Numerical Optimization 2025 - Project 3 (individual)

Maximum number of points: 20
Deadline: Sunday, August 31, 23:59.

I have set the deadline for August 31 so that you have flexibility regarding when you want to work on it. If you submit it earlier, please, *inform me about it via email* so that I can evaluate it earlier.

General description

The main goal of Project 3 is to implement and test 2 nonsmooth optimization algorithms:

- Forward-Backward (Proximal Gradient) method (FB)
- Projected Gradient method (PG) — a specialization of FB

For comparion with PG, you will also need the Active-Set Method (ASM) — just use your implementation from Project 2.

For information about the FB and PG, see Recording Lecture 10 from 19.05.2025 as well as the last section of the slides NumOptimPresentation — OtherTopics. (see also the Proximal Point Algorithm in there). I will also talk about P3 during the June 30 lecture.

You will apply these methods to again approximate the function $\sin(t)$ over $[-2\pi, 2\pi]$ using a polynomial of degree n. At the same time, however, you will encourage sparsity in the coefficient vector using Lasso-type formulations, which in turn leads to nonsmooth or constrained optimization problem.

Problem formulations

Just like in Project 1, the approximation is done with the polynomial

$$\phi(x;t) = \sum_{i=0}^{n} x_i t^i, \quad x \in \mathbb{R}^{n+1}$$

using $m = 100$ uniformly spaced sample points $a_j \in [-2\pi, 2\pi]$ and $b_j = \sin(a_j)$. Thus, the first (smooth) part of the objective function is

$$f(x) = \frac{1}{2} \sum_{j=1}^{m} \left(\phi(x; a_j) - b_j\right)^2 ,$$

which is a quadratic function in x. Let $A \in \mathbb{R}^{m\times(n+1)}$ be the matrix whose j-th row is $(1, a_j, a_j^2, \ldots, a_j^n)$, i.e.

$$A = \begin{bmatrix} 1 & a_1 & a_1^2 & \cdots & a_1^n \\ 1 & a_2 & a_2^2 & \cdots & a_2^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & a_m & a_m^2 & \cdots & a_m^n \end{bmatrix}$$

(the so called **Vandermonde matrix**) and $b \in \mathbb{R}^m$ be the vector with entries $b_j = \sin(a_j)$. Then the function f can be equivalently written in matrix-vector form as

$$f(x) = \frac{1}{2}\|Ax - b\|_2^2,$$

The gradient is given by $\nabla f(x) = A^\top(Ax - b)$ and the Hessian by $\nabla^2 f(x) = A^\top A$.

Note that ∇f is Lipschitz continuous with constant

$$L = \|A^\top A\|_2 = \sigma_{\max}(A)^2,$$

where $\sigma_{\max}(A)$ denotes the largest singular value of A = the largest eigenvalue of $A^\top A$.

With this notation, the problems we consider are:

(A) Penalized Lasso:

$$\min_{x\in\mathbb{R}^{n+1}} \frac{1}{2}\|Ax - b\|_2^2 + \lambda\|x\|_1,$$

where $\lambda > 0$ is a parameter.

(B) Constrained Lasso:

$$\min_{x\in\mathbb{R}^{n+1}} \frac{1}{2}\|Ax - b\|_2^2 \quad \text{subject to } \|x\|_1 \le 1$$

Tasks

Task 1: (4 pts) Implement the FB for the the penalized Lasso and the PG for the constrained Lasso.
For both problems/methods:

- write down the optimality conditions
- write down the algorithm
- detail the formulas for the prox/projection step (you can find hints for the prox step in the lecture; how to project onto the set $\|x\|_1 \le 1$ you need to figure out)

With this foundation, you can program both methods:

- step size: use the fixed step size $1/L$ for L = the largest eigenvalue of $A^\top A$
- stopping criterion: use a bound on number of iterations (e.g. 10^5) together with a suitable optimality measure
  See the definitions of proximal residual and projected gradient below — explain how are they related to optimality conditions on one hand and to the iteration of the FB/PG on the other; then use them to formulate a suitable stopping criterion.

Task 2: (4 pts) Apply the FB to the penalized Lasso and the PG as well as the AMS to the constrained Lasso.

For AMS, reformulate the constraints like in P2.

Try different sizes of n (up to 15), starting points, and parameters λ.

Report the number of iterates needed and the optimality measure. Plot the graphs of resulting approximations as well as of the function $\sin(t)$ over $[-2\pi, 2\pi]$. Based on the optimality measure and the plots, comment on how good is the approximation.

Task 3: (4 pts) Condition number of matrix A and pre-conditioning.

Vandermonde matrices are known to be *ill-conditioned*, especially for large degrees n or when the evaluation points a_j are close together, which can cause numerical instability in polynomial approximation problems.

- compute the condition number of A for various n
- pre-conditioning: divide each column of A by its norm and compute the condition number again
- find a diagonal matrix D such that the above operation (column normalization) corresponds to multiplying A with D^{-1}
- how to use the normalized version of A?
  Define new variables $\tilde{x} = Dx$ (i.e. $x = D^{-1}\tilde{x}$) and rewrite both Lasso problems in terms of $\tilde{x}$.
  Be carefull about how the constraints/nonsmooth part changes!

Task 4: (8 pts) Run the algorithms (PGD, FB, ASM) on the pre-conditioned problems.

- adjust all the algorithms so that they are applicable to the pre-conditioned problems (weighted l_1 norm)
- do plenty of testing — similar considerations as in Task 2 apply here.

Submission

Submit all codes as well as a pdf report.

Proximal residual and Projected gradient

For problems of the form

$$\min_{x \in \mathbb{R}^n} \; f(x) + g(x),$$

where f is smooth and g is convex (possibly non-smooth), the *proximal residual* at x^k is defined as

$$r^k := \frac{1}{\alpha}\left(x^k - \operatorname{prox}_{\alpha g}(x^k - \alpha \nabla f(x^k))\right).$$

In the special case when $g = \delta_C$ for a convex set C, the proximal residual is call the *projected gradient* and it reads

$$g^k := \frac{1}{\alpha}\left(x^k - \Pi_C\left(x^k - \alpha \nabla f(x^k)\right)\right),$$

where Π_C denotes the projection onto the set C.