Optimization for data science

Introduction to optimization

R. Flamary

Master Data Science, Institut Polytechnique de Paris

September 10, 2024

Objective of this course

Optimization in machine learning and data science

- All ML and data science methods rely on numerical optimization.
- Understanding the method $\equiv$ understanding the optimization problem.
- What is inside the black box of the skikit-learn `.fit()` function ?

Your objectives

- Recognize the properties of optimization problems.
- Understand the optimization problems in ML approaches.
- Know the theory behind the optimization algorithms.
- Find a proper algorithm for a given problem.
- Be able to implement an optimization algorithm.
- Model new optimization problems (new ML method).

Course organization

Information

- 6 ECTS, 12 x 3h30 + Exam
- From 11/09/24 to 08/01/25
- Teaching material :
  - Moodle : https://moodle.polytechnique.fr/course/view.php?id=20498
  - My website : https://remi.flamary.com/cours/optim_ds.html
- All student uploads projects on Moodle (emails not graded)
- Grading:
  - 30% 2/3 Labs with Jupyter notebooks (python)
  - 30% Final project as Jupyter notebook
  - 40% 3h final exam

Strong suggestion

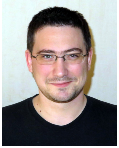
- Labs use jupyter notebooks, install locally or use Google Colab.
- Come with your laptop for all courses.
- Always try to do the small exercises : better understanding.
- Asks questions if you don't understand (interrupt if needed).

Course teaching staff

Professors

- **Alexandre Gramfort**
  Senior Research Scientist at Meta Reality Labs, Paris.
  Research topics: Machine learning, optimization, signal processing, deep learning, brain imaging.

- **Rémi Flamary**
  Professor at École Polytechnique, Palaiseau.
  Research topics: Machine learning, optimal transport, domain adaptation, signal processing.



Teaching assistants

- **Matthieu Terris**
  Postdoctoral researcher, INRIA Saclay, Mind team.
  Research topics: Optimization, image processing.

- **Joël Garde**
  PhD Student, Telecom Paris, S2A Team.
  Research topics: Optimization, optimal transport.

Full course overview

1. **Introduction to optimization for data science**
   1.1 ML optimization problems and linear algebra recap
   1.2 Optimization problems and their properties (Convexity, smoothness)
2. **Smooth optimization : Gradient descent**
   2.1 First order algorithms, convergence for smooth and strongly convex functions
3. **Smooth Optimization : Quadratic problems**
   3.1 Solvers for quadratic problems, conjugate gradient
   3.2 Linesearch methods
4. **Non-smooth Optimization : Proximal methods**
   4.1 Proximal operator and proximal algorithms
   4.2 Lab 1: Lasso and group Lasso
5. **Stochastic Gradient Descent**
   5.1 SGD and variance reduction techniques
   5.2 Lab 2: SGD for Logistic regression
6. **Standard formulation of constrained optimization problems**
   6.1 LP, QP and Mixed Integer Programming
7. **Coordinate descent**
   7.1 Algorithms and Labs
8. **Newton and quasi-newton methods**
   8.1 Second order methods and Labs
9. **Beyond convex optimization**
   9.1 Nonconvex reg., Frank-Wolfe, DC programming, autodiff

Current course overview

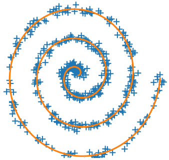
Machine learning and data science

Objective of Machine Learning (ML) and Data Science

Teach a machine to process automatically a large amount of data (signals, images, text, objects) in order to solve a given problem.

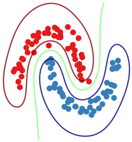
Unsupervised learning: Understanding the data.

- Clustering
- Probability Density Estimation
- Generative modeling
- Dimensionality reduction



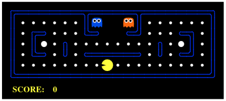
Supervised learning: Learning to predict.

- Classification
- Regression



Reinforcement learning: Learn from environment.

Train a machine to choose actions that maximize a reward (games, autonomous vehicles, control).



Optimization is at the core of all ML methods.

Empirical risk minimization

Supervised Machine learning

$$\min_f \quad \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i)) \tag{1}$$

- Find the function f that minimizes the average error L of prediction on a finite dataset of size N.
- Usually $f_{\boldsymbol{\theta}}$ is parametrized by $\boldsymbol{\theta} \in \mathbb{R}^n$ so the optimization is done *w.r.t.* θ.
- The objective above is called Empirical Risk Minimization, but beware of over-fitting when the model f is too complex.

Structural Risk Minimization [Vapnik, 2013]

$$\min_f \quad \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i)) + \lambda R(f) \tag{2}$$

- $R(f)$ is a regularization term that measure the complexity of f.
- λ is a regularization parameter that weight the regularization.

Least Square and ridge regression

Linear regression

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|^2+\lambda\frac{1}{2}\|\mathbf{x}\|^2$$

- Objective: predict a continuous value with a linear model (regression).
- Quadratic loss : $L(y, f(\mathbf{x})) = \frac{1}{2}(y - f(\mathbf{x}))^2$
- Quadratic regularization for Ridge : $R(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|^2$.
- Smooth and strictly convex problem when $\lambda > 0$.
- Can be solved by solving a linear problem (linear equations).

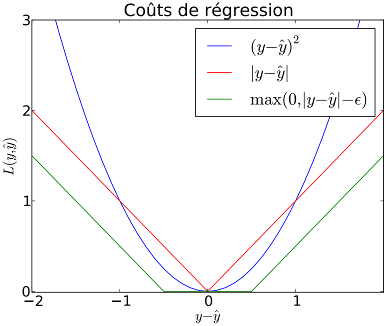
Non-linear regression

$$\min_{\boldsymbol{\theta}} \quad \frac{1}{N}\sum_{i=1}^{N}(y_i - f_{\boldsymbol{\theta}}(\mathbf{x}_i))^2$$

- Classical formulation for regression with neural networks.
- Can be non-convex and non-smooth depending on the architecture of $f_{\boldsymbol{\theta}}$.
- Harder to regularize (what is the complexity of $f_{\boldsymbol{\theta}}$?).

Data fitting for regression

Cost	$L(y, \hat{y})$
Square	$(y - \hat{y})^2$
Absolute value	$\lvert y - \hat{y} \rvert$
ϵ insensible	$\max(0, \lvert y - \hat{y} \rvert - \epsilon)$

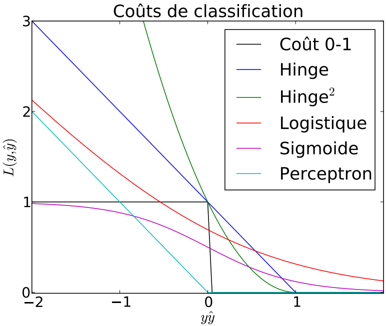


Regression problem

- **Objective**: predict a real value.
- Error if $y \neq \hat{y}$.
- **Error measure**: $|y - \hat{y}|$

Data fitting for binary classification

Cost	$L(y, \hat{y})$
0-1 loss	$(1 - \text{sgn}(y\hat{y}))/2$
Hinge	$\max(0, 1 - y\hat{y})$
Squared Hinge	$\max(0, 1 - y\hat{y})^2$
Logistic	$\log(1 + \exp(-y\hat{y}))$
Sigmoid	$(1 - \tanh(y\hat{y}))/2$
Perceptron	$\max(0, -y\hat{y})$



Classification problem

- **Objective**: predict a binary value.
- Error when $y \neq \text{signe}(\hat{y})$ i.e. if y and $\hat{y}$ have a different sign.
- **Error measure**: $y\hat{y}$
- Non symmetric loss.
- Multi-class classification with Softmax output and categorical cross-entropy.

Maximum Likelihood estimation

Maximum likelihood principle

- p_θ is a probability distribution in $\mathbb{R}^d$.
- We have access to samples $\mathbf{x}_i$ drawn I.I.D. from the distribution.
- The likelihood for independent samples can be expressed as

$$\prod_i p_\theta(\mathbf{x}_i)$$

- The maximum likelihood estimator of θ

$$\hat{\theta} = \arg\max_\theta \prod_i p_\theta(\mathbf{x}_i)$$

- In practice one can minimize the negative log-likelihood

$$\hat{\theta} = \arg\min_\theta -\sum_i \log(p_\theta(\mathbf{x}_i))$$

  That is a special case of empirical risk minimization (least square, logistic regression).

Sparsity and variable selection

Variable selection

- In supervised learning variable section aim at finding a subset $I \in \{1, \ldots, n\}$ of all variables that leads to a good prediction.
- It is a combinatorial problem *w.r.t.* the number of variables n.
- There is a compromise between number of variables and performance.

Sparsity and linear model

For a linear model the sparsity prior can be expressed as two optimization problems

$$\min_{\mathbf{x}} \; L(\mathbf{H}\mathbf{x}, \mathbf{y}) + \lambda \|\mathbf{x}\|_0 \qquad \text{or} \qquad \min_{\mathbf{x}, \|\mathbf{x}\|_0 \leq \tau} L(\mathbf{H}\mathbf{x}, \mathbf{y})$$

- $\lambda \geq 0$ and $\tau \geq 0$ are regularization parameters.
- $\|\mathbf{x}\|_0 = \sum_i 1_{|x_i|>0}$ is the number of components in $\mathbf{x}$.
- The problem can be reformulated as a Mixed Integer Program.
- Often a continuous approximation of the problem is solved (Lasso).

Lasso estimator

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|^2 + \lambda \sum_{k=1}^{d} |x_k| \tag{3}$$

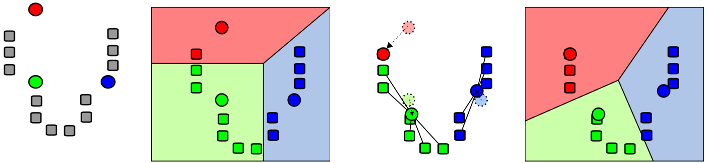
Optimization problem

- $\|\mathbf{x}\|_1 = \sum_{k=1}^{d} |x_k|$ is the L1 norm of vector $\mathbf{w}$.
- Objective function is non differentiable in $x_k = 0, \forall k$.
- For a large enough λ the solution of the problem is sparse.
- The problem is equivalent to

$$\min_{\mathbf{x}, \|\mathbf{x}\|_1 \leq \mu} \quad \frac{1}{2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|^2 \tag{4}$$

I.e. there exists a μ that leads to the same solution of the problem for a given λ.

K-means clustering

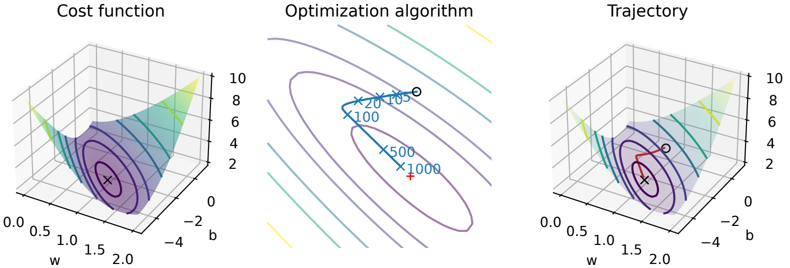


- Non convex Optimization problem:

$$\min_{\bar{\mathbf{x}}_k, \forall_k} \quad \sum_{i=1}^{N} \min_k \|\bar{\mathbf{x}}_k - \mathbf{x}_i\|^2$$

- Very simple algorithm :
  1. Update cluster membership (find closest $\bar{\mathbf{x}}_k$ for each samples)
  2. Update cluster positions $\bar{\mathbf{x}}_k$ as mean of all cluster members.
- Decrease the objective value at each iteration (can be formulated as block coordinate descent).

Optimization for data science



ML and DS are built on numerical optimization

- Most ML methods are optimization problems.
- The objective function is the error of the model.
- Importance of the optimization algorithm.

Important questions

- What are the properties of the optimization problem (F, constraints)?
- How to find the good algorithm for a given problem (no free lunch)?
- What are the properties on an algorithm (convergence, complexity)?
- How to implement an optimization algorithm (speed, scaling)?

Section

Linear Algebra recap

Notation: vectors and matrices

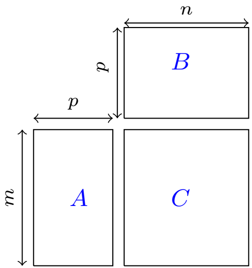
- A vector $\mathbf{x} \in \mathbb{R}^n$ is a column of n real numbers (always column).
- A matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a table of m rows and n columns.
- The i-th row of $\mathbf{A}$ is denoted $\mathbf{A}_{i,:}$ and the j-th column $\mathbf{A}_{:,j}$.
- The transpose of $\mathbf{A}$ is denoted $\mathbf{A}^\top$: $\mathbf{C} = \mathbf{A}^\top \quad \Leftrightarrow \quad c_{i,j} = a_{j,i}$
- Matrix addition and multiplication are defined as for real numbers.

Matrix product

- The product of $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times p}$ is

$$\mathbf{C} = \mathbf{AB} \in \mathbb{R}^{m \times p}$$

- The element $c_{i,j}$ of $\mathbf{C}$ is : $c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j}$
- Matrix product is not commutative ($\mathbf{AB} \neq \mathbf{BA}$).
- Special case with $\mathbf{B} = \mathbf{b}$ a vector, $\mathbf{Ab}$ is a linear combination of the columns of $\mathbf{A}$.



Linear map and properties

Linear maps

- A linear map (or linear function) $f_{\mathbf{A}} : \mathbb{R}^n \to \mathbb{R}^m$ can be expressed as

$$f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x}$$

- $\mathbf{A} \in \mathbb{R}^{m \times n}$ is the matrix of the linear map.

Properties and rank

- The image (or range) of $\mathbf{A}$ is the space spanned by the columns of $\mathbf{A}$:

$$\mathrm{im}(\mathbf{A}) = \{\mathbf{y} \in \mathbb{R}^m | \mathbf{y} = \mathbf{A}\mathbf{x},\ \forall \mathbf{x} \in \mathbb{R}^n\}$$

- The kernel (or null space) of $\mathbf{A}$ is the set of vectors $\mathbf{x}$ such that $\mathbf{A}\mathbf{x} = 0$.

$$\mathrm{ker}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{R}^n \ \mathbf{A}\mathbf{x} = 0\}$$

- The rank of $\mathbf{A}$ is the dimension of its range : $\mathrm{rank}(\mathbf{A}) = \mathrm{dim}(\mathrm{im}(\mathbf{A}))$
- $\mathrm{rank}(\mathbf{A}) \leq \min(m, n)$ and we have

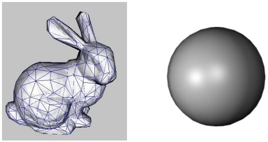
$$\mathrm{dim}(\mathrm{ker}(\mathbf{A})) + \mathrm{rank}(\mathbf{A}) = n$$

.

Linear maps in real life applications

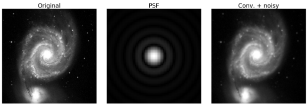
Computer graphics

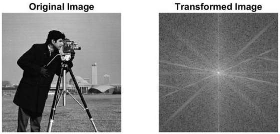
- Rotation/deformation of objects.
- Illumination of objects.
- Projection of 3D objects on 2D screen.



Signal and image processing

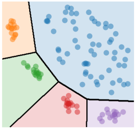
- Many physical processes are linear (wave propagation, optics, filtering).
- Observed signals (astronomy, medial imaging).
- Fourier Transform, Wavelet Transform.
- Inverse problems/reconstruction to cancel map.

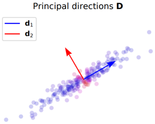




Machine learning

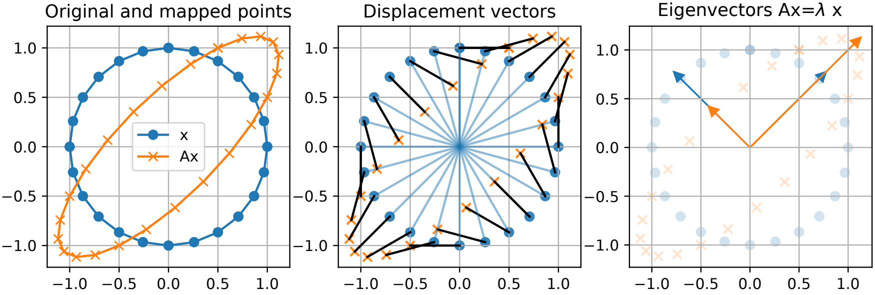
- Linear regression and classification (logistic).
- Neural networks layers.
- Principal Component Analysis (PCA).





Dedicated processors (DSP, GPU, TPU) are optimized for linear algebra operations.

Eigenvalues and linear operators



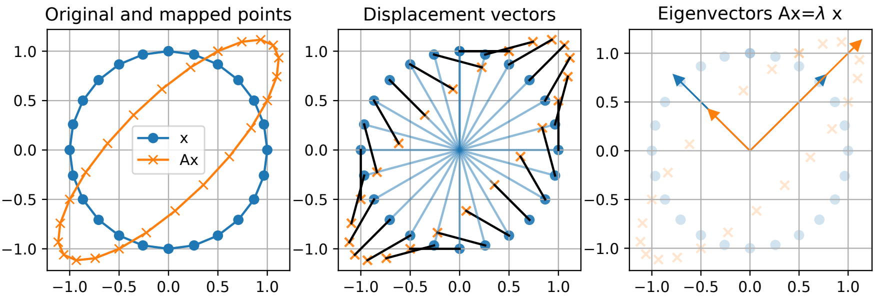
Eigenvalues and eigenvectors

- A vector $\mathbf{x} \in \mathbb{R}^n$ is an eigenvector of a square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ if

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$$

- λ is the eigenvalue associated with eigenvector $\mathbf{x}$.
- The eigenvectors of $\mathbf{A}$ are the solutions of the equation $(\mathbf{A} - \lambda \mathbf{I})\mathbf{x} = 0$.
- The eigenvectors of a symmetric matrix are orthogonal.
- If $\mathbf{x}$ is an eigenvector then $-\mathbf{x}$ and $\alpha\mathbf{x}$ are also eigenvectors.

Spectral theorem and eigendecomposition



Spectral Theorem

Le $\mathbf{A} = \mathbf{A}^\top$ a symmetric matrix, then there exists a basis of eigenvectors $\mathbf{x}_i \in \mathbb{R}^n$ and their sorted eigenvalues λ_i $(\lambda_1 \leq \cdots \leq \lambda_n)$ of $\mathbf{A}$ such that

1. **Orthogonality**: $\mathbf{x}_i^\top \mathbf{x}_j = 0$ for $i \neq j$.
2. **Unit norm**: $\|\mathbf{x}_i\| = 1$.
3. **Eigenvalues**: $\mathbf{A}\mathbf{x}_i = \lambda_i \mathbf{x}_i$.

The matrix $\mathbf{A}$ can be decomposed as $\mathbf{A} = \sum_i \lambda_i \mathbf{x}_i \mathbf{x}_i^\top$.

Matrix formulation of the spectral theorem

Matrix formulation

- Let $\mathbf{U} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ the matrix of eigenvectors of $\mathbf{A}$.
- Let $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_n]^\top$ the vector of sorted eigenvalues.
- Let $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_n) = \text{diag}(\boldsymbol{\lambda})$ the diagonal matrix of eigenvalues.
- **Ortogonality+unit norm** : $\mathbf{U}^\top\mathbf{U} =$
- **Eigenvalue**: $\mathbf{AU} = \mathbf{U}\boldsymbol{\Lambda}$.
- **Eigendecomposition/reconstruction** :

Imlementation in Python

- Decomposition : lambd,U = numpy.linalg.eig(A)
- Reconstruction :
- For symmetric matrices : Use numpy.linalg.eig.

Matrix formulation of the spectral theorem

Matrix formulation

- Let $\mathbf{U} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ the matrix of eigenvectors of $\mathbf{A}$.
- Let $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_n]^\top$ the vector of sorted eigenvalues.
- Let $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_n) = \text{diag}(\boldsymbol{\lambda})$ the diagonal matrix of eigenvalues.
- **Ortogonality+unit norm** : $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_n = \mathbf{U}\mathbf{U}^\top$ (orthonormality).
- **Eigenvalue**: $\mathbf{A}\mathbf{U} = \mathbf{U}\boldsymbol{\Lambda}$.
- **Eigendecomposition/reconstruction** :

Imlementation in Python

- Decomposition : lambd,U = numpy.linalg.eig(A)
- Reconstruction :
- For symmetric matrices : Use numpy.linalg . eig .

Matrix formulation of the spectral theorem

Matrix formulation

- Let $\mathbf{U} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ the matrix of eigenvectors of $\mathbf{A}$.
- Let $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_n]^\top$ the vector of sorted eigenvalues.
- Let $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_n) = \text{diag}(\boldsymbol{\lambda})$ the diagonal matrix of eigenvalues.
- **Ortogonality+unit norm** : $\mathbf{U}^\top \mathbf{U} = \mathbf{I}_n = \mathbf{U}\mathbf{U}^\top$ (orthonormality).
- **Eigenvalue**: $\mathbf{A}\mathbf{U} = \mathbf{U}\boldsymbol{\Lambda}$.
- **Eigendecomposition/reconstruction** :

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$$

Imlementation in Python

- Decomposition : lambd,U = numpy.linalg.eig(A)
- Reconstruction : Arec = U @ numpy.diag(lambd) @ U.T
- For symmetric matrices : Use numpy.linalg.eig.

Singular Value Decomposition (SVD)

SVD Decomposition

Let $\mathbf{A} \in \mathbb{R}^{n\times p}$ a non-square matrix with $n > p$, then there exists $\mathbf{U} \in \mathbb{R}^{n\times p}$ and $\mathbf{V} \in \mathbb{R}^{p\times p}$ such that

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

- **Ortogonality** : $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_p$ and $\mathbf{V}^\top\mathbf{V} = \mathbf{I}_p$.
- **Singular values** : $\boldsymbol{\sigma} = [\sigma_1, \ldots, \sigma_p]^\top$ with $0 \leq \sigma_1 \leq \cdots \leq \sigma_p$, $\mathbf{\Sigma} = \text{diag}(\boldsymbol{\sigma})$.
- **Other properties and tools** :
  - The columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{A}^\top\mathbf{A}$.
  - Low rank approximation : $\mathbf{A} \approx \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^\top$ from k largest singular values.
  - Sparse SVD : low rank approximation from sparse matrix (missing data).

Implementation in Python

- Decomposition : U,s,Vt = numpy.linalg.svd(A)
- Reconstruction : Arec = U @ numpy.diag(s) @ Vt
- Sparse SVD : Uk,sk,Vkt = scipy.sparse.linalg.svds(A,k)

Matrix norms

Norms

- **Frobenius norm** : $\|\mathbf{A}\|_F = \sqrt{\sum_{i,j} a_{i,j}^2} = \|\text{vec}(\mathbf{A})\|$.
- **Spectral norm** : $\|\mathbf{A}\|_2 = \max_{\mathbf{x}\neq 0} \frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_2} = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2$.
- **Norm induced by** $\|\cdot\|_p$: $\|\mathbf{A}\|_p = \max_{\|\mathbf{x}\|_p} \|\mathbf{A}\mathbf{x}\|_p$.

Relation with SVD

- $\|\mathbf{A}\|_F = \sqrt{\sum_i \sigma_i^2(\mathbf{A})}$
- $\|\mathbf{A}\|_2 = \max_i \sigma_i(\mathbf{A}) = \sigma_{\text{max}}(\mathbf{A}) = \|\boldsymbol{\sigma}(\mathbf{A})\|_\infty$.
- **Nuclear norm** : $\|\mathbf{A}\|_* = \sum_i \sigma_i(\mathbf{A}) = \|\boldsymbol{\sigma}(\mathbf{A})\|_1$.

Implementation in Python

- Frobenius norm : numpy.linalg.norm(A,'fro')
- Spectral norm : numpy.linalg.norm(A,2)
- Nuclear norm : numpy.linalg.norm(A,'nuc')

Numerical optimization problem

Problem formulation

$$\min_{\mathbf{x}\in\mathcal{C}} \quad F(\mathbf{x}) \tag{5}$$

- F is the objective function (sometimes called cost function).
- $\mathbf{x} = [x_1, \dots, x_n]^\top \in \mathbb{R}^n$ is a vector of n variables.
- $\mathcal{C} \subseteq \mathbb{R}^n$ is the set of admissible solutions.
- Objective : Find a solution $\mathbf{x}^\star \in \mathcal{C}$, having the minimal value for F such that

$$F(\mathbf{x}^\star) \leq F(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{C}.$$

Assumptions (in this course)

- The problem is proper (there exists a solution), F is lower bounded on $\mathcal{C}$.
- You have access to F and $\mathcal{C}$ (mathematical expression, no black box).

Notation : Lowercase bold is a vector, Uppercase bold is a matrix.

Standard constrained optimization

Problem reformulation

$$\begin{array}{lll} \min\limits_{\mathbf{x}\in\mathbb{R}^n} & F(\mathbf{x}) & \\ \text{with} & h_j(\mathbf{x}) = 0 & \forall j = 1,\dots,p \\ \text{and} & g_i(\mathbf{x}) \leq 0 & \forall i = 1,\dots,q. \end{array} \tag{6}$$

- This problem is equivalent to (27) when $\mathcal{C}$ can be expressed as

$$\mathcal{C} = \left\{\mathbf{x} \in \mathbb{R}^n \;\middle|\; h_j(x) = 0,\ \forall j = 1,\dots,p \text{ and } g_i(x) \leq 0,\ \forall i = 1,\dots,q\right\}.$$

- h_j and g_i define respectively the equality and inequality constraints.
- When $p = q = 0$ the problem is said to be unconstrained and $\mathcal{C} = \mathbb{R}^n$.
- The complexity of solving problems (27) and (6) depends on the properties of F and $\mathcal{C}$.
- Problem above is a standard formulation for constrained optimization.

Definitions

$$\min_{\mathbf{x}\in\mathcal{C}} \quad F(\mathbf{x})$$

Feasible point

Any point $\mathbf{x} \in \mathcal{C}$ that satisfies the constraints in set $\mathcal{C}$.

Optimal value

Minimal value function on the feasible set $\mathcal{C}$, often denoted as $F^\star$.

Optimality/Optimal solution

$\mathbf{x}^\star \in \mathcal{C}$ is a solution of the optimization problem if satisfies the constraints in set $\mathcal{C}$ and

$$F(\mathbf{x}^\star) \leq F(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{C}.$$

$\mathbf{x}^\star$ might not be unique in the general case.

Sub-optimal point

$\mathbf{x} \in \mathcal{C}$. is an ϵ-suboptimal point of the problem for $\epsilon > 0$ if

$$F(\mathbf{x}) \leq F(\mathbf{x}^\star) + \epsilon$$

Active constraint

g_i is considered an active constraint in $\mathbf{x}$ if $g_i(\mathbf{x}) = 0$.

Exercise 1: Positive least squares reformulation

Problem

$$\min_{\mathbf{x} \geq \mathbf{0}} \quad \|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2, \qquad \text{with } \mathbf{y} \in \mathbb{R}^m \text{ and } \mathbf{H} \in \mathbb{R}^{m \times n}$$

Exercise

1. Express $F(\mathbf{x})$ and $\mathcal{C}$ for the problem above.

Exercise 1: Positive least squares reformulation

Problem

$$\min_{\mathbf{x}\geq\mathbf{0}} \quad \|\mathbf{y}-\mathbf{H}\mathbf{x}\|^2, \qquad \text{with } \mathbf{y}\in\mathbb{R}^m \text{ and } \mathbf{H}\in\mathbb{R}^{m\times n}$$

Exercise

1. Express $F(\mathbf{x})$ and $\mathcal{C}$ for the problem above.

$$F(\mathbf{x}) = \|\mathbf{y}-\mathbf{H}\mathbf{x}\|^2, \quad \mathcal{C} = \{\mathbf{x}\in\mathbb{R}^n | x_i \geq 0\ \forall i\} = \mathbb{R}^{+n}$$

2. Find p, q the number of constraints :

Exercise 1: Positive least squares reformulation

Problem

$$\min_{\mathbf{x}\geq\mathbf{0}} \quad \|\mathbf{y}-\mathbf{H}\mathbf{x}\|^2, \qquad \text{with } \mathbf{y}\in\mathbb{R}^m \text{ and } \mathbf{H}\in\mathbb{R}^{m\times n}$$

Exercise

1. Express $F(\mathbf{x})$ and $\mathcal{C}$ for the problem above.

$$F(\mathbf{x}) = \|\mathbf{y}-\mathbf{H}\mathbf{x}\|^2, \quad \mathcal{C} = \{\mathbf{x}\in\mathbb{R}^n | x_i \geq 0 \ \forall i\} = \mathbb{R}^{+n}$$

2. Find p, q the number of constraints : p=0, q=n
3. Express h_j and g_i if there are somme constraints:

Exercise 1: Positive least squares reformulation

Problem

$$\min_{\mathbf{x}\geq\mathbf{0}} \quad \|\mathbf{y}-\mathbf{H}\mathbf{x}\|^2, \qquad \text{with } \mathbf{y}\in\mathbb{R}^m \text{ and } \mathbf{H}\in\mathbb{R}^{m\times n}$$

Exercise

1. Express $F(\mathbf{x})$ and $\mathcal{C}$ for the problem above.

$$F(\mathbf{x}) = \|\mathbf{y}-\mathbf{H}\mathbf{x}\|^2, \quad \mathcal{C} = \{\mathbf{x}\in\mathbb{R}^n | x_i \geq 0 \; \forall i\} = \mathbb{R}^{+n}$$

2. Find p, q the number of constraints : p=0, q=n
3. Express h_j and g_i if there are somme constraints:

$$g_i(\mathbf{x}) = -x_i, \quad \forall i \in 1,\ldots,n$$

Numerical optimization algorithm

$$\min_{\mathbf{x}\in\mathcal{C}} \quad F(\mathbf{x})$$

Iterative optimization algorithm

An iterative algorithm A is an algorithm providing a series $\mathbf{x}^{(k)}$ for $k = 0, 1, \dots$ of iterates $\mathbf{x}^{(k+1)} = A(\mathbf{x}^{(k)})$ that converges to a solution $\mathbf{x}^\star$ of the optimization problem starting from an initial guess $\mathbf{x}^{(0)}$.

- If $F(\mathbf{x}^{(k+1)}) \leq F(\mathbf{x}^{(k)}),\ \forall k$ then it is called a **descent algorithm**.
- In practice iterations are stopped when a convergence criterion is met.

Convergence of iterative methods

- The convergence speed can be expressed in objective value

$$|F(\mathbf{x}^{(k+1)}) - F(\mathbf{x}^\star)| \leq \gamma |F(\mathbf{x}^{(k)}) - F(\mathbf{x}^\star)|^q \tag{7}$$

- Or it can be expressed in terms of iterates:

$$\|\mathbf{x}^{(k+1)} - \mathbf{x}^\star\| \leq \gamma \|\mathbf{x}^{(k)} - \mathbf{x}^\star\|^q \tag{8}$$

where $\gamma \in [0, 1)$ and $q \geq 1$ is the convergence order ($q = 1$ linear, $q = 2$ quadratic...).

Properties of optimization problems

Know your optimization problem (and its properties)

- They with guide you toward the proper solver.
- They tell you how much you can trust the solution (well posed, unique solution).
- They will help you design the optimization problem.

Convexity

- Well posed problem.
- Unique solution when strict convexity.

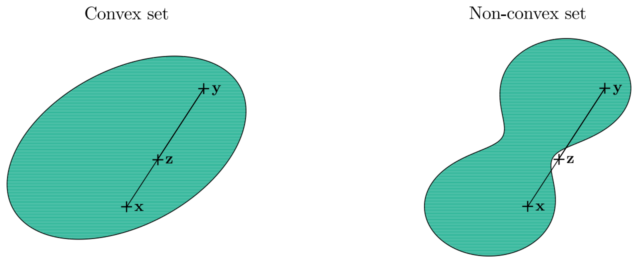
Smoothness

- Continuity, differentiability
- When function smooth, one can use its gradients.

Solutions

- What is a solution of the optimization problem ?
- Criterions for reaching a solution (stopping the algorithm).

Convex set



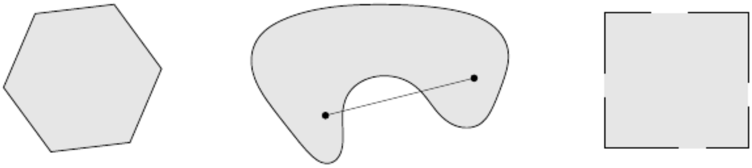
Definition: Convex set

$\mathcal{C} \subset \mathbb{R}^n$ is a convex set if for any two points $\mathbf{x}, \mathbf{y} \in \mathcal{C}^2$ and for any $0 \leq \alpha \leq 1$ we have

$$\alpha \mathbf{x} + (1 - \alpha)\mathbf{y} \in \mathcal{C}$$

Image from [Boyd and Vandenberghe, 2004]

Convex set



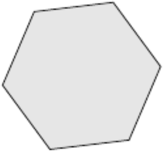
Definition: Convex set

$\mathcal{C} \subset \mathbb{R}^n$ is a convex set if for any two points $\mathbf{x}, \mathbf{y} \in \mathcal{C}^2$ and for any $0 \leq \alpha \leq 1$ we have

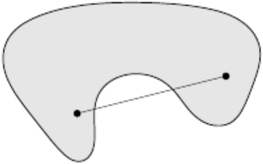
$$\alpha \mathbf{x} + (1 - \alpha)\mathbf{y} \in \mathcal{C}$$

Image from [Boyd and Vandenberghe, 2004]

Convex set



Convex



Non convex

Non convex

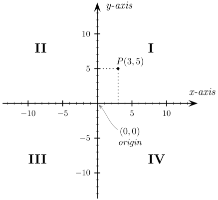
Definition: Convex set

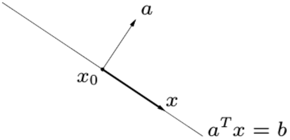
$\mathcal{C} \subset \mathbb{R}^n$ is a convex set if for any two points $\mathbf{x}, \mathbf{y} \in \mathcal{C}^2$ and for any $0 \leq \alpha \leq 1$ we have

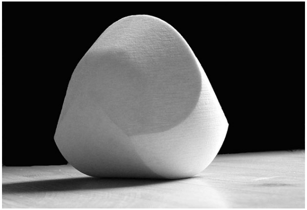
$$\alpha \mathbf{x} + (1 - \alpha)\mathbf{y} \in \mathcal{C}$$

Image from [Boyd and Vandenberghe, 2004]

Examples of convex sets



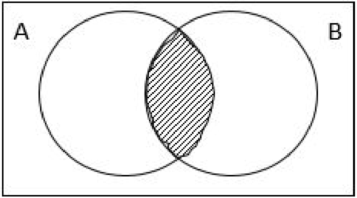


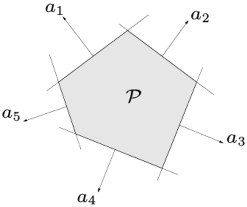


Examples

- $\mathbb{R}^n$
- Positive orthant of $\mathbb{R}^n$: $\mathbb{R}^n_+$.
- Hyperplan : $\{\mathbf{x} \in \mathbb{R}^d : \mathbf{a}^\top \mathbf{x} = b\}$
- Half space: $\{\mathbf{x} \in \mathbb{R}^d : \mathbf{a}^\top \mathbf{x} \leq b\}$
- Polyhedra: $\{\mathbf{x} \in \mathbb{R}^d : \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$
- Gömböc

Operations on set preserving convexity (1)





Intersection

If $\mathcal{X}_k$ are convex set $\forall k$ then their intersection

$$\bigcap_{k=1}^{K} \mathcal{X}_k$$

is also convex.

Operations on set preserving convexity (2)

Cartesian product

If $\mathcal{X}_k \subset \mathbb{R}^{n_k}$, are convex $\forall k = 1, \cdots, M$ then

$$\mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \times \mathcal{X}_M = \{(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M) : \mathbf{x}_k \in \mathcal{X}_k\}$$

is convex.

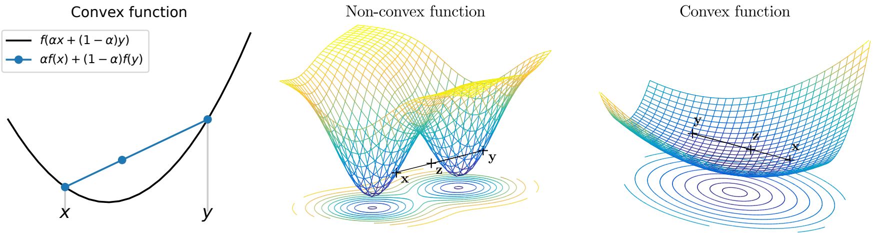
Affine transform

If $\mathcal{X} \subset \mathbb{R}^d$ is convex and $\mathcal{A}(\mathbf{x}) \mapsto \mathbf{A}\mathbf{x} + \mathbf{b}$ is an affine transform defined by matrix $\mathbf{A} \in \mathbb{R}^{p \times d}$ and vector $\mathbf{b}$ then

$$\mathcal{A}(\mathcal{X}) = \{\mathcal{A}(\mathbf{x}) : \mathbf{x} \in \mathcal{X}\}$$

is convex. These transformations include translation and rotations.

Convex function



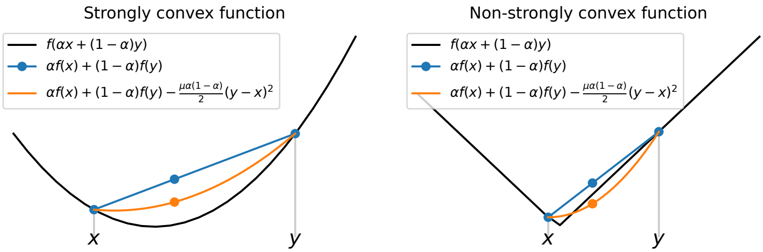
Definition: Convex function

A function F is said to be convex if it lies below its chords, that is $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

$$F(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha F(\mathbf{x}) + (1-\alpha)F(\mathbf{y}), \text{ with } 0 \leq \alpha \leq 1. \tag{9}$$

- A function is said to be strictly convex when the two inequalities are strict.
- Strict convexity implies that the function has a unique minimum.
- If a function F is convex, then the set $\{\mathbf{x} \in \mathbb{R}^n \mid F(\mathbf{x}) \leq 0\}$ is convex.
- A function F is concave if $-F$ is convex.

Strongly convex function



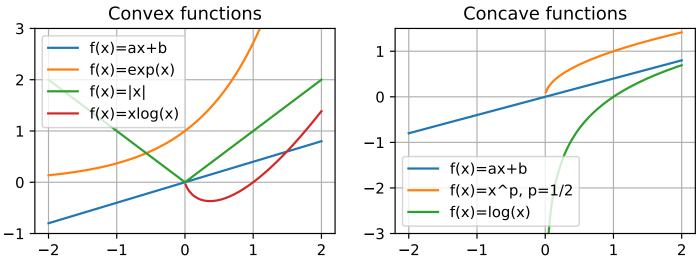
Definition: strong convexity

A function F is said to be μ-strongly convex with $\mu > 0$ if it satisfies $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and $0 \leq \alpha \leq 1$

$$F(\alpha \mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha F(\mathbf{x}) + (1-\alpha)F(\mathbf{y}) - \frac{\mu}{2}\alpha(1-\alpha)\|\mathbf{x}-\mathbf{y}\|^2, \tag{10}$$

- A μ-strongly convex function is convex and has a unique minimum.
- A μ-strongly convex function is upper bounded by a convex quadratic function.

Examples of functions in $\mathbb{R}$



Convex functions

- Affine functions : $x \mapsto ax + b$ for all $a, b \in \mathbb{R}$.
- Exponential functions : $x \mapsto e^{ax}$ for all $a \in \mathbb{R}$.
- Power of absolute value : $x \mapsto |x|^p$,for all $p \geq 1$.
- Neg-entropy : $x \mapsto x \log x$ for $x > 0$

Concave Functions

- Affine functions : $x \mapsto ax + b$ for all $a, b \in \mathbb{R}$.
- Power : $x \mapsto x^p$, for $x > 0$ and for all $0 \leq p \leq 1$.
- Logarithm : $x \mapsto \log x$ for $x > 0$

Operations preserving convexity (1)

Positive sum

Let $\lambda_1, \lambda_2 \geq 0$ and $f_1,\ f_2$ two convex function then

$$\lambda_1 f_1 + \lambda_2 f_2$$

is convex.

Composition with affine function

let $\mathbf{A} \in \mathbb{R}^{p \times d}$ and $b \in \mathbb{R}^p$ and $f : \mathbb{R}^p \mapsto \mathbb{R}$ be a convex function, the the composition

$$f(\mathbf{A}\mathbf{x} + b)$$

is convex

Example

- Log barrier : $f(\mathbf{x}) = -\sum_{i=1}^{m} \log(b_i - \mathbf{a}_i^\top \mathbf{x})$ with $\mathrm{dom} f = \{\mathbf{x} : \mathbf{a}_i^\top \mathbf{x} \leq b_i\}$
- Norm of an affine function : $f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} + b\|$

Operations preserving convexity (2)

Composition

- let $g : \mathbb{R}^d \mapsto \mathbb{R}$ be a convex function and $h : \mathbb{R} \mapsto \mathbb{R}$ be a convex and increasing function, then

$$f(\mathbf{x}) = h(g(\mathbf{x}))$$

  is convex.

Maximum

- If $f_1, \cdots, f_m$ are convex functions then

$$f(\mathbf{x}) = \max_i \{f_1(\mathbf{x}), \cdots, f_m(\mathbf{x})\}$$

  is convex.

Example

- Piecewise linear function: $f(\mathbf{x}) = \max_{i=1,\cdots,m}(\mathbf{a}_i^\top \mathbf{x} + b)$

Convexity in optimization

Convex optimization problem

$$\min_{\mathbf{x} \in \mathcal{C}} \quad F(\mathbf{x})$$

- The problem is convex if F is a convex function and $\mathcal{C}$ is a convex set.
- Any local minimizer of a convex function is a global minimizer.
- If the function is strictly convex the minimizer is unique.
- Maximizing a concave function under convex constraints is a convex problem.

Disciplined Convex Programming [Grant et al., 2006]

- Express the objective function and constraints as combination and composition of operations preserving convexity.
- Allows for designing generic solvers (Matlab [Grant and Boyd, 2014], Python [Diamond and Boyd, 2016]).

Smoothness and continuity

Differentiability classes in 1D

Let f be a real function. Then f is of differentiability class C^k if and only if $\frac{d^k f(x)}{dx^k}$ is continuous.

- C^0 is the set of continuous real functions.
- C^1 is the set of real functions with continuous derivatives.
- C^2 is the set of real functions with continuous second derivatives.

Exercise 2: Differentiability and convexity

Function	Diff. Class	Convexity
$f(x) = x^2$		
$f(x) = e^x$		
$f(x) = \|x\|$		
$f(x) = \max(x, 0)$		
$f(x) = \mathrm{sign}(x)$		
$f(x) = \log(1 + \exp(x))$		
$f(x) = 2x + 1$		
$f(x) = \max(x, 0)^2$		

Smoothness and continuity

Differentiability classes in 1D

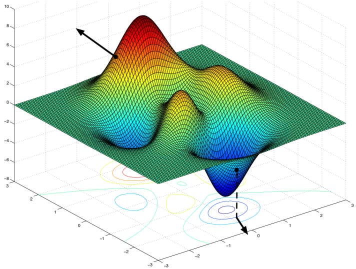
Let f be a real function. Then f is of differentiability class C^k if and only if $\frac{d^k f(x)}{dx^k}$ is continuous.

- C^0 is the set of continuous real functions.
- C^1 is the set of real functions with continuous derivatives.
- C^2 is the set of real functions with continuous second derivatives.

Exercise 2: Differentiability and convexity

Function	Diff. Class	Convexity
$f(x) = x^2$	C^∞	✓
$f(x) = e^x$	C^∞	✓
$f(x) = \lvert x\rvert$	C^0	✓
$f(x) = \max(x, 0)$	C^0	✓
$f(x) = \text{sign}(x)$	Not continuous	
$f(x) = \log(1 + \exp(x))$	C^∞	✓
$f(x) = 2x + 1$	C^∞	✓
$f(x) = \max(x, 0)^2$	C^1	✓

Gradient of a function



Gradient

The gradient $\boldsymbol{\nabla} F(\mathbf{x})$ of a function $F : \mathbb{R}^n \to \mathbb{R}$ at point $\mathbf{x}$ is the vector whose components are the partial derivatives of F

$$\nabla_{\mathbf{x}} F(\mathbf{x}) = \left[\frac{\partial F(\mathbf{x})}{\partial x_1}, \ldots, \frac{\partial F(\mathbf{x})}{\partial x_n} \right]^T \tag{11}$$

- If the gradient exists $\forall \mathbf{x}$ in the domain of F, the function F is called **differentiable**.
- $\nabla_{\mathbf{x}} F(\mathbf{x})$ give the steepest direction (where F is increasing the most).
- The vector normal to surface $(\mathbf{x}, F(\mathbf{x}))$ is given by $(\nabla_{\mathbf{x}} F(\mathbf{x}), -1)$.

Exercise 3: Gradient computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) =$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) =$$

Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T \mathbf{x} + b)$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) =$$

Exercise 3: Gradient computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) = \begin{bmatrix} 1 - x_2 \\ -1 - x1 \end{bmatrix}$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) =$$

Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T \mathbf{x} + b)$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) =$$

Exercise 3: Gradient computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) = \begin{bmatrix} 1 - x_2 \\ -1 - x1 \end{bmatrix}$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) = 2\mathbf{H}^T(\mathbf{H}\mathbf{x} - \mathbf{y})$$

Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T\mathbf{x} + b)$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) =$$

Exercise 3: Gradient computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) = \begin{bmatrix} 1 - x_2 \\ -1 - x1 \end{bmatrix}$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) = 2\mathbf{H}^T(\mathbf{H}\mathbf{x} - \mathbf{y})$$

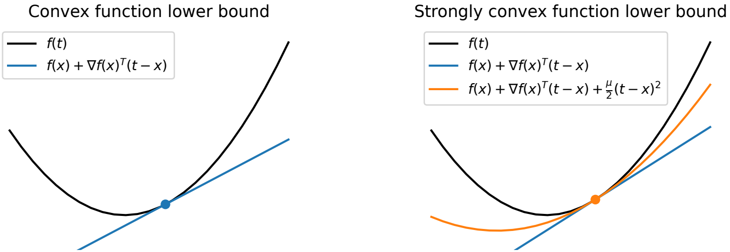
Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T\mathbf{x} + b)$$

Compute the gradient $\nabla_{\mathbf{x}} F(\mathbf{x})$:

$$\nabla_{\mathbf{x}} F(\mathbf{x}) = \mathbf{w} \exp(\mathbf{w}^T\mathbf{x} + b)$$

Smoothness and convexity



Convex function (first order definition)

F a differentiable function is **convex** if and only if

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \nabla F(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}), \quad \forall \mathbf{y}, \mathbf{x} \in \mathrm{dom} F \tag{12}$$

- A convex function is lower bounded by its local linear approximation.
- For $\mathcal{C} = \mathbb{R}^n$, if $\mathbf{x}$ if a global minimum if and only if $\nabla_{\mathbf{x}} F(\mathbf{x}) = \mathbf{0}$.

Strongly convex function

If F is a differentiable μ-strongly convex then

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \nabla F(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) + \frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2, \quad \forall \mathbf{y}, \mathbf{x} \in \mathrm{dom} F$$

Hessian and second derivatives

Hessian of a function

The Hessian matrix $\mathbf{H} = \nabla^2_{\mathbf{x}} F(\mathbf{x})$ of a twice differentiable function F is the matrix whose components can be expressed as

$$H_{i,j} = \left(\nabla^2_{\mathbf{x}} F(\mathbf{x})\right)_{i,j} = \frac{\partial^2 F(\mathbf{x})}{\partial x_i \partial x_j}$$

- **Convex function**: F is convex if and only if $\nabla^2_{\mathbf{x}} F(\mathbf{x})$ is semi definite positive $\forall \mathbf{x}$.
- If $\nabla^2_{\mathbf{x}} F(\mathbf{x})$ is strictly positive definite $\forall \mathbf{x}$ then F is strictly convex.
- if F is μ-strongly convex then $\nabla^2_{\mathbf{x}} F(\mathbf{x}) \succeq \mu \mathbf{I}$ $(\lambda_{\text{min}}(\nabla^2_{\mathbf{x}} F(\mathbf{x})) \geq \mu)$.

Second Order Taylor approximation

The function can be approximated around $\mathbf{x}_0$ with

$$F(\mathbf{x}) \approx F(\mathbf{x}_0) + \underbrace{\nabla_{\mathbf{x}_0} F(\mathbf{x}_0)^T (\mathbf{x} - \mathbf{x}_0)}_{\text{Linear term}} + \underbrace{(\mathbf{x} - \mathbf{x}_0)^T \mathbf{H} (\mathbf{x} - \mathbf{x}_0)}_{\text{Quadratic term}} \tag{13}$$

The approximation is exact if F is a polynomial of order ≤ 2

Exercise 4: Hessian computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) =$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) =$$

Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T \mathbf{x} + b)$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) =$$

Exercise 4: Hessian computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}, \quad \text{Not PSD}$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) =$$

Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T \mathbf{x} + b)$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) =$$

Exercise 4: Hessian computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}, \quad \text{Not PSD}$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) = 2\mathbf{H}^T\mathbf{H}, \quad \text{PSD}$$

Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T\mathbf{x} + b)$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) =$$

Exercise 4: Hessian computation

Two variables

$$F(\mathbf{x}) = x_1 - x_1 x_2 - x_2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}, \quad \text{Not PSD}$$

Quadratic loss

$$F(\mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) = 2\mathbf{H}^T\mathbf{H}, \quad \text{PSD}$$

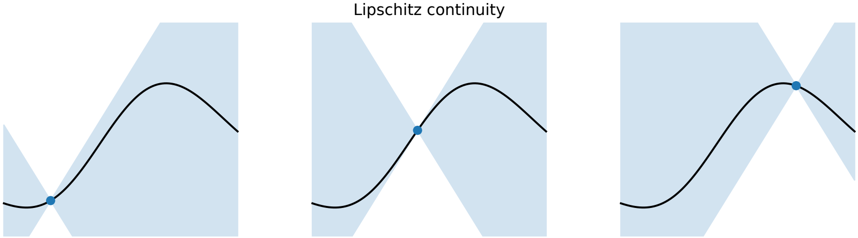
Exponential with linear function

$$F(\mathbf{x}) = \exp(\mathbf{w}^T\mathbf{x} + b)$$

Compute the Hessian $\nabla^2_{\mathbf{x}} F(\mathbf{x})$, is is positive semi definite ?

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) = \exp(\mathbf{w}^T\mathbf{x} + b)\mathbf{w}\mathbf{w}^T, \quad \text{PSD}$$

Lipschitz continuity



Lipschitz function

Function F is called **Lipschitz** or **Lipschitz continuous** if there exists a constant $L > 0$ such that $\forall \mathbf{x}, \mathbf{y} \in \mathcal{C}^2$

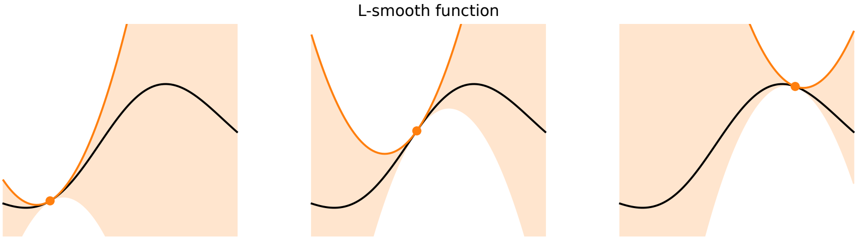
$$|F(\mathbf{x}) - F(\mathbf{y})| \leq L\|\mathbf{x} - \mathbf{y}\| \tag{14}$$

- A L satisfying the above constraint is called a Lipschitz constant of the function.
- If $L < 1$ the function is a contraction.
- Function F is **gradient Lipschitz**, also called L-**smooth**, if $\forall \mathbf{x}, \mathbf{y} \in \mathcal{C}^2$

$$\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\| \tag{15}$$

- Lipschitz functions can be easily upper bounded,

Properties Lipschitz functions



Upper bounds for Lipschitz functions

- If F is L-**smooth**, then the following inequality holds

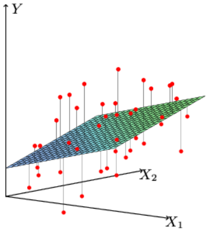
$$F(\mathbf{x}) \leq F(\mathbf{y}) + \nabla F(\mathbf{y})^\top (\mathbf{x} - \mathbf{y}) + \frac{L}{2} \|\mathbf{x} - \mathbf{y}\|^2 \tag{16}$$

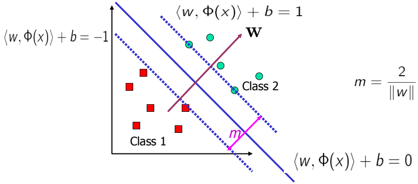
  the function can be upper-bounded by a quadratic function.

- If F is L-**smooth**, then the following inequality holds

$$\nabla^2_{\mathbf{x}} F(\mathbf{x}) \preceq L\mathbf{I} \quad (\lambda_{\mathsf{max}}(\nabla^2_{\mathbf{x}} F(\mathbf{x})) \leq L) \tag{17}$$

Convexity and smoothness in machine learning





Convex and smooth problems

- Smooth problem provides us with gradients for iterative methods.
- Convexity means the a solution of the problem is global.
- Convexity leads to several efficient algorithms.

ML approaches relying on convex problems

- Least square regression, Lasso.
- Support Vector Machines.
- Logistic and multinomial regression.

Local and global solutions

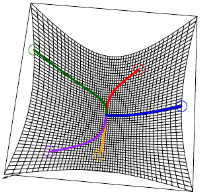
$$\min_{\mathbf{x}\in\mathcal{C}} \quad F(\mathbf{x})$$

Local solution

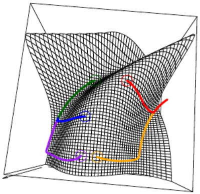
For the optimization problem above, a feasible point $\mathbf{x}^\star \in \mathcal{C}$ is a local optimum if there exists $R > 0$ such that

$$F(\mathbf{x}^\star) \leq F(\mathbf{x}) \quad \forall \mathbf{x} \in \{\mathbf{x} \in \mathcal{C}, \|\mathbf{x} - \mathbf{x}^\star\| \leq R\}$$

- If the problem is convex, all local optimum are global.
- For non-convex function, the optimum is global only if the equation is true for all $R > 0$.



Convex



Nonconvex

First order optimality condition

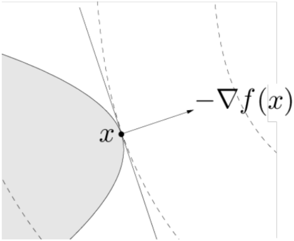
Convex and differentiable function

For the following convex problem

$$\min_{\mathbf{x} \in \mathcal{C}} \quad F(\mathbf{x})$$

the feasible point $\mathbf{x}^\star \in \mathcal{C}$ is globally optimal if and only if

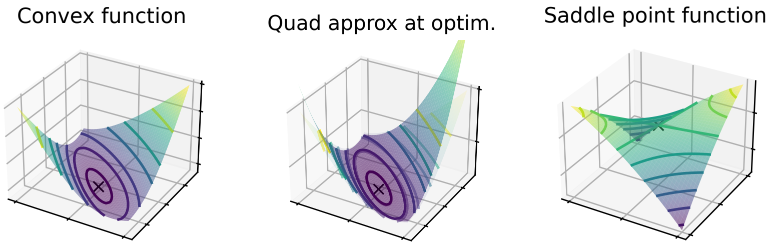
$$\nabla F(\mathbf{x}^\star)^\top (\mathbf{y} - \mathbf{x}^\star) \geq 0 \quad \forall \mathbf{y} \in \mathcal{C}$$



- Any feasible direction from $\mathbf{x}^\star$ is aligned with an increasing gradient.
- If $\mathcal{C} = \mathbb{R}^n$, the condition is equivalent to

$$\nabla F(\mathbf{x}^\star) = 0$$

Second order optimality conditions



Twice differentiable function

For the following problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x})$$

the feasible point $\mathbf{x}^\star \in \mathbb{R}^n$ is locally optimal if and only if

$$\nabla F(\mathbf{x}^\star) = 0 \quad \text{and} \quad \nabla^2 F(\mathbf{x}^\star) \succeq 0$$

- On general functions $\nabla F(\mathbf{x}^\star) = 0$ is not sufficient (saddle points).
- Equivalent to the first order condition on convex functions.

Section

Conclusion

Machine learning and optimization

- Learning is an optimization problem.
- Design a new machine learning method $\equiv$ design a new optimization problem.
- Convexity, smoothness lead to specific solver and guarantees.

Know your optimization problems

- If smooth and unconstrained $\rightarrow$ Gradient descent and variants.
- If non-smooth $\rightarrow$ proximal, projected, conditional gradients.
- If convex and/or constrained standard problems (LP,QP) $\rightarrow$ standard solvers.

Those are the next parts of the course.

Bibliography I

References books for the whole course.

Convex Optimization [Boyd and Vandenberghe, 2004]

- Available freely online: `https://web.stanford.edu/~boyd/cvxbook/`.
- Perfect introduction to convex optimization (the whole book).
- Convex sets (Ch. 2), Convex functions (Ch 3), Convex problems (Ch. 4).

Elements of statistical learning [Friedman et al., 2001]

- Freely available `https://web.stanford.edu/~hastie/Papers/ESLII.pdf`
- Perfect introduction to statistical learning and machine learning.
- Most of them are optimization problems!

Nonlinear Programming [Bertsekas, 1997]

- Reference optimization book, contains also most of the course.
- Unconstrained optimization (Ch. 1), duality and lagrangian (Ch. 3, 4 ,5).

Bibliography II

Other references

Convex analysis and monotone operator theory in Hilbert spaces [Bauschke et al., 2011]

- Awesome book with lot's of algorithms, and convergence proofs.
- All definitions (convexity, lower semi continuity) in specific chapters.
- All you need to know about proximal methods.

Numerical optimization [Nocedal and Wright, 2006]

- Classic introduction to numerical optimization.
- Very detailed unconstrained optimization, specific chapters for LP and QP.

Optimization for Machine Learning [Sra et al., 2012]

- Specific chapters for precise problems (non-convex, sparsity, interior points)
- For this course: Convex with sparsity (Ch. 2), Interior points (Ch. 3).

Linear Programming [Vanderbei et al., 2015]

- Reference book of LP (Simplex, interior point)

References I

[Bauschke et al., 2011] Bauschke, H. H., Combettes, P. L., et al. (2011).
Convex analysis and monotone operator theory in Hilbert spaces, volume 408.
Springer.

[Bertsekas, 1997] Bertsekas, D. P. (1997).
Nonlinear programming.
Journal of the Operational Research Society, 48(3):334–334.

[Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004).
Convex optimization.
Cambridge university press.

[Diamond and Boyd, 2016] Diamond, S. and Boyd, S. (2016).
Cvxpy: A python-embedded modeling language for convex optimization.
The Journal of Machine Learning Research, 17(1):2909–2913.

[Friedman et al., 2001] Friedman, J., Hastie, T., and Tibshirani, R. (2001).
The elements of statistical learning, volume 1.
Springer series in statistics New York.

References II

[Grant and Boyd, 2014] Grant, M. and Boyd, S. (2014).
CVX: Matlab software for disciplined convex programming, version 2.1.
`http://cvxr.com/cvx`.

[Grant et al., 2006] Grant, M., Boyd, S., and Ye, Y. (2006).
Disciplined convex programming.
Springer.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. (2006).
Numerical optimization.
Springer Science & Business Media.

[Sra et al., 2012] Sra, S., Nowozin, S., and Wright, S. J. (2012).
Optimization for machine learning.
Mit Press.

[Vanderbei et al., 2015] Vanderbei, R. J. et al. (2015).
Linear programming.
Springer.

[Vapnik, 2013] Vapnik, V. (2013).
The nature of statistical learning theory.
Springer science & business media.