function $g(\alpha) = -\mathbb{I}_{[0,C]^n}(\alpha)$. Stochastic Dual Coordinate Ascent (SDCA) is proximal coordinate ascent (the version of coordinate descent for concave functions) on this problem.

Exercise 9. *Write an implement of SDCA. It may be useful to maintain "residuals" w_k defined by $w_k^{(j)} = \sum_{i=1}^n Z_{i,j} y_i \alpha_k^{(i)}$ for all $j \in \{1, \dots, p\}$.*

Even if we are running the algorithm in the dual, we are interested in the primal problem. The following result shows that we can recover a good primal solution from the dual solution and gives theoretical guarantees for the convergence in the primal.

Theorem 7 ([Shalev-Shwartz and Zhang, 2013])**.** *Let us define a primal point $w_k = Z^\top Diag(y)\,\alpha_k$, where $(\alpha_k)_{k\geq 0}$ is generated by SDCA. The duality gap satisfies for all $K \geq n$,*

$$\mathbb{E}\Big[\frac{1}{K}\sum_{k=K}^{2K-1} P(w_k) - D(\alpha_k)\Big] \leq \frac{n}{K+n}\Big((1-\frac{1}{n})(D(\alpha_*) - D(\alpha_0)) + \frac{1}{2}\|\alpha_* - \alpha_0\|_L^2\Big) + \frac{n}{2K}C^2\sum_{i=1}^n L_i$$

where the primal value is $P(w_k) = C\sum_{i=1}^n \max(0, 1 - y_i z_i^\top w_k) + \frac{1}{2}\|w_k\|_2^2$, the dual value is $D(\alpha_k) = -\frac{1}{2}\|Z^\top Diag(y)\,\alpha_k\|_2^2 + \sum_{i=1}^n \alpha_k^{(i)} - \mathbb{I}_{[0,C]^n}(\alpha_k)$ and $\forall i, L_i = y_i^2\|z_i\|^2$.

PROOF. As SDCA solves the dual problem with coordinate ascent, by Theorem 6,

$$\mathbb{E}[D(\alpha_*) - D(\alpha_{k+1})] \leq \frac{n}{k+n}\Big((1-\frac{1}{n})(D(\alpha_*) - D(\alpha_0)) + \frac{1}{2}\|\alpha_* - \alpha_0\|_L^2\Big).$$

The goal of the theorem is to upper bound $\mathbb{E}[P(w_k) - D(\alpha_k)]$ by quantities involving $\mathbb{E}[D(\alpha_*) - D(\alpha_{k+1})]$. Note that by weak duality, $P(w_k) - D(\alpha_k) \geq D(\alpha_*) - D(\alpha_{k+1})$ but what we need is an inequality in the other way. For this, we will need to use the fact that $(\alpha_k)_{k\geq 0}$ is generated by the coordinate ascent method.

Using the feasibility of α_k and the definition of w_k, we can simplify $D(\alpha_{k+1})$ as

$$D(\alpha_{k+1}) = -\frac{1}{2}\|Z^\top \mathrm{Diag}\,(y)\,\alpha_{k+1}\|_2^2 + \sum_{i=1}^n \alpha_{k+1}^{(i)} - \mathbb{I}_{[0,C]^n}(\alpha_{k+1}) = -\frac{1}{2}\|w_{k+1}\|^2 + e^\top \alpha_{k+1}$$

As $\alpha_{k+1} = \alpha_k + U_{i_{k+1}}(\bar{\alpha}_{k+1}^{(i_{k+1})} - \alpha_k^{(i_{k+1})})$, $w_{k+1} = w_k + z_{i_{k+1}} y_{i_{k+1}}(\bar{\alpha}_{k+1}^{(i_{k+1})} - \alpha_k^{(i_{k+1})})$ and

$$D(\alpha_{k+1}) = -\frac{1}{2}\|w_k + z_{i_{k+1}} y_{i_{k+1}}(\bar{\alpha}_{k+1}^{(i_{k+1})} - \alpha_k^{(i_{k+1})})\|^2 + e^\top \alpha_k + \bar{\alpha}_{k+1}^{(i_{k+1})} - \alpha_k^{(i_{k+1})}$$

To simplify notations, we will write here $i = i_{k+1}$. Note that

$$\bar{\alpha}_{k+1}^{(i)} = \arg\max_{a\in[0,C]} (y_i z_i Z^\top \mathrm{Diag}\,(y)\,\alpha_k + 1)(a - \alpha_k^{(i)}) - \frac{\|y_i z_i\|^2}{2}(a - \alpha_k^{(i)})^2$$

$$= \arg\max_{a\in[0,C]} -\frac{1}{2}\|w_k + z_i y_i (a - \alpha_k^{(i)})\|^2 + a - \alpha_k^{(i)}.$$

So let us consider $\phi : x \mapsto C\max(0, 1-x)$, $u \in -\partial\phi(y_i z_i^\top w_k) \subseteq [0, C]$ and $s \in [0, 1]$.

$$\begin{aligned}
D(\alpha_{k+1}) &= \max_{a\in[0,C]} -\frac{1}{2}\|w_k + z_i y_i (a - \alpha_k^{(i)})\|^2 + e^\top \alpha_k + a - \alpha_k^{(i)} \\
&\geq -\frac{1}{2}\|w_k + z_i y_i ((su + (1-s)\alpha_k^{(i)}) - \alpha_k^{(i)})\|^2 + e^\top \alpha_k + (su + (1-s)\alpha_k^{(i)}) - \alpha_k^{(i)} \\
&\geq -\frac{1}{2}\|w_k + z_i y_i s(u - \alpha_k^{(i)})\|^2 + e^\top \alpha_k + s(u - \alpha_k^{(i)}) \\
&= -\frac{1}{2}\|w_k\|^2 - \frac{s^2}{2}\|z_i y_i\|_2^2 (u - \alpha_k^{(i)})^2 - s(u - \alpha_k^{(i)}) y_i z_i^\top w_k + e^\top \alpha_k + s(u - \alpha_k^{(i)}) \\
&= D(\alpha_k) - \frac{s^2}{2}\|z_i y_i\|_2^2 (u - \alpha_k^{(i)})^2 - s(u - \alpha_k^{(i)}) y_i z_i^\top w_k + s(u - \alpha_k^{(i)})
\end{aligned}$$

As $u \in -\partial\phi(y_i z_i^\top w_k) \subseteq [0, C]$ and $\phi^*(q) = q + \mathbb{I}_{[-C,0]}(q)$, Fenchel-Young equality leads to: $\phi(y_i z_i^\top w_k) - u = -u y_i z_i^\top w_k$. Hence,

$$D(\alpha_{k+1}) \geq D(\alpha_k) - \frac{s^2}{2}\|z_i y_i\|_2^2 (u - \alpha_k^{(i)})^2 + s\phi(y_i z_i^\top w_k) + s\alpha_k^{(i)} y_i z_i^\top w_k - s\alpha_k^{(i)}$$

Applying conditional expectation, we get

$$\mathbb{E}[D(\alpha_{k+1})|\mathcal{F}_k] \geq D(\alpha_k) - \frac{s^2}{2n}\sum_{i=1}^n \|z_i y_i\|_2^2 (u - \alpha_k^{(i)})^2 + \frac{s}{n}\sum_{i=1}^n \left(\phi(y_i z_i^\top w_k) + \alpha_k^{(i)} y_i z_i^\top w_k - \alpha_k^{(i)}\right)$$

Now,

$$\begin{aligned}
P(w_k) - D(\alpha_k) &= C\sum_{i=1}^n \max(0, 1 - y_i z_i^\top w_k) + \frac{1}{2}\|w_k\|_2^2 - (-\frac{1}{2}\|w_k\|^2 + e^\top \alpha_k) \\
&= \sum_{i=1}^n \phi(y_i z_i^\top w_k) + \alpha_k^{(i)} y_i z_i^\top w_k - \alpha_k^{(i)}
\end{aligned}$$

So that

$$\begin{aligned}
\mathbb{E}[D(\alpha_{k+1})|\mathcal{F}_k] - D(\alpha_k) &\geq -\frac{s^2}{2n}\sum_{i=1}^n \|z_i y_i\|_2^2 (u - \alpha_k^{(i)})^2 + \frac{s}{n}(P(w_k) - D(\alpha_k)) \\
&\geq -\frac{s^2}{2n}\sum_{i=1}^n (\|z_i y_i\|_2^2) C^2 + \frac{s}{n}(P(w_k) - D(\alpha_k))
\end{aligned}$$

where the last inequality derives from $\alpha_k^{(i)} \in [0, C]$ and $u \in [0, C]$.

We apply total expectation and we sum for k from K_1 to $K - 1$:

$$\begin{aligned}
\frac{s}{n}\sum_{k=K_1}^{K-1} \mathbb{E}[P(w_k) - D(\alpha_k)] &\leq \mathbb{E}[D(\alpha_K)] - \mathbb{E}[D(\alpha_{K_1})] + \frac{s^2}{2n} C^2 \sum_{i=1}^n (\|z_i y_i\|_2^2)(K - K_1) \\
&\leq \mathbb{E}[D(\alpha_*)] - \mathbb{E}[D(\alpha_{K_1})] + \frac{s^2}{2n} C^2 \sum_{i=1}^n (\|z_i y_i\|_2^2)(K - K_1) \\
&\leq \frac{c_0 n}{K_1 + n} + \frac{s^2}{2n} C^2 \sum_{i=1}^n (\|z_i y_i\|_2^2)(K - K_1)
\end{aligned}$$

where $c_0 = (1 - \frac{1}{n})(D(\alpha_*) - D(\alpha_0)) + \frac{1}{2}\|\alpha_* - \alpha_0\|_L^2$. Choosing $K = 2K_1$ and $s = \frac{n}{K_1}$, we obtain, for $K_1 \geq n$ (because we need $s \leq 1$)

$$\frac{1}{K_1}\sum_{k=K_1}^{2K_1-1} \mathbb{E}[P(w_k) - D(\alpha_k)] \leq \frac{c_0 n}{K_1 + n} + \frac{n}{2K_1} C^2 \sum_{i=1}^n (\|z_i y_i\|_2^2)$$

References

[Beck and Tetruashvili, 2013] Beck, A. and Tetruashvili, L. (2013). On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060.

[Collins et al., 2002] Collins, M., Schapire, R. E., and Singer, Y. (2002). Logistic regression, adaboost and bregman distances. *Machine Learning*, 48(1-3):253–285.

[Nesterov, 2012] Nesterov, Y. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362.

[Powell, 1976] Powell, M. (1976). Some global convergence properties of a variable metric algorithm for minimization without exact line searches. *Nonlinear programming*, 9:53–72.

[Richtárik and Takáč, 2014] Richtárik, P. and Takáč, M. (2014). Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1-2):1–38.

[Seidel, 1874] Seidel, L. (1874). Ueber ein verfahren, die gleichungen, auf welche die methode der kleinsten quadrate führt, sowie lineäre gleichungen überhaupt, durch successive annäherung aufzulösen:(aus den abhandl. dk bayer. akademie ii. cl. xi. bd. iii. abth. *Verlag der k. Akademie*.

[Shalev-Shwartz and Zhang, 2013] Shalev-Shwartz, S. and Zhang, T. (2013). Stochastic dual coordinate ascent methods for regularized loss. *The Journal of Machine Learning Research*, 14(1):567–599.

[Tseng, 2001] Tseng, P. (2001). Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494.

[Warga, 1963] Warga, J. (1963). Minimizing certain convex functions. *Journal of the Society for Industrial & Applied Mathematics*, 11(3):588–593.

[Wright, 2015] Wright, S. J. (2015). Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34.