EE4.66 Topics in Large Dimensional Data Processing

Wei Dai and Stefan Vlaski

Last Compiled: October 16, 2021

Contents

Contents ii

1 Syllabus 1
- 1.1 What is this module about . . . 1
- 1.2 The miscellaneous . . . 1
- 1.3 Assessment . . . 1
- 1.4 Syllabus . . . 2

2 Introduction 5
- 2.1 Key Challenges . . . 5
- 2.2 Data Points Are Isolated . . . 5
- 2.3 False Structure . . . 6
- 2.4 Overfitting . . . 7
  - 2.4.1 Univariate Polynomial Regression . . . 7
  - 2.4.2 Multivariate Polynomial Regression . . . 9

3 Least Squares 10
- 3.1 Problem Statement . . . 10
- 3.2 Matrix Decomposition . . . 12
- 3.3 Projecton . . . 15
- 3.4 The Underdetermined Case . . . 17
- 3.5 Numerical Stability . . . 18

4 Sparse Inverse Problems: Introduction 19
- 4.1 Problem Statement . . . 19
- 4.2 Exhaustive Search . . . 20
  - 4.2.1 The Algorithm . . . 20
  - 4.2.2 Performance Guarantees . . . 22
- 4.3 Applications . . . 23

5 Sparse Inverse Problems: L0 Methods 26
- 5.1 Some Easy Problems . . . 26
- 5.2 OMP . . . 27
  - 5.2.1 Algorithm . . . 27
  - 5.2.2 Performance Guarantees . . . 28
- 5.3 SP and Similar Algorithms . . . 30
  - 5.3.1 Algorithms . . . 30
  - 5.3.2 Performance Guarantees . . . 32

5.4 Majorization Minimization (MM) . . . 37
5.5 Appendix . . . 39

6 Convex Optimization: Introduction **41**
6.1 Convexity . . . 41
6.1.1 Definitions . . . 41
6.1.2 Conditions of Convexity . . . 42
6.2 Unconstrained Convex Optimization . . . 44
6.3 Line Search Methods . . . 45

7 Sparse Inverse Problems: ℓ_1-minimization **47**
7.1 Convex Relaxation . . . 47
7.2 LASSO Solvers . . . 50
7.2.1 Soft Thresholding Function . . . 50
7.2.2 Cyclic Coordinate Descent . . . 51
7.2.3 Iterative Shrinkage Thresholding Algorithm (ISTA) . . . 51
7.2.4 Majorization Minimization . . . 52
7.2.5 Least Angle Regression (LAR) . . . 52

8 Bilinear Inverse Problems **55**
8.1 Bilinear Inverse Problems . . . 55
8.1.1 Matrix Norms . . . 57
8.2 Examples . . . 58
8.2.1 Low-Rank Matrix Completion . . . 58
8.2.2 Blind Deconvolution . . . 60
8.2.3 Sparse PCA and Dictionary Learning . . . 61
8.3 Methods . . . 61
8.3.1 Alternating Optimization . . . 61
8.3.2 Greedy Algorithms . . . 62
8.3.3 Convex Relaxation . . . 62
8.4 Appendix . . . 63
8.4.1 Proof of Eckart-Young-Mirsky Theorem . . . 63

9 Constrained Convex Optimization **66**
9.1 General Form . . . 66
9.1.1 Sublevel Sets . . . 66
9.2 Duality . . . 67
9.3 KKT Conditions . . . 69
9.4 Examples . . . 70

10 ADMM **73**
10.1 Precursors . . . 73
10.1.1 Dual Ascent Method . . . 73

10.1.2 Method of Multipliers 74
10.2 ADMM 75
10.2.1 Optimality Conditions 77

11 Stochastic Optimization **78**
11.1 Motivation 78
11.2 Stochastic Least Squares 78
11.2.1 Gradient Descent for LS 78
11.2.2 Stochastic Gradient Descent for LS 79

Syllabus 1

1.1 What is this module about . 1
1.2 The miscellaneous 1
1.3 Assessment 1
1.4 Syllabus 2

1.1 What is this module about

It is *not* about

- Programming
- Computer architecture

It is about

- Concepts, Principles, and Technical Tools
  - Main Topics
    - Sparse linear inverse problems
    - Sparse bilinear inverse problems
  - Technical Tools
    - Linear algebra (matrix computations).
    - Convex and nonconvex optimisation
  - Selected applications
    - Google PageRank
    - Denoising.
    - Face recognition with block occlusion.
    - Recommendation engine: Netflix problem.
    - Dictionary learning
    - Blind deconvolution.
    - ...

1.2 The miscellaneous

GTA

Miss Farwa Abbas, Mr Zhengang Guo, and Mr Yifan Ran

1.3 Assessment

Coursework 100% [1]

- Project 1: ℓ_0 Methods for Sparse Recovery (16%)
- Project 2: ℓ_1 Methods for Sparse Recovery (20%)
- Project 3: Bilinear Inverse Problems (26%)
- Project 4: Read and Tell (30%)
- Peer Marking (8%)

1: More details on coursework will be announced.

1.4 Syllabus

The purpose of the module

The aim of the module is to introduce formulations and methods (convex and nonconvex optimization) arisen to process and analyse large dimensional data, and expose students to various application domains including image/video processing, machine learning, collaborative filtering, etc.

Prerequisites

Students are expected to have basic knowledge of linear algebra and algorithms.

Balance between theory and practice:

The lectures will cover ideas, formulations, and algorithms behind large dimensional data processing.

Some representative applications will be discussed in the lectures to show that difficult problems can solved by clever formulation and appropriately designed methods.

Coursework will help students

- use modern and popular software tools for big data processing and machine learning, including VSCode, Jupyter Notebook, Markdown, LaTex, and Julia;
- design and implement algorithmic solutions to test taught concepts and solve real-world problems;
- explore recent research results, expand knowledge, get inspired by modern applications and novel solutions, and develop future career.

Learning outcomes

1. Knowledge:
   Students will learn the theory underlying large dimensional data processing and analysis. We will
   a) present ideas, formulations, and algorithms for large dimensional data processing;
   b) demonstrate how basic tools in linear algebra and optimization can be applied to significantly improve the efficiency and accuracy of data processing;
   c) expose students to relevant real applications, including imaging processing, online recommendation, machine learning, etc.
2. Subject specific skills
   a) Intellectual skills:
   Students will learn essential paradigms and tools to handle large and high-dimensional data (e.g. in engineering, commerce, machine learning)

b) Practical Skills:
Students will learn to use modern tools popular for data science and machine learning, design and implement efficient algorithms for big data application, and get exposed to modern data processing methods and applications.

c) Transferable / Key Skills:
Students will learn to apply their knowledge to real-world situations. Through examples where real-world problems are addressed by different approaches, students will witness critical thinking and innovations.

Selected Applications

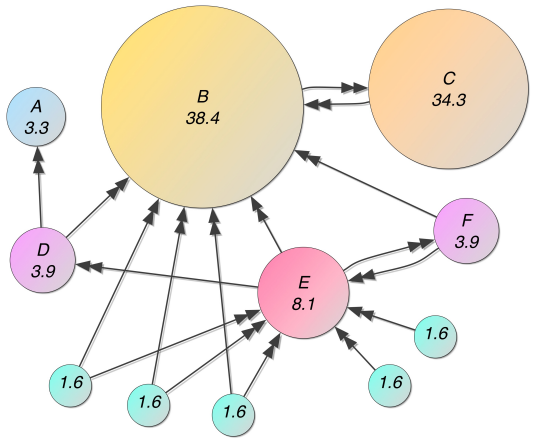


Figure 1.1: Google pagerank.
By 345Kai, Public Domain, https://en.wikipedia.org/w/index.php?curid=14424767

Figure 1.2: Image Denoising.

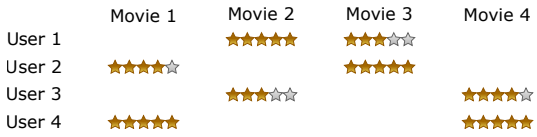


Figure 1.3: Netflix Problem.

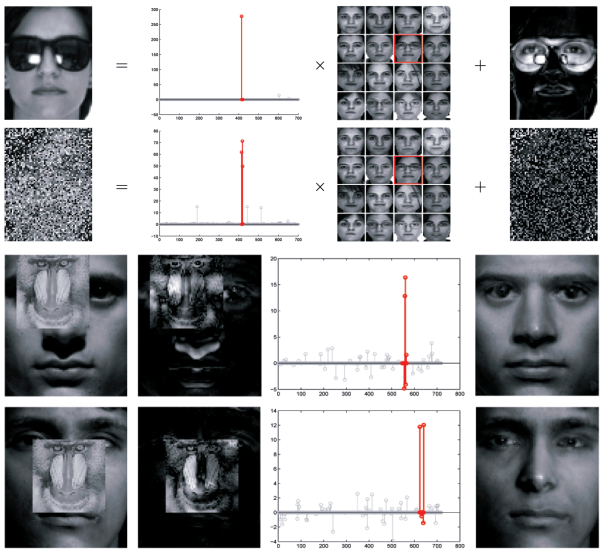


Figure 1.4: Face Recognition with Block Occlusion J. Wright et al., 2009.

Introduction 2

2.1 Key Challenges

2.1 Key Challenges 5
2.2 Data Points Are Isolated . 5
2.3 False Structure 6
2.4 Overfitting 7
2.4.1 Univariate Polynomial Regression 7
2.4.2 Multivariate Polynomial Regression 9

What is data analysis?

Data analysis is to learn an unknown function f from training data $\boldsymbol{x}$ such that

$$y_i \approx f(\boldsymbol{x}_i),$$

where $\boldsymbol{x}_i \in \mathbb{R}^n, \quad i = 1, 2, \cdots, m.$

Curse of Dimensionality

Use some image database as an example.[1]

In Classical data processing

- A small number n of parameters.
- A large number m of observations.

In many modern applications

- A large number n of parameters.
- A relatively small sample size m ($m \approx n$ or $m < n$).

Curse of dimensionality:

- The computational difficulty
- The intrinsic statistical difficulty
  - Data points are isolated.
  - False structures.
  - Overfitting (the inferred model describes the noise instead of the underlying relationship).

To address it: low dimensional structure.

1:

- THE MNIST database of handwritten digits.
  - $28 \times 28 = 784$ pixels.
  - 10 categories, 60,000 training samples, 10,000 test samples.
- Caltech 256: Pictures of objects belonging to 256 categories.
  - Pictures of various sizes: normally $100 \times 100 = 10,000$ pixels.
  - 256 categories, 30,607 images in total.
- Modern database
  - Including Imagenet, Labelme, etc.
  - Pictures of various sizes, including HD ones.
  - Less training images per category in general.

2.2 Data Points Are Isolated

Example 2.2.1 (K-nearest neighbors algorithm) Suppose we have training data $\{\boldsymbol{x}_1, y_1\}, \{\boldsymbol{x}_2, y_2\}, \cdots, \{\boldsymbol{x}_m, y_m\}$ taking values in $\mathbb{R}^n \times \{0, 1\}$, where y is the class label of $\boldsymbol{x}$ for classification, or the value of the underlying (unknown) function for regression.

For a given test data $\boldsymbol{x}_t$, reorder the training data such that $\|\boldsymbol{x}_{(1)} - \boldsymbol{x}_t\| \leq \cdots \leq \|\boldsymbol{x}_{(m)} - \boldsymbol{x}_t\|$.

- For classification: majority vote using K-nearest neighbors.
- For regression: average value of the K-nearest neighbors.

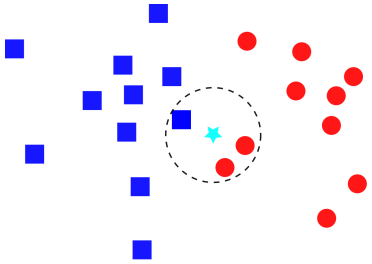


Figure 2.1: KNN Method

The performance of KNN is decided by how dense the training points are.

Question: In the n-dimensional space, how many training samples are needed so that for any given test sample, there exists at least one training sample that is *at most distance 1 away*?

Mathematical Question: how many unit balls are needed to cover the whole space the hypercube $[0,1]^n$?

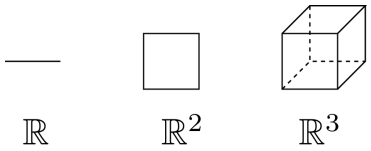


Figure 2.2: Unit tube in n-dimensional space

Cover the hypercube $[0,1]^n$ by unit balls:

- The volume of $V_n(r)$ of a n-dimensional ball of radius $r > 0$:

$$V_n(r) = \frac{\pi^{n/2}}{\Gamma(n/2+1)} r^n \approx \left(\frac{2\pi e r^2}{n}\right)^{n/2} (n\pi)^{-1/2}.$$

- To cover the hypercube $[0,1]^n$ by unit balls, one must have

$$[0,1]^n \subset \bigcup_{i=1}^{m} B_n\left(\boldsymbol{x}^{(i)}, 1\right).$$

- That is, $1 \leq mV_n(1)$, or

$$m \geq \frac{\Gamma(n/2+1)}{\pi^{n/2}} \approx \left(\frac{n}{2\pi e}\right)^{n/2} \sqrt{n\pi}.$$

Required number of data points for covering:

n	20	30	50	100	150
m	39	45630	5.7×10^{12}	42×10^{39}	1.28×10^{72}

2.3 False Structure

Task: Estimate empirical covariance matrix.

Given an n-dimensional random vector $\boldsymbol{X}$, the covariance matrix is defined as

$$\boldsymbol{\Sigma} := \mathrm{E}\left[(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}])(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}])^T\right].$$

Suppose that the random variable $\boldsymbol{X}$ is normally distributed. Given a sample consisting of m independent observations $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m$, the maximum likelihood estimator of the covariance matrix is given by

$$\begin{aligned}\hat{\boldsymbol{\Sigma}} &:= \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^{\mathsf{T}} \\ &= \frac{1}{m}\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^{\mathsf{T}},\end{aligned}$$

where $\bar{\boldsymbol{x}} = \frac{1}{m}\sum_i \boldsymbol{x}_i$ is the sample average and $\tilde{\boldsymbol{X}} = [\boldsymbol{x}_1 - \bar{\boldsymbol{x}}, \cdots, \boldsymbol{x}_m - \bar{\boldsymbol{x}}]$ is the sample covariance matrix.

Case 1: If n is fixed and $m \to \infty$, by the law of large numbers[2]

$$\hat{\boldsymbol{\Sigma}} \to \boldsymbol{I}.$$

2: Interested readers see https://en.wikipedia.org/wiki/Law_of_large_numbers for details.

Case 2: What happens if m and n are at the same scale?

Example 2.3.1 Let $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Compute the eigenvalues of the sample covariance matrix $\hat{\boldsymbol{\Sigma}}$ with $n = 200$ and $m = 10^5$ ($m \gg n$). The histogram of the eigenvalues is shown in Figure 2.3.

Let $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Compute the eigenvalues of $\hat{\boldsymbol{\Sigma}}$ with $n = 2000$ and $m = 10^4$ ($m \gtrsim n$). The histogram of the eigenvalues is shown in Figure 2.4.

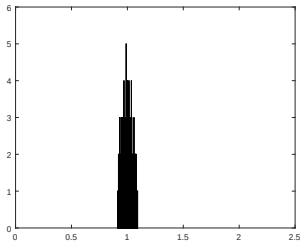


Figure 2.3: Eigenvalue histogram when number of samples is much more than the sample dimension.

Remark 2.3.1 The behaviour of eigenvalues of the empirical covariance can be predicted by random matrix theory.

If $n, m \to \infty$ proportionally with $n/m \to r \in \mathbb{R}^+$, then the empirical distribution of eigenvalues of empirical covariance matrix $\hat{\boldsymbol{\Sigma}}$ converges to Marchenko-Pastur distribution (see https://en.wikipedia.org/wiki/Marchenko%E2%80%93Pastur_distribution for more details). See Figure 2.5 for simulations.

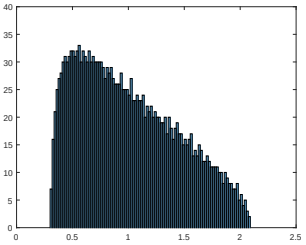


Figure 2.4: Eigenvalue histogram when number of samples is roughly equal to the sample dimension.

2.4 Overfitting

Definition 2.4.1 (Linear Regression) *Given training samples* $(\boldsymbol{x}_i, y_i)$, $i = 1, \cdots, m$, *want to estimate a linear function represented by* $\boldsymbol{\alpha} \in \mathbb{R}^n$ *s.t.* $y_i \approx \langle \boldsymbol{x}_i, \boldsymbol{\alpha} \rangle$.

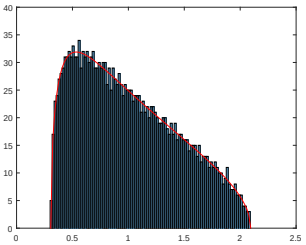


Figure 2.5: Empirical distribution of eigenvalues converges to Marchenko-Pastur distribution.

Solution: Let $\boldsymbol{y} = \left[y_1, \cdots, y_m\right]^T, \boldsymbol{X} = \left[\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m\right]^T$, and $\boldsymbol{e} = [e_1, \cdots, e_m]$. Write

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\alpha} + \boldsymbol{e},$$

Then solve for $\boldsymbol{\alpha}$.[3]

Issue: when $m < n$, there are infinite many valid solutions to $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\alpha}$.

3: See https://en.wikipedia.org/wiki/Linear_regression for a more detailed description.

2.4.1 Univariate Polynomial Regression

Task: Suppose that the input data $x \in \mathbb{R}$ (the univariate case). Assume that the unknown function f can be approximated by a degree S univariate polynomial:

$$f(x) = \sum_{s=0}^{S} \alpha_s x^s.$$

Based on the input-output pairs (x_i, y_i), $i = 1, 2, \cdots, m$, one would find the unknown polynomial so that $y_i \approx f(x_i)$.

Fact
Given m distinct samples, there exists a polynomial of degree $m - 1$ to match the data perfectly.

Proof. Since

$$\sum_{\ell=0}^{m-1} \alpha_\ell x_i^\ell = y_i,\ i = 1, 2, \cdots, m,$$

It holds that

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}_{\boldsymbol{y}} = \underbrace{\begin{bmatrix} 1 & x_1 & \cdots & x_1^{m-1} \\ 1 & x_2 & \cdots & x_2^{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & \cdots & x_m^{m-1} \end{bmatrix}}_{\boldsymbol{X}} \underbrace{\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{m-1} \end{bmatrix}}_{\boldsymbol{\alpha}}.$$

The matrix $\boldsymbol{X}$ is a Vandermonde matrix and has full rank. [4] Hence, the unknown vector $\boldsymbol{\alpha}$ can be computed from $\boldsymbol{\alpha} = \boldsymbol{X}^{-1}\boldsymbol{y}$. □

4: See https://en.wikipedia.org/wiki/Vandermonde_matrix for the definition and property of Vandermonde matrix.

Issue: The polynomial obtained using the above method can be an overfitting of noise/numerical errors.

Example 2.4.1 Fit a univariate polynomial to the data points shown in Figure 2.6.

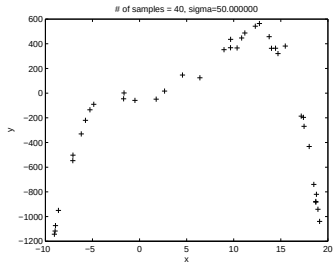


Figure 2.6: The data to be fit with.

Now we use the above method to fit a univariate polynomial to the data. It is clear that $f(x_i) = y_i$ from the results shown in Figure 2.7.

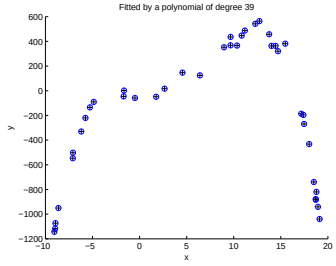


Figure 2.7: Fitted polynomial at data points.

However, the fitted polynomial is actually an overfitting. See Figure 2.8 for details.

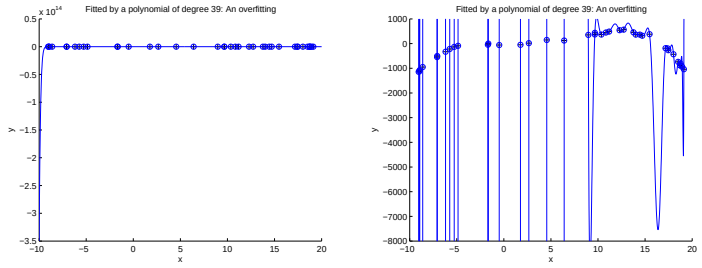


Figure 2.8: Fitted polynomial is an overfitting.

The reason of overfitting,[5] in this case, is the poor condition number of the matrix $\boldsymbol{X}$.[6]

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\alpha}_0 + \boldsymbol{w} \quad \Rightarrow \quad \hat{\boldsymbol{\alpha}} = \boldsymbol{X}^{-1}\boldsymbol{y} = \boldsymbol{\alpha}_0 + \boldsymbol{X}^{-1}\boldsymbol{w}.$$

Solution: A sparse solution of $\boldsymbol{\alpha}$.
$\boldsymbol{\alpha}$ is sparse (only a few nonzeros): force $\alpha_5 = \alpha_6 = \cdots = \alpha_{39} = 0$.

As presented in Figure 2.9, the sparse solution provides a much better fitting.

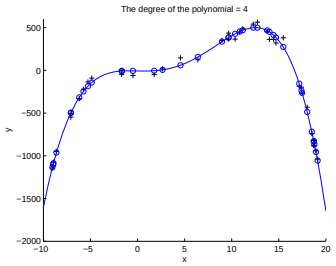


Figure 2.9: A sparse solution gives a much better fitting.

2.4.2 Multivariate Polynomial Regression

Task: Suppose that the input data $x \in \mathbb{R}^d$ (the multivariate case). Assume that the unknown function f can be approximated by a multivariate polynomial.

Here, we are particularly interested in a degree 2 multivariate polynomial

$$\begin{aligned} f(\boldsymbol{x}) =& \alpha_0 + \alpha_1 x_1 + \cdots + \alpha_d x_d \\ &+ \alpha_{d+1} x_1^2 + \alpha_{d+2} x_1 x_2 + \cdots + \alpha_{2d} x_1 x_d \\ &+ \alpha_{2d+1} x_2^2 + \cdots + \alpha_{n-2} x_{d-1} x_d \\ &+ \alpha_{n-1} x_d^2. \end{aligned}$$

The regression task:
Given $\left(\boldsymbol{x}(j), y(j)\right), j = 1, 2, \cdots, m$, try to find $\boldsymbol{\alpha} = [\alpha_0, \alpha_1, \cdots, \alpha_{n-1}]$.

$$\underbrace{\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(m) \end{bmatrix}}_{\boldsymbol{y}} = \underbrace{\begin{bmatrix} 1 & x_1(1) & \cdots & x_d^2(1) \\ 1 & x_1(2) & \cdots & x_d^2(2) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1(m) & \cdots & x_d^2(m) \end{bmatrix}}_{\boldsymbol{X}} \underbrace{\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{n-1} \end{bmatrix}}_{\boldsymbol{\alpha}}.$$

Two issues:

- No sufficient data:
  - Number of terms $n = O\left(d^2\right) \gg m$.
- Even when data is abundant: need to avoid overfitting.

Sparsity plays a key role: Enforce most of the elements of $\boldsymbol{\alpha}$ to be zero.

5: See https://en.wikipedia.org/wiki/Overfitting for overfitting.

6: See https://en.wikipedia.org/wiki/Condition_number. There is also an exercise in the first coursework.

3 Least Squares

3.1 Problem Statement 10
3.2 Matrix Decomposition . . . 12
3.3 Projecton 15
3.4 The Underdetermined Case 17
3.5 Numerical Stability 18

3.1 Problem Statement

Consider the model

$$y = Ax + w,$$

where y is the measurement vector, x is the unknown signal/parameter vector, and w is the noise vector.

The least squares estimation of x is given by

$$\hat{x} = \min_x \underbrace{\frac{1}{2}\|y - Ax\|^2}_{f(x)}.$$

It can be verified that

$$\nabla f(x) = A^T(Ax - y).$$

The optimal solution (in least squares sense) is obtained by setting $\nabla f(x) = \mathbf{0}$. One has

$$(A^T A)\hat{x} = A^T y.$$

We have two cases.

1. $A^\top A$ is of full rank. In this case,

$$\hat{x} = (A^\top A)^{-1} A^\top y.$$

2. $A^\top A$ is rank deficient. In this case, the global minimizer $x^\star$ is not unique. We will study one particular global minimizer defined as follows. [1] Consider the compact singular value decomposition (SVD) $A = U\Sigma V^\top$. Set

$$\hat{x} = V\Sigma^{-1}U^\top y.$$

1: The details will be covered in the rest of this chapter.

For both cases, we can write

$$\hat{x} = A^\dagger y = V\Sigma^{-1}U^\top y,$$

where $A^\dagger$ is the Moore-Penrose pseudoinverse.

This chapter discusses

- Matrix decomposition: How to compute $A^\dagger$ especially when $A^\top A$ is not of full rank.
  - Application: Google PageRank
- Projection: $A\hat{x} = \text{proj}(y, \text{span}(A))$.
- The underdetermined case: $\hat{x} = A^\dagger y$ has the least length among all global minimizers.
- Numerical stability

Pseudoinverse

Linear Independence and Dependence: Vectors $v_1, \cdots, v_n \in \mathbb{R}^m$ are linearly independent if

$$\sum \lambda_i v_i = \mathbf{0} \;\Rightarrow\; \lambda_i = 0, \; \forall i.$$

In matrix format,

$$[v_1, \cdots, v_n]\boldsymbol{\lambda} = \mathbf{0} \in \mathbb{R}^m \;\Rightarrow\; \boldsymbol{\lambda} = \mathbf{0} \in \mathbb{R}^n.$$

Vectors $v_1, \cdots, v_n \in \mathbb{R}^m$ are linearly dependent if $\exists \boldsymbol{\lambda} \neq \mathbf{0}$ such that

$$[v_1, \cdots, v_n]\boldsymbol{\lambda} = \mathbf{0}.$$

Matrix Rank: Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ be a given matrix.
Column rank: the maximum number of linearly independent columns.
Row rank: the maximum number of linearly independent rows.
Rank: For every matrix, column rank = row rank = rank. [2]

2: In Julia, use function `rank` (requiring LinearAlgebra package) to calculate the rank of a matrix.

Definition 3.1.1 (Matrix Inverse and Pseudoinverse)

- *A matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is invertible if there exists a matrix $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ such that*
$$\boldsymbol{AB} = \boldsymbol{BA} = \boldsymbol{I}.$$
The matrix $\boldsymbol{B}$ is the inverse of $\boldsymbol{A}$.
- *For a matrix $\boldsymbol{A}$, its Moore-Penrose pseudoinverse $\boldsymbol{A}^\dagger$ is defined to satisfy*
  - $\boldsymbol{A}\boldsymbol{A}^\dagger\boldsymbol{A} = \boldsymbol{A}$, *and* $\boldsymbol{A}^\dagger\boldsymbol{A}\boldsymbol{A}^\dagger = \boldsymbol{A}^\dagger$.
  - $\left(\boldsymbol{A}\boldsymbol{A}^\dagger\right)^T = \boldsymbol{A}\boldsymbol{A}^\dagger$, *and* $\left(\boldsymbol{A}^\dagger\boldsymbol{A}\right)^T = \boldsymbol{A}^\dagger\boldsymbol{A}$.

[3]

3: In Julia, use functions `inv` and `LinearAlgebra > pinv` for inverse and pseduoinverse respectively.

- $\boldsymbol{A}$ is invertible $\Leftrightarrow$ $\boldsymbol{A}$ is of full rank.
- Moore-Penrose pseudoinverse is well defined no matter whether $\boldsymbol{A}$ has full rank or not.

Example 3.1.1 It is easy to compute the inverse and Moore-Penrose pseudoinverse of diagonal matrices.

- $$\boldsymbol{A} = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \Rightarrow \boldsymbol{A}^{-1} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1 \end{bmatrix} = \boldsymbol{A}^\dagger.$$
- $$\boldsymbol{A} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \Rightarrow \boldsymbol{A}^\dagger = \begin{bmatrix} 1/2 & 0 \\ 0 & 0 \end{bmatrix}.$$
  Its inverse is not well defined.

Matrix decomposition helps compute the inverse and Moore-Penrose pseudoinverse of a non-diagonal matrix.

3.2 Matrix Decomposition

Eigenvalue decomposition

Definition 3.2.1 (Eigendecomposition, spectral decomposition) *A non-zero vector $\boldsymbol{v} \in \mathbb{R}^n$ is an eigenvector of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ if there is a constant λ such that*

$$\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{v},$$

where λ is called the eigenvalue corresponding to $\boldsymbol{v}$.

4

4: In Julia, use function `LinearAlgebra > eigen` for engenvalue decomposition.

Example 3.2.1

This example is linked to Google PageRank discussed in Section **??**.

1. Let
$$\boldsymbol{A} = \begin{bmatrix} 5 & 3 \\ 2 & 6 \end{bmatrix}.$$
Show that $\boldsymbol{A}$ has an eigenvalue 8 and the corresponding eigenvector is $\boldsymbol{v} = [1, 1]^\mathsf{T}$.
Use `Julia > LinearAlgebra > eigen` to numerically compute the eigenvalues and eigenvectors of $\boldsymbol{A}$. Verify the above claim.
Note that the two eigenvectors are linearly independent but not orthogonal.
2. Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ be a square matrix such that the sum of every row is the same, i.e.,
$$c = \sum_j A_{i,j} \; \forall i.$$
Show that $\boldsymbol{A}$ has an eigenvalue c and the corresponding eigenvector is $\boldsymbol{v} = [1, 1, \cdots, 1]^\mathsf{T}$.
3. Suppose that the eigenvalues and eigenvectors of $\boldsymbol{A}$ are given by $\lambda_1, \cdots, \lambda_n$, and $\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n$, respectively.
Let $\boldsymbol{\Lambda} = \text{diag}([\lambda_1, \cdots, \lambda_n])$ and $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n]$. It can be verified that
$$\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{\Lambda} \Rightarrow \boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^{-1}.$$
Now let $\boldsymbol{B} = \boldsymbol{A}^\mathsf{T}$. Denote the eigenvalues and eigenvectors of $\boldsymbol{B}$ by $\lambda_{B,1}, \cdots, \lambda_{B,n}$, and $\boldsymbol{q}_{B,1}, \cdots, \boldsymbol{q}_{B,n}$, respectively. Let $\boldsymbol{\Lambda}_B = \text{diag}([\lambda_{B,1}, \cdots, \lambda_{B,n}])$ and $\boldsymbol{Q}_B = [\boldsymbol{q}_{B,1}, \cdots, \boldsymbol{q}_{B,n}]$.
What can you say about the relationship between $\boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}_B$, and that between $\boldsymbol{Q}$ and $\boldsymbol{Q}_B$?
4. With the above, you can prove that the adjacent matrix in the Google PageRank example (Example 3.2.4) has an eigenvalue 1.

Singular Value Decomposition

Definition 3.2.2 (Singular Value Decomposition) *For an arbitrary matrix $A \in \mathbb{R}^{m\times n}$, there exists a factorization of the form*

$$A = U\Sigma V^T = \sum_i \sigma_i u_i v_i^\mathsf{T},$$

where $U \in \mathbb{R}^{m\times m}$ is unitary (contains m orthonormal columns), $\Sigma \in \mathbb{R}^{m\times n}$ is diagonal matrix with non-negative diagonal entries, and $V \in \mathbb{R}^{n\times n}$ is unitary.

The m columns of U and the n columns of V are called the left-singular vectors and right-singular vectors of M, respectively. The diagonal entries σ_i of Σ are known as the singular values of M.

5

5: In Julia, use function `LinearAlgebra > svd` for singular value decomposition.

Remark 3.2.1

1. Let $\sigma_i \geq 0$.
   Note that if $A = \sum_i \sigma_i u_i v_i^\mathsf{T}$ is a valid SVD, then so is $A = \sum_i -\sigma_i(-u_i)v_i^\mathsf{T}$. WLOG, it is a convention to assume $\sigma_i \geq 0$.
2. List the singular values in descending order, i.e., $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(m,n)} \geq 0$. In this case, the diagonal matrix Σ is uniquely determined.
3. The matrix U and V are not unique.
   This is clear by observing that $A = \sum_i \sigma_i u_i v_i^\mathsf{T} = \sum_i \sigma_i(-u_i)(-v_i)^\mathsf{T}$.

Remark 3.2.2 (Compact SVD) Suppose that the matrix $A \in \mathbb{R}^{m\times n}$ has the rank $r < \min(m, n)$. Its compact SVD is given by

$$A = U_r\Sigma_r V_r^\mathsf{T},$$

where $U_r = U_{:,1:r} \in \mathbb{R}^{m\times r}$, $\Sigma_r = \Sigma_{1:r,1:r} \in \mathbb{R}^{r\times r}$, and $V_r = V_{:,1:r} \in \mathbb{R}^{n\times r}$.

Example 3.2.2 (Full and Compact SVDs)

$$\begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}$$

$$= \underbrace{\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 2 & \\ & 0 \end{bmatrix}}_{\Sigma} \underbrace{\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}^\mathsf{T}}_{V^\mathsf{T}} \qquad \text{(Full SVD)}$$

$$= \underbrace{\frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{U} \underbrace{[2]}_{\Sigma} \underbrace{\frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix}^\mathsf{T}}_{V^\mathsf{T}} \qquad \text{(Compact SVD)}$$

Example 3.2.3 Find full SVD of the matrices

$$A = \frac{1}{2}\begin{bmatrix} 1 & 1 & -2 \\ 1 & 1 & 2 \\ 1 & -1 & 2 \\ 1 & -1 & -2 \end{bmatrix}$$

and

$$\boldsymbol{B} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ 2 & -2 & -2 & 2 \end{bmatrix}.$$

6

6: A computing software is not needed. Mind the pattern of columns of $\boldsymbol{A}$ and $\boldsymbol{B}$.

EVD and SVD

Let

$$\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T.$$

Then

$$\boldsymbol{M}\boldsymbol{M}^T = \boldsymbol{U}\boldsymbol{\Sigma}^2\boldsymbol{U}^T \text{ and } \boldsymbol{M}^T\boldsymbol{M} = \boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^T.$$

- The left-singular vectors $\boldsymbol{u}_i$'s of $\boldsymbol{M}$ are eigenvectors of $\boldsymbol{M}\boldsymbol{M}^T$.
- The right-singular vectors $\boldsymbol{v}_i$'s of $\boldsymbol{M}$ are eigenvectors of $\boldsymbol{M}^T\boldsymbol{M}$.
- The singular values σ_i's of $\boldsymbol{M}$ are the square roots of the eigenvalues of both $\boldsymbol{M}\boldsymbol{M}^T$ and $\boldsymbol{M}^T\boldsymbol{M}$. That is, $\lambda_i = \sigma_i^2$.

Decomposition and Inverse

Eigenvalue decomposition and inverse: If matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ can be eigendecomposed and if none of its eigenvalues are zero, then A is invertible (nonsingular) and its inverse is given by

$$\boldsymbol{A}^{-1} = \boldsymbol{Q}\boldsymbol{\Lambda}^{-1}\boldsymbol{Q}^{-1},$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with

$$\boldsymbol{\Lambda}_{i,i} = \lambda_i.$$

Singular value decomposition and pseudoinverse: Consider the **compact SVD** of a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ given by

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T.$$

Then it holds that

$$\boldsymbol{A}^\dagger = \boldsymbol{V}\boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T. \tag{3.1}$$

It is straightforward to verify the following [7]

7: A nice exercise for you.

- An equivalent definition is

$$\boldsymbol{A}^\dagger = (\boldsymbol{A}^T\boldsymbol{A})^\dagger\boldsymbol{A}^T.$$

- The definition in (3.1) is equivalent to Definition 3.1.1.
  - $$\boldsymbol{A}\boldsymbol{A}^\dagger\boldsymbol{A} = \boldsymbol{A},\ \boldsymbol{A}^\dagger\boldsymbol{A}\boldsymbol{A}^\dagger = \boldsymbol{A}^\dagger.$$
  - $$\boldsymbol{A}\boldsymbol{A}^\dagger = \boldsymbol{U}\boldsymbol{U}^T,\ \boldsymbol{A}^\dagger\boldsymbol{A} = \boldsymbol{V}\boldsymbol{V}^T. \tag{3.2}$$

Application

Google PageRank: Assign importance scores (ranks) to webpages.

Mathematical Model: Denote the rank of a webpage i by x_i. The goal is to find x_i such that

$$x_i = \sum_{j \in \mathcal{C}_i} \frac{1}{N_j} x_j,$$

where

$$\mathcal{C}_i = \{j : \text{There is a link from } j \text{ to } i\}$$
$$N_i = \text{Number of links from } j$$

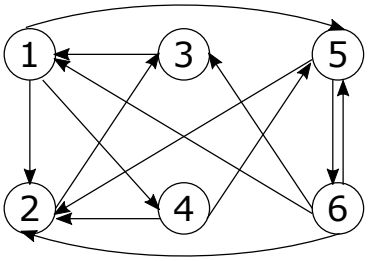


Figure 3.1: A simple directed graph as an example

Example 3.2.4 The graph in Figure 3.1 can be represented by the adjacent matrix

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & \frac{1}{4} \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{4} \\ 0 & 1 & 0 & 0 & 0 & \frac{1}{4} \\ \frac{1}{3} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \end{bmatrix}.$$

It can be proved that $\boldsymbol{A}$ has an eigenvalue 1 and the corresponding eigenvector $\boldsymbol{x} = [x_1, x_2, \cdots, x_n]^\mathsf{T}$ satisfies that

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{x} \Rightarrow x_i = \sum_{j \in \mathcal{C}_i} \frac{1}{N_j} x_j.$$

Compute the eigenvalue decomposition of $\boldsymbol{A}$ using a scientific computation software. One gets

$$\boldsymbol{x} = [0.57, 0.49, 0.53, 0.19, 0.32, 0.16]^\mathsf{T}.$$

3.3 Projecton

Definition 3.3.1 (Linear subspace) *Let $\mathcal{B} = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\} \subset \mathbb{R}^m$ containing linearly independent vectors.*

The linear span of $\mathcal{B}$ is defined as

$$span\,(\mathcal{B}) = \left\{ \sum_{i=1}^{n} \lambda_i \boldsymbol{v}_i : \ \lambda_i \in \mathbb{R} \right\}.$$

It is a linear subspace of $\mathbb{R}^m$.

- The set $\mathcal{B}$ is a basis for the linear subspace $\mathcal{S} = \langle \mathcal{B} \rangle$.
  - The basis $\mathcal{B}$ may not be unique, but its dimension is.
  - $\dim(\mathcal{S}) = n$: the # of vectors in a basis.

- $\mathcal{B}$ is orthonormal if $\langle v_i, v_j \rangle = 0$ for $i \neq j$ and $\langle v_i, v_i \rangle = 1$.
- For convenience, we use span $(\mathcal{B})$ and span $(\boldsymbol{B})$ interchangeably where $\boldsymbol{B} = [v_1, \cdots, v_n]$.

Definition 3.3.2 (Projection) *The projection of $\boldsymbol{v} \in \mathbb{R}^m$ onto the subspace span $(\boldsymbol{A})$ is defined as*

$$v_p = proj\,(\boldsymbol{v}, \boldsymbol{A}) = \arg \min_{\boldsymbol{u} \in span(\boldsymbol{A})} \|\boldsymbol{v} - \boldsymbol{u}\|_2.$$

And the projection residue is given by

$$v_r = resid\,(\boldsymbol{v}, \boldsymbol{A}) = \boldsymbol{v} - v_p.$$

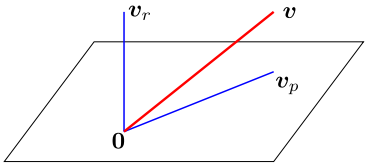


Figure 3.2: Projection

Example 3.3.1 Let

$$\boldsymbol{v} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \ \boldsymbol{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \ \boldsymbol{B} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & -\frac{2}{\sqrt{6}} \end{bmatrix}.$$

Find proj$(\boldsymbol{v}, \boldsymbol{A})$ and proj$(\boldsymbol{v}, \boldsymbol{B})$.

Note that the above example is easy to solve because both matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are orthonormal. The following proposition provides a way to calculate projection for general tall matrix $\boldsymbol{A}$.

Proposition 3.3.1 *Consider the compact SVD $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$. Then*

1. $v_p = \boldsymbol{A}\boldsymbol{A}^\dagger \boldsymbol{v} = \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{v}$.
2. $v_r \perp span(\boldsymbol{A})$.

Proof. 1. Any $\boldsymbol{u} \in \text{span}(\boldsymbol{A})$ can be written as $\boldsymbol{u} = \boldsymbol{A}\boldsymbol{x}$ for some $\boldsymbol{x}$. Hence

$$\begin{aligned} v_p &= \arg \min_{\boldsymbol{u} \in \text{span}(\boldsymbol{A})} \|\boldsymbol{v} - \boldsymbol{u}\|_2^2 \\ &= \boldsymbol{A} \arg \min_{\boldsymbol{x}} \|\boldsymbol{v} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \end{aligned}$$

By setting $\nabla \|\boldsymbol{v} - \boldsymbol{A}\boldsymbol{x}\|_2^2 = \boldsymbol{0}$, it is clear that

$$\hat{\boldsymbol{x}} = \boldsymbol{A}^\dagger \boldsymbol{v} \text{ and } v_p = \boldsymbol{A}\boldsymbol{A}^\dagger \boldsymbol{v}.$$

That $v_p = \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{v}$ follows from (3.2).

2.

$$\begin{aligned} \boldsymbol{A}^T v_r &= \boldsymbol{A}^T \left(\boldsymbol{v} - \boldsymbol{A}\boldsymbol{A}^\dagger \boldsymbol{v} \right) \\ &= \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^T \left(\boldsymbol{v} - \boldsymbol{U}\boldsymbol{U}^T \boldsymbol{v} \right) \\ &= \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^T \boldsymbol{v} - \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^T \boldsymbol{v} \\ &= \boldsymbol{0} \end{aligned}$$

□

Remark 3.3.1 Now it is clear that the least squares solution is nothing but a projection.

3.4 The Underdetermined Case

Consider the problem $\min_x \|y - Ax\|^2$ where $A^\top A$ does not have full rank.

Let $r < n$ be the rank of A. Consider the compact SVD $A = U\Sigma V^\top$, where $U \in \mathbb{R}^{m\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$, and $V \in \mathbb{R}^{n\times r}$. We have the following observations.

$\hat{x}$ is a global minimizer.

This can be verified by the facts that $f(x) = \|y - Ax\|^2$ is convex in x and that

$$\begin{aligned}\nabla f(\hat{x}) &= A^\top(A\hat{x} - y) = -A^\top \text{resid}(y, A)\\ &= 0.\end{aligned}$$

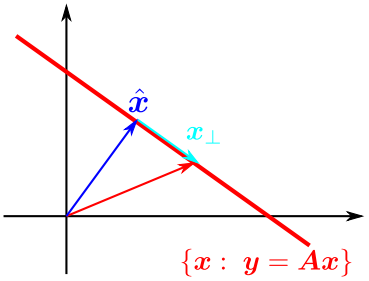


Figure 3.3: Solution set $\mathcal{X}$

The global minimizer is not unique.

Define the set of global minimizer as

$$\begin{aligned}\mathcal{X} &:= \left\{x : \|y - Ax\|^2 = \min_z \|y - Az\|^2\right\}\\ &= \{x : x = \hat{x} + V_\perp c_\perp,\ c_\perp \in \mathbb{R}^{n-m}\},\end{aligned}$$

where $V_\perp \in \mathbb{R}^{n\times(n-r)}$ is an orthogonal complement to V, i.e., $[V, V_\perp] \in \mathbb{R}^{n\times n}$ [8] is orthonormal.

8: $[V, V_\perp]$ can be calculated from a full SVD of A.

A quick check: For all $x = \hat{x} + V_\perp c_\perp$, it holds that

$$\begin{aligned}\|y - Ax\|^2 &= \left\|y - A\left(\hat{x} + V_\perp c_\perp\right)\right\|^2\\ &= \left\|y - A\hat{x} + 0\right\|^2\\ &= \left\|y - A\hat{x}\right\|^2.\end{aligned}$$

The least squares solution $\hat{x} = A^\dagger y$ has the least length among all global minimizers.

The least squares solution $\hat{x} = A^\dagger y$ satisfies

$$\hat{x} = \arg\min \left\{\|x\| : \|y - Ax\|^2 = \min_z \|y - Az\|^2\right\}.$$

Proof. SVD provides a way to decompose any $x \in \mathbb{R}^n$ into two parts:

$$\begin{aligned}x = Ix &= [V, V_\perp]\begin{bmatrix} V^T \\ V_\perp^T \end{bmatrix} x\\ &= VV^T x + V_\perp V_\perp^T x.\end{aligned}$$

Consider an arbitrary $\boldsymbol{x} \in \mathcal{X}$, it holds that

$$\begin{aligned}\|\boldsymbol{x}\|^2 &= \|\hat{\boldsymbol{x}} + \boldsymbol{V}_\perp \boldsymbol{c}_\perp\|^2 = \|\boldsymbol{V}\boldsymbol{c} + \boldsymbol{V}_\perp \boldsymbol{c}_\perp\|^2 \\ &= \|\boldsymbol{V}\boldsymbol{c}\|^2 + \langle \boldsymbol{V}\boldsymbol{c}, \boldsymbol{V}_\perp \boldsymbol{c}_\perp \rangle + \|\boldsymbol{V}_\perp \boldsymbol{c}_\perp\|^2 \\ &= \|\boldsymbol{V}\boldsymbol{c}\|^2 + \|\boldsymbol{V}_\perp \boldsymbol{c}_\perp\|^2 \\ &\geq \|\boldsymbol{V}\boldsymbol{c}\|^2 = \|\hat{\boldsymbol{x}}\|^2.\end{aligned}$$

□

3.5 Numerical Stability

Consider the model

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{w},$$

where $\boldsymbol{x}_0$ is the ground truth signal and $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$.

Assume that $\boldsymbol{A}$ is tall and has full rank. Consider the compact SVD $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$.[9]

9: What are the dimensions of $\boldsymbol{U}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{V}$?

The least squares solution is given by

$$\begin{aligned}\hat{\boldsymbol{x}} &= \boldsymbol{A}^\dagger \boldsymbol{y} = \boldsymbol{A}^\dagger (\boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{w}) \\ &= (\boldsymbol{A}^\top \boldsymbol{A})^{-1} \boldsymbol{A}^\top (\boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{w}) \\ &= \boldsymbol{x}_0 + \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top \boldsymbol{w}.\end{aligned}$$

Hence,

$$\|\hat{\boldsymbol{x}} - \boldsymbol{x}_0\|_2 = \|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^T \boldsymbol{w}\|.$$

The estimation error $\|\hat{\boldsymbol{x}} - \boldsymbol{x}_0\|$ can be then bounded by using the following steps: [10]

10: This is actually a coursework problem for you.

1. Show that $\|\boldsymbol{V}\boldsymbol{z}\|^2 = \|\boldsymbol{z}\|^2, \forall \boldsymbol{z} \in \mathbb{R}^n$.
2. Show that $\forall \boldsymbol{z} \in \mathbb{R}^n$,

$$\sigma_{\max}^{-1} \|\boldsymbol{z}\| \leq \|\boldsymbol{\Sigma}^{-1}\boldsymbol{z}\| \leq \sigma_{\min}^{-1} \|\boldsymbol{z}\|.$$

3. Show that[11]

$$0 \leq \|\boldsymbol{U}^\top \boldsymbol{w}\| \leq \|\boldsymbol{w}\|,$$

and

$$\mathrm{E}[\|\boldsymbol{U}^\top \boldsymbol{w}\|^2] = n\sigma^2.$$

11: What is the dimension of $\boldsymbol{U}^\top \boldsymbol{w}$? Its distribution?

4. Combine the results above, you should be able to derive bounds on $\|\hat{\boldsymbol{x}} - \boldsymbol{x}_0\|$ (in terms of $\|\boldsymbol{w}\|$) and $\mathrm{E}[\|\hat{\boldsymbol{x}} - \boldsymbol{x}_0\|^2]$ (in terms of σ^2).

In numerical analysis, the condition number of the matrix $\boldsymbol{A}$, [12] defined as

$$\kappa(\boldsymbol{A}) := \frac{\sigma_{\max}(\boldsymbol{A})}{\sigma_{\min}(\boldsymbol{A})}$$

is an important criterion to measure the stability of the least squares solution.

12: See https://en.wikipedia.org/wiki/Condition_number for details.

Sparse Inverse Problems: Introduction

4

4.1 Problem Statement 19
4.2 Exhaustive Search 20
4.2.1 The Algorithm 20
4.2.2 Performance Guarantees . 22
4.3 Applications 23

4.1 Problem Statement

Definition 4.1.1 (ℓ_0-pseudonorm) *The ℓ_0-pseudonorm counts the total number of nonzero elements of a vector.*

Definition 4.1.2 (Sparse signals) *A sparse signal is the one which contains only a small number of non-zero elements compared to its dimension.*

A signal $\boldsymbol{x} \in \mathbb{R}^n$ is sparse if

- $\|\boldsymbol{x}\|_0 = S \ll n$.
- $\boldsymbol{x} = \boldsymbol{B}\boldsymbol{c}$ and $\|\boldsymbol{c}\|_0 = S \ll n$.

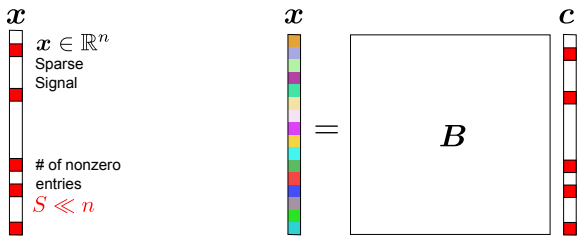


Figure 4.1: Sparse Signals

In reality, sparse signals are rare but compressible signals are ubiquitous.

Compressible signals are those that can be approximated by sparse signals.

- Natural pictures are compressible under DCT/Wavelet transform.
- Communication signals are often compressible under Fourier transform.
- In function approximation, it is typically assumed that the unknown function can be well approximated by a few 'kernel' functions.

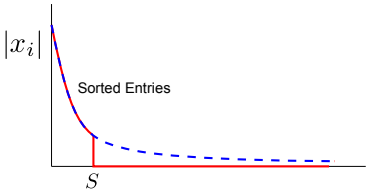


Figure 4.2: Compressible Signals

Sparse Inverse Problem

Given the observations $\boldsymbol{y}$ and the matrix $\boldsymbol{A}$, try to find a sparse $\boldsymbol{x}$ such that $\boldsymbol{y} \approx \boldsymbol{A}\boldsymbol{x}$.

In mathematical terms:

$$\hat{\boldsymbol{x}}_{\text{spa}} = \arg \min_{\boldsymbol{x}: \boldsymbol{y} \approx \boldsymbol{A}\boldsymbol{x}} \|\boldsymbol{x}\|_0 \tag{4.1}$$

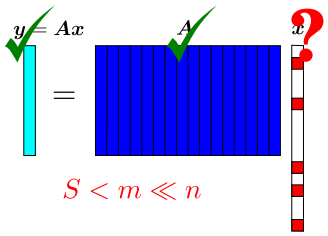


Figure 4.3: Sparse inverse problem

4.2 Exhaustive Search

Challenges

It is very challenging to solve sparse inverse problem (4.1).

A Look at Least Squares Solution:

- In big data applications, it is typically $m << n$. Hence there are infinitely many feasible solutions.
  Define
  $$\mathcal{X} := \{\boldsymbol{x} : \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}\}.$$
  When $m < n$, $|\mathcal{X}| = \infty$.
- The least squares solution
  $$\hat{\boldsymbol{x}} = \boldsymbol{A}^{\dagger}\boldsymbol{y} = \arg \min_{\boldsymbol{x}:\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}} \|\boldsymbol{x}\|_2$$
  is typically *not* sparse.

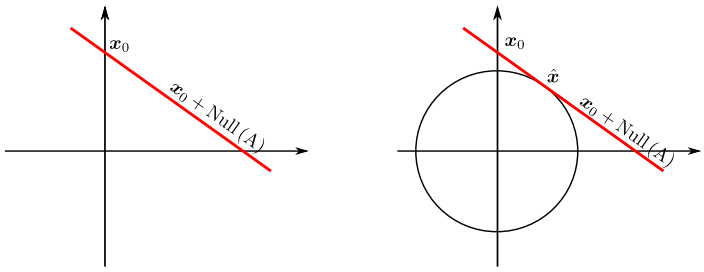


Figure 4.4: $|\mathcal{X}| = \infty$ and $\hat{\boldsymbol{x}}$ is not sparse.

A Look at ℓ_0-pseudonorm

Normal optimization techniques cannot be applied to solve (4.1).

- ℓ_0 pseudo-norm is discontinuous[1] and nonconvex[2].
  - Discontinuity ⇒ Gradient based technique cannot be applied.
  - Nonconvexity ⇒ Global optimality is not guaranteed.

1: Draw contours of ℓ_0-norm in 1-D and 2-D space. Convince yourself that ℓ_0-norm is not continuous.

2: We will discuss convexity in detail later.

4.2.1 The Algorithm

Definition 4.2.1 *Let $\boldsymbol{x} \in \mathbb{R}^n$. Its support set is defined as*

$$supp\,(\boldsymbol{x}) = \{i : x_i \neq 0\}\,.$$

Example 4.2.1

$$\begin{aligned} & \boldsymbol{x} = [0.1,\ 0,\ 0,\ -1,\ 0]^T \\ \Rightarrow\ & \mathcal{I} = \{1, 4\}\,. \end{aligned}$$

Definition 4.2.2 *Let $\mathcal{I} \subset \{1, 2, \cdots, n\}$ be an index set. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{x} \in \mathbb{R}^n$.*

- *$\boldsymbol{A}_{\mathcal{I}}$: a matrix formed by columns of $\boldsymbol{A}$ indexed by $\mathcal{I}$.*
- *$\boldsymbol{x}_{\mathcal{I}}$: a vector formed by entries of $\boldsymbol{A}$ indexed by $\mathcal{I}$.*

Example 4.2.2 Suppose that $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ where $\boldsymbol{x}$ is sparse. Let $\mathcal{I} = \text{supp}\,(\boldsymbol{x})$. Then $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}}$.

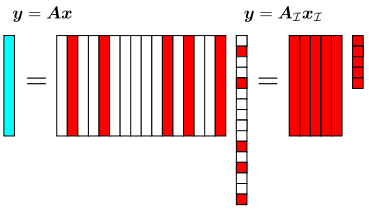


Figure 4.5: $y = Ax = A_{\mathcal{I}}x_{\mathcal{I}}$

Exhaustive Search Algorithm

1. For $s = 1, 2, \cdots$
   a) Try all $\mathcal{I} \subset [n] \triangleq \{1, 2, \cdots, n\}$ s.t. $|\mathcal{I}| = s$.
   b) If $\boldsymbol{y} = \boldsymbol{A}_{\mathcal{I}}\hat{\boldsymbol{x}}_{\mathcal{I}}$, then terminate the search. Otherwise, continue.
2. Finalization: Set $\boldsymbol{x}_{\mathcal{I}} = \boldsymbol{A}_{\mathcal{I}}^{\dagger}\boldsymbol{y}$ and $\boldsymbol{x}_{\mathcal{I}^c} = \boldsymbol{0}$.

Exhaustive search is not practical.

- Exhaustive search terminates when $S^{\#} = \|\boldsymbol{x}\|_0$.
- The computational complexity is approximately

$$\sum_{s=1}^{S^{\#}} \binom{n}{s} \geq \binom{n}{S^{\#}} = \frac{n!}{S^{\#}!\,(n - S^{\#})!}.$$

- Use *Stirling approximation*: $n! \approx \sqrt{2\pi}n^{n+1/2}e^{-n}$. It holds that the complexity is

$$O\left(n^{S^{\#}}\right).$$

Exhaustive search is not practical for large n.

Example 4.2.3 (Finding Sparse Solutions) Given

1.
$$\begin{bmatrix} 2 \\ 2 \\ -2 \\ -2 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 & 2 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 0 & 2 & 0 & 0 \\ 1 & 1 & -1 & -1 & 0 & 0 & 2 & 0 \\ 1 & -1 & -1 & 1 & 0 & 0 & 0 & 2 \end{bmatrix}\boldsymbol{x}.$$

2.
$$\begin{bmatrix} 2 \\ 2 \\ 0 \\ 0 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 & 2 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 0 & 2 & 0 & 0 \\ 1 & 1 & -1 & -1 & 0 & 0 & 2 & 0 \\ 1 & -1 & -1 & 1 & 0 & 0 & 0 & 2 \end{bmatrix}\boldsymbol{x}.$$

3.

$$\begin{bmatrix} 2 \\ 3 \\ 0 \\ 0 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 & 2 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 0 & 2 & 0 & 0 \\ 1 & 1 & -1 & -1 & 0 & 0 & 2 & 0 \\ 1 & -1 & -1 & 1 & 0 & 0 & 0 & 2 \end{bmatrix} \boldsymbol{x}.$$

- What would the sparsest solution be?
- Is the sparsest solution unique?

4.2.2 Performance Guarantees

The question:

- Let $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0$ where $\|\boldsymbol{x}_0\|_0 = S < m$.
- Let $\hat{\boldsymbol{x}}$ be a sparsest solution of $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$.

Under which conditions $\hat{\boldsymbol{x}} = \boldsymbol{x}_0$?

Assumption:
Every m-column sub-matrix of $\boldsymbol{A}$ is of full rank.

- Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ $(m < n)$ be generated from standard Gaussian distribution.
- The generated $\boldsymbol{A}$ satisfies the assumption with probability one.

Theorem 4.2.1 (Worst Case Performance Guarantees) *If $m > 2S$, then $\hat{\boldsymbol{x}} = \boldsymbol{x}_0$ for all S-sparse signals $\boldsymbol{x}_0$.*

Proof. Suppose that there exists an $\boldsymbol{x}_1$ such that

$$\boldsymbol{x}_1 \neq \boldsymbol{x}_0, \ \|\boldsymbol{x}_1\|_0 \leq S, \ \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_1.$$

Then

$$\boldsymbol{0} = \boldsymbol{A}(\boldsymbol{x}_0 - \boldsymbol{x}_1).$$

Note that

$$\|\boldsymbol{x}_0 - \boldsymbol{x}_1\|_0 \leq 2S < m$$

This contradicts that every m-columns of $\boldsymbol{A}$ are linearly independent. □

Theorem 4.2.2 (Average Case Performance Guarantees) *If $m > S$, then $\hat{\boldsymbol{x}} = \boldsymbol{x}_0$ for nearly all S-sparse signals $\boldsymbol{x}_0$.*

Proof. Suppose that there exists an $\boldsymbol{x}_1$ such that

$$\boldsymbol{x}_1 \neq \boldsymbol{x}_0, \ \|\boldsymbol{x}_1\|_0 \leq S, \ \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_1.$$

Let $\mathcal{T}_0 = \text{supp}(\boldsymbol{x}_0)$ and $\mathcal{T}_1 = \text{supp}(\boldsymbol{x}_1)$.

1. That $\boldsymbol{x}_1 \neq \boldsymbol{x}_0$ implies that $\mathcal{T}_0 \neq \mathcal{T}_1$ and $\text{span}\left(\boldsymbol{A}_{\mathcal{T}_0}\right) \neq \text{span}\left(\boldsymbol{A}_{\mathcal{T}_1}\right)$. Otherwise, $\boldsymbol{x}_0 = \boldsymbol{A}_{\mathcal{T}_0}^{\dagger}\boldsymbol{y} = \boldsymbol{A}_{\mathcal{T}_1}^{\dagger}\boldsymbol{y} = \boldsymbol{x}_1$.

2. $\boldsymbol{y} \in \text{span}(\boldsymbol{A}_{\mathcal{T}_0}) \cap \text{span}(\boldsymbol{A}_{\mathcal{T}_1}) =: \mathcal{A}_\cap$.
   This follows from that

$$\begin{aligned}\boldsymbol{y} &= \boldsymbol{A}\boldsymbol{x}_0 = \boldsymbol{A}_{\mathcal{T}_0}\boldsymbol{x}_{0,\mathcal{T}_0} \\ &= \boldsymbol{A}\boldsymbol{x}_1 = \boldsymbol{A}_{\mathcal{T}_1}\boldsymbol{x}_{1,\mathcal{T}_1}.\end{aligned}$$

3. $\dim(\mathcal{A}_\cap) < \dim(\text{span}(\boldsymbol{A}_{\mathcal{T}_0})) = \dim(\text{span}(\boldsymbol{A}_{\mathcal{T}_1}))$.
   This follows from that $\text{span}\left(\boldsymbol{A}_{\mathcal{T}_0}\right) \neq \text{span}\left(\boldsymbol{A}_{\mathcal{T}_1}\right)$.
4. For any $\boldsymbol{x}_0$ generated from a continuous distribution,

$$\Pr\left(\boldsymbol{y} \in \text{span}(\boldsymbol{A}_{\mathcal{T}_0}) \cap \text{span}(\boldsymbol{A}_{\mathcal{T}_1})\right) = 0.$$

□

4.3 Applications

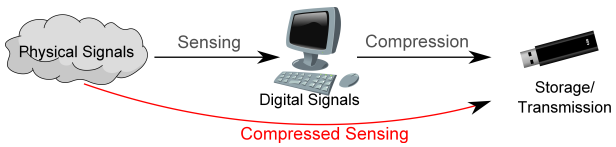
Compressed Sensing

Challenges: In many applications, the cost (in physical space, time, spectrum, or operations) of sensing can be very high.

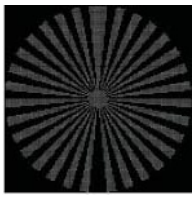
- Magnetic Resonance Imaging (MRI):

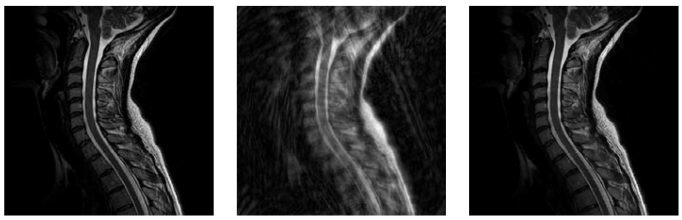
- Infrared sensing:

Compressed sensing: A paradigm shift to reduce the cost/time of sensing.



An example in MRI:



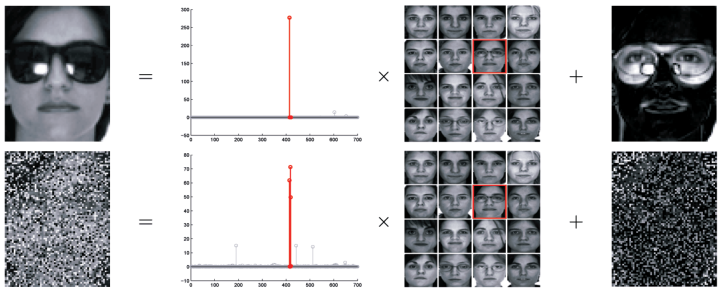


Fully sampled | 6 × undersampled classical | 6 × undersampled CS

Trzasko, Manduca, Borisch (Mayo Clinic)

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \text{ subject to } \boldsymbol{y} \approx \boldsymbol{F}_{\Omega,:}\boldsymbol{x}.$$

Robust Face Recognition



[Wright et al., 2009]

The setup:

- A set of training samples $\{\boldsymbol{\phi}_i, l_i\}$
  - $\boldsymbol{\phi}_i \in \mathbb{R}^m$ is the vector representation of the images.
  - $l_i \in \{1, 2, \cdots, C\}$ label for the C subjects.
- Test sample $\boldsymbol{y}$
- Assumption:
  - For simplicity, assume a good face alignment.

Face recognition via sparse regression:

- Images of the same subject i form a low-dimensional linear subspace in $\mathbb{R}^m$.

$$\boldsymbol{y} \approx \sum_{\{j|l_j=i\}} \boldsymbol{\phi}_j c_j =: \boldsymbol{\Phi}_i \boldsymbol{c}_i,$$

and

$$\boldsymbol{y} \approx [\boldsymbol{\Phi}_1, \boldsymbol{\Phi}_2, \cdots, \boldsymbol{\Phi}_C]\, \boldsymbol{c} = \boldsymbol{\Phi}\boldsymbol{c} \in \mathbb{R}^m$$
$$\text{where } \boldsymbol{c} = \left[\cdots, \boldsymbol{0}^T, \boldsymbol{c}_i^T, \boldsymbol{0}^T, \cdots\right]^T.$$

- Face recognition: the coefficient vector $\boldsymbol{c}$ is group sparse.

$$\min_{\boldsymbol{c}} \|\boldsymbol{c}\|_{2,0} \text{ s.t. } \boldsymbol{y} \approx \boldsymbol{\Phi}\boldsymbol{c}.$$

Robust face recognition:

- Model corruption and occlusion as

$$\boldsymbol{y} \approx \boldsymbol{\Phi c} + \boldsymbol{e},$$

  where $\boldsymbol{e}$ is an unknown error vector whose entries can be very large.
- Both coefficient vector $\boldsymbol{c}$ and noise vector $\boldsymbol{e}$ are sparse.

$$\min_{\boldsymbol{c},\boldsymbol{e}} \|\boldsymbol{c}\|_{2,0} + \|\boldsymbol{e}\|_0 \text{ s.t. } \boldsymbol{y} \approx \boldsymbol{\Phi c} + \boldsymbol{e}.$$

Sparse Inverse Problems: L0 Methods

5

5.1 Some Easy Problems

5.1 Some Easy Problems . . . 26
5.2 OMP . . . 27
5.2.1 Algorithm . . . 27
5.2.2 Performance Guarantees . 28
5.3 SP and Similar Algorithms 30
5.3.1 Algorithms . . . 30
5.3.2 Performance Guarantees . 32
5.4 Majorization Minimization (MM) . . . 37
5.5 Appendix . . . 39

L0 Indicator Function

Definition 5.1.1 *Define indicator function*

$$\delta(\cdot) = \begin{cases} 0 & \text{if the statement is true,} \\ \infty & \text{otherwise.} \end{cases}$$

Example 5.1.1 Find the value of the indicator function $\delta(\|\boldsymbol{x}\|_0 \leq 1)$ for

- $\boldsymbol{x} = [0, 0, 0]^\mathsf{T}$,
- $\boldsymbol{x} = [1, 0, 0]^\mathsf{T}$, and
- $\boldsymbol{x} = [1, 2, 0]^\mathsf{T}$.

We consider some easy ℓ_0 minimization problems, which help design efficient algorithms for general sparse inverse problems.

Example 5.1.2 Consider the optimization problem

$$\min_{\boldsymbol{x}} \ \delta(\|\boldsymbol{x}\|_0 \leq 1) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|^2.$$

Find the optimal solution $\boldsymbol{x}^\star$ and the corresponding optimal value of the objective function for the following choices of $\boldsymbol{z}$.

- $\boldsymbol{z} = [0, 0, 0]^\mathsf{T}$,
- $\boldsymbol{z} = [1, 2, 3]^\mathsf{T}$, and
- $\boldsymbol{z} = [1, 2, 2]^\mathsf{T}$.

Definition 5.1.2 (Moreau Envelope and Proximal Function) *For a given function f and a given constant $\gamma > 0$, its Moreau envelope is defined as*

$$\mathcal{M}_{\gamma f}(\boldsymbol{z}) := \min_{\boldsymbol{x}} \ f(\boldsymbol{x}) + \frac{1}{2\gamma}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2,$$

and the proximal operator is given by [1]

$$\boldsymbol{x}^\star = Prox_{\gamma f}(\boldsymbol{z}) := \arg\min_{\boldsymbol{x}} \ f(\boldsymbol{x}) + \frac{1}{2\gamma}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2.$$

Example 5.1.3 Let $f(\boldsymbol{x}) = \delta(\|\boldsymbol{x}\|_0 \leq S)$ for some $S \in \mathbb{Z}_+$. Find the corresponding Moreau envelope $\mathcal{M}_{\gamma f}(\boldsymbol{z})$ and proximal operator $\text{Prox}_{\gamma f}(\boldsymbol{z})$ for some $\boldsymbol{z} \in \mathbb{Z}^n$. Discuss the continuity of them. Discuss how the parameter γ affects the solutions. [2]

1: Moreau envelope and proximal operator are very important in modern optimization theory and algorithms.

2: For any given vector $\boldsymbol{z} \in \mathbb{R}^n$, let $|z_{i_1}| \geq |z_{i_2}| \geq \cdots \geq |z_{i_n}|$ be the sequence of magnitude-sorted elements. The closed forms of Moreau envelope and proximal operator can be written using this sequence.

L0 Function

Example 5.1.4 Let $f(\boldsymbol{x}) = \|\boldsymbol{x}\|_0$. Find the corresponding Moreau envelope $\mathcal{M}_{\gamma f}(\boldsymbol{z})$ and proximal operator $\text{Prox}_{\gamma f}(\boldsymbol{z})$ for some $\boldsymbol{z} \in \mathbb{R}^n$. Discuss the continuity of them. Discuss how the parameter γ affects the solutions.

If you have difficulties in solving this problem, the following simplified case may give you clues.

Consider the scalar case, i.e., $x \in \mathbb{R}$. One has $f(x) = 0$ if $x = 0$ and $f(x) = 1$ if $x \neq 0$. Let $\gamma = 1$.Then

$$\mathcal{M}_{\|\cdot\|_0}(z) = \min_x \ \|x\|_0 + \frac{1}{2}(x - z)^2.$$

1. Show that
$$\mathcal{M}_{\|\cdot\|_0}(z) \leq \|z\|_0 + \frac{1}{2}(z - z)^2 \leq 1.$$
2. For $z = 0, 1, 2$, find $\mathcal{M}_{\|\cdot\|_0}(z)$ and $x^\star = \text{Prox}_{\|\cdot\|_0}(z)$ respectively.

Towards General Problems

Example 5.1.5 Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ be a matrix of which columns are normalized, i.e., $\|\boldsymbol{A}_{:,n}\|_2 = 1$. Consider the optimization problem

$$\min_{\boldsymbol{x}} \ \delta(\|\boldsymbol{x}\|_0 \leq 1) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2.$$

Find the optimal solution $\boldsymbol{x}^\star$ and the corresponding optimal value of the objective function. [3]

3: The results in Example **??** are useful to justify the solutions.

While the above problem can be easily solved, the problem

$$\min_{\boldsymbol{x}} \ \delta(\|\boldsymbol{x}\|_0 \leq S) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$$

for $S > 1$ is much more difficult to solve.

5.2 OMP

5.2.1 Algorithm

The intuition: We solve the difficult problem

$$\min_{\boldsymbol{x}} \ \delta(\|\boldsymbol{x}\|_0 \leq S) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$$

by solving a sequence of easier problems

$$\min_{\boldsymbol{x}} \ \delta(\|\boldsymbol{x}\|_0 \leq 1) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2.$$

Algorithm 1: Orthogonal Matching Pursuit (OMP)

Require: $\boldsymbol{y}, \boldsymbol{A}, S$
Ensure: $\hat{\boldsymbol{x}}$ such that $\|\hat{\boldsymbol{x}}\|_0 \leq S$
1: **Initialization**: $\ell = 1$, $\hat{\boldsymbol{x}} = \boldsymbol{0}$, $\mathcal{T}^\ell = \phi$, and $\boldsymbol{y}_r = \boldsymbol{y}$.
2: **while** $\ell \leq S$ **do**
3: $\quad i_\ell = \arg\max_j \left|\left\langle \boldsymbol{a}_j, \boldsymbol{y}_r \right\rangle\right|$
4: $\quad \mathcal{T}^\ell = \mathcal{T}^{\ell-1} \bigcup \{i_\ell\}$ {Add one index}
5: $\quad \hat{\boldsymbol{x}}_{\mathcal{T}^\ell} = \boldsymbol{A}_{\mathcal{T}^\ell}^\dagger \boldsymbol{y}$.
6: $\quad \boldsymbol{y}_r = \boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{x}}$.
7: $\quad \ell = \ell + 1$.
8: **end while**

5.2.2 Performance Guarantees

Definition 5.2.1 (Mutual coherence) *The mutual coherence of a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, denoted by $\mu(\boldsymbol{A})$, is the maximal correlation (in magnitude) between two (normalized) columns.*

$$\mu(\boldsymbol{A}) = \max_{i\neq j} \frac{\left|\left\langle \boldsymbol{a}_i, \boldsymbol{a}_j \right\rangle\right|}{\|\boldsymbol{a}_i\|_2 \left\|\boldsymbol{a}_j\right\|_2}.$$

When $\|\boldsymbol{a}_i\|_2 = 1,\ \forall i \in [n]$, then $\mu(\boldsymbol{A}) = \max_{i\neq j} \left|\left\langle \boldsymbol{a}_i, \boldsymbol{a}_j \right\rangle\right|$.

Theorem 5.2.1 *Suppose that $\boldsymbol{A}$ satisfies that*

$$\mu < \frac{1}{2S}.$$

Then the OMP algorithm is guaranteed to exactly recover all *S-sparse $\boldsymbol{x}$. That is, $\hat{\boldsymbol{x}} = \boldsymbol{x}_0$.*

The key for the proof: To show $\hat{\boldsymbol{x}} = \boldsymbol{x}_0$:

- Want to show that $\text{supp}(\hat{\boldsymbol{x}}) = \text{supp}(\boldsymbol{x}_0)$.
- Or show that at the ℓ-th iteration of OMP, the chosen index $i_\ell \in \mathcal{T}_0 := \text{supp}(\boldsymbol{x}_0)$.

The proof needs Cauchy-Schwartz Inequality: [4]

4: Also see Theorem 5.5.2.

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^n |x_i| \leq \sqrt{n}\,\|\boldsymbol{x}\|_2.$$

Proof. The proof is divided into two steps. The first step concerns the first iteration and proves that show that $i_1 := \arg\max_i \left|\left\langle \boldsymbol{a}_i, \boldsymbol{y} \right\rangle\right| \in \mathcal{T}_0$. In the second step, a very similar technique is used to show that $i_\ell \in \mathcal{T}_0$.

Step 1: $i_1 \in \mathcal{T}_0$.

It holds that $\forall i$,

$$\begin{aligned}\left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\right| &= \left|\left\langle \boldsymbol{a}_i, \sum_{j\in\mathcal{T}_0} \boldsymbol{a}_j x_{0,j}\right\rangle\right| \\ &= \left|\sum_{j\in\mathcal{T}_0} x_{0,j}\left\langle \boldsymbol{a}_i, \boldsymbol{a}_j\right\rangle\right|.\end{aligned}$$

1. For all $i \notin \mathcal{T}_0$:

$$\begin{aligned}\left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\right| &= \left|\sum_{j\in\mathcal{T}_0} x_{0,j}\left\langle \boldsymbol{a}_i, \boldsymbol{a}_j\right\rangle\right| \le \sum_{j\in\mathcal{T}_0} \left|x_{0,j}\right|\left|\left\langle \boldsymbol{a}_i, \boldsymbol{a}_j\right\rangle\right| \\ &\le \mu \sum_{j\in\mathcal{T}_0} \left|x_{0,j}\right| \overset{(a)}{\le} \mu\sqrt{S}\,\|\boldsymbol{x}\|_2\end{aligned}$$

   where (a) follows from Cauchy–Schwartz Inequality (Theorem 5.5.2). Hence,

$$\max_{i\notin\mathcal{T}_0} \left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\right| \le \mu\sqrt{S}\,\|\boldsymbol{x}\|_2\,. \tag{5.1}$$

2. For all $i \in \mathcal{T}_0$:

$$\begin{aligned}\left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\right| &= \left|\sum_{j\in\mathcal{T}_0} x_{0,j}\left\langle \boldsymbol{a}_i, \boldsymbol{a}_j\right\rangle\right| \ge \left|x_{0,i}\left\langle \boldsymbol{a}_i, \boldsymbol{a}_i\right\rangle\right| - \left|\sum_{j\neq i} x_{0,j}\left\langle \boldsymbol{a}_i, \boldsymbol{a}_j\right\rangle\right| \\ &\ge |x_{0,i}| - \mu\sum_{j\neq i}\left|x_{0,j}\right| \overset{(a)}{\ge} |x_{0,i}| - \mu\sqrt{S}\,\|\boldsymbol{x}\|_2\,,\end{aligned}$$

   where (a) follows from Cauchy-Schwartz Inequality.
   As a result,

$$\max_{i\in\mathcal{T}_0} \left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\right| \ge \frac{1}{\sqrt{S}}\|\boldsymbol{x}\|_2 - \mu\sqrt{S}\,\|\boldsymbol{x}\|_2\,, \tag{5.2}$$

   where we have used the fact that

$$\frac{1}{\sqrt{S}}\|\boldsymbol{x}\|_2 = \frac{\left(\sum x_i^2\right)^{\frac{1}{2}}}{\sqrt{S}} \le \frac{\left(\sum\left(\max_i\ |x_i|\right)^2\right)^{\frac{1}{2}}}{\sqrt{S}} = \max_{i\in\mathcal{T}_0}\ |x_i|\,.$$

3. Now suppose that $\mu < \frac{1}{2S}$ (the assumption in Theorem 5.2.1). Then

$$\frac{1}{\sqrt{S}}\|\boldsymbol{x}\|_2 > 2\mu\sqrt{S}\,\|\boldsymbol{x}\|_2\,,$$

   Or equivalently,

$$\max_{i\in\mathcal{T}_0} \left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\right| \ge \frac{1}{\sqrt{S}}\|\boldsymbol{x}\|_2 - \mu\sqrt{S}\,\|\boldsymbol{x}\|_2 > \mu\sqrt{S}\,\|\boldsymbol{x}\|_2 \ge \max_{i\notin\mathcal{T}_0} \left|\langle \boldsymbol{a}_i, \boldsymbol{y}\rangle\right|.$$

   One concludes that

$$i_1 \in \mathcal{T}_0.$$

Step 2: $i_\ell \in \mathcal{T}_0$. (Mathematical induction)

Let $i_1, \cdots, i_{\ell-1}$ be the indices chosen in the first $\ell - 1$ iterations. Let $\mathcal{T}^{\ell-1} = \{i_1, \cdots, i_{\ell-1}\}$. Assume that $\mathcal{T}^{\ell-1} \subset \mathcal{T}_0$.

Then

$$\boldsymbol{y}_r = \boldsymbol{y} - \boldsymbol{A}_{\mathcal{T}^{\ell-1}} \boldsymbol{A}_{\mathcal{T}^{\ell-1}}^{\dagger} \boldsymbol{y} = \boldsymbol{y} - \boldsymbol{A}_{\mathcal{T}^{\ell-1}} \hat{\boldsymbol{x}}_{\ell-1} \in \operatorname{span}\left(\boldsymbol{A}_{\mathcal{T}_0}\right).$$

Or

$$\boldsymbol{y}_r = \boldsymbol{A}_{\mathcal{T}_0} \tilde{\boldsymbol{v}}_{\mathcal{T}_0}.$$

for some $\tilde{\boldsymbol{v}}_{\mathcal{T}_0}$.

Use the same arguments as before, $i_\ell \in \mathcal{T}_0$.

Final Step: At the same time, $\boldsymbol{A}_{\mathcal{T}^{\ell-1}}^T \boldsymbol{y}_r = \boldsymbol{0}$ and hence $i_\ell \notin \mathcal{T}^{\ell-1}$. So $\left|\mathcal{T}^\ell\right| = \ell$.

OMP algorithm needs S iterations to recover S-sparse signals. □

5.3 SP and Similar Algorithms

The intuition: If

- columns of $\boldsymbol{A}$ are normalised,
- columns of $\boldsymbol{A}$ are near orthogonal,

then $\boldsymbol{A}^\mathsf{T}\boldsymbol{y} = \boldsymbol{A}^\mathsf{T}\boldsymbol{A}\boldsymbol{x} \approx \boldsymbol{I}\boldsymbol{x}$. We then try to solve

$$\min_{\boldsymbol{x}} \ \delta(\|\boldsymbol{x}\|_0 \le S) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{A}^\mathsf{T}\boldsymbol{y}\|^2$$

Key Differences Between OMP and Other Greedy Algorithms

- OMP
  - One index is added per iteration.
  - Analysis is based on near orthogonality of column pairs.
- SP, CoSaMP, IHT
  - Multiple indices are updated per iteration.
  - Analysis is based on near orthogonality of sub-matrices.

5.3.1 Algorithms

Hard thresholding function $H_S(\boldsymbol{a})$:

$$H_S(\boldsymbol{a}) = \operatorname{Prox}_{\delta(\|\cdot\|_0 \le S)}.$$

Set all but the largest (in magnitude) S elements of $\boldsymbol{a}$ to zero.

Example 5.3.1 $\boldsymbol{a} = [3, -4, 1]^T$.

- $H_1(\boldsymbol{a}) = [0, -4, 0]^T$ and $\operatorname{supp}(H_1(\boldsymbol{a})) = \{2\}$.

Subspace pursuit (SP) algorithm

Algorithm 2: Subspace Pursuit (SP)

Require: $\boldsymbol{y}, \boldsymbol{A}, S$

Ensure: $\hat{\boldsymbol{x}}$ such that $\|\hat{\boldsymbol{x}}\|_0 \leq S$

1: **Initialization**: $\mathcal{T}^0 = \text{supp}\left(H_S\left(\boldsymbol{A}^T\boldsymbol{y}\right)\right)$, and $\boldsymbol{y}_r = \text{resid}\left(\boldsymbol{y}, \boldsymbol{A}_{\mathcal{T}^0}\right)$.
2: **while** $\ell = 1, 2, \cdots$ until exit criteria are satisfied **do**
3: $\quad \tilde{\mathcal{T}}^\ell = \mathcal{T}^{\ell-1} \bigcup \text{supp}\left(H_S\left(\boldsymbol{A}^T\boldsymbol{y}_r\right)\right)$. {Expand support}
4: $\quad$ Let $\boldsymbol{b}_{\tilde{\mathcal{T}}^\ell} = \boldsymbol{A}^\dagger_{\tilde{\mathcal{T}}^\ell}\boldsymbol{y}$ and $\boldsymbol{b}_{\left(\tilde{\mathcal{T}}^\ell\right)^c} = \boldsymbol{0}$. {Estimate a $2S$-sparse signal}
5: $\quad$ Set $\mathcal{T}^\ell = \text{supp}\left(H_S\left(\boldsymbol{b}\right)\right)$. {Shrink support}
6: $\quad$ Let $\boldsymbol{x}^\ell_{\mathcal{T}^\ell} = \boldsymbol{A}^\dagger_{\mathcal{T}^\ell}\boldsymbol{y}$ and $\boldsymbol{x}^\ell_{\left(\mathcal{T}^\ell\right)^c} = \boldsymbol{0}$. {Estimate S-sparse signal}
7: $\quad$ Let $\boldsymbol{y}_r = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^\ell$. {Compute estimation error}
8: **end while**

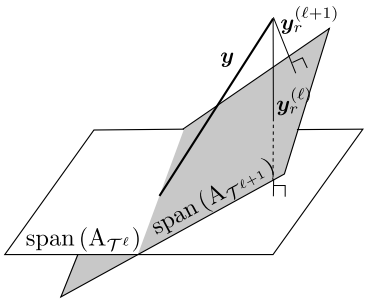


Figure 5.1: A geometric interpretation

Compressive Sampling Matching Pursuit (CoSaMP) Algorithm

Algorithm 3: Compressive Sampling Matching Pursuit (CoSaMP)

Require: $\boldsymbol{y}, \boldsymbol{A}, S$

Ensure: $\hat{\boldsymbol{x}}$ such that $\|\hat{\boldsymbol{x}}\|_0 \leq S$

1: **Initialization**: $\boldsymbol{x}^0 = \boldsymbol{0}$ and $\boldsymbol{y}_r = \boldsymbol{y}$.
2: **while** $\ell = 1, 2, \cdots$ until exit criteria are satisfied **do**
3: $\quad \tilde{\mathcal{T}}^\ell = \mathcal{T}^{\ell-1} \bigcup \text{supp}\left(H_{2S}\left(\boldsymbol{A}^T\boldsymbol{y}_r\right)\right)$. {Expand support}
4: $\quad$ Let $\boldsymbol{b}_{\tilde{\mathcal{T}}^\ell} = \boldsymbol{A}^\dagger_{\tilde{\mathcal{T}}^\ell}\boldsymbol{y}$ and $\boldsymbol{b}_{\left(\tilde{\mathcal{T}}^\ell\right)^c} = \boldsymbol{0}$. {Estimate a $3S$-sparse signal}
5: $\quad \boldsymbol{x}^\ell = H_S\left(\boldsymbol{b}\right)$. ($\mathcal{T}^\ell = \text{supp}\left(H_S\left(\boldsymbol{b}\right)\right)$.) {Shrink support}
6: $\quad \boldsymbol{y}_r = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^\ell$. {Compute estimation error}
7: **end while**

Iterative Hard Thresholding (IHT) Algorithm

Algorithm 4: Iterative Hard Thresholding (IHT)

Require: $\boldsymbol{y}, \boldsymbol{A}, S$

Ensure: $\hat{\boldsymbol{x}}$ such that $\|\hat{\boldsymbol{x}}\|_0 \leq S$

1: **Initialization**: $\boldsymbol{x}^0 = \boldsymbol{0}$.
2: **while** $\ell = 1, 2, \cdots$ until exit criteria are satisfied **do**
3:

$$\boldsymbol{x}^\ell = H_S\left(\boldsymbol{x}^{\ell-1} + \boldsymbol{A}^T\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^{\ell-1}\right)\right)$$

4: **end while**

A more general form: for some step size $\tau > 0$,

$$\boldsymbol{x}^\ell = H_S\left(\boldsymbol{x}^{\ell-1} + \tau\boldsymbol{A}^T\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^{\ell-1}\right)\right).$$

Remark 5.3.1 (History)

- MP: Friedman and Stuetzle, 1981; Mallat and Zhang, 1993; Qian and Chen, 1994.
- OMP: Chen, et al., 1989; Pati, et al., 1993; Davis, et al., 1994.

Analysed by Tropp, 2004.

- SP: Dai and Milenkovic, 2009. (Online available 06/03/2008)
  CoSaMP: Needell and Tropp, 2009. (Online available 17/03/2008)
  IHT: Blumensath and Davies, 2009. (Online available 05/05/2008)

5.3.2 Performance Guarantees

Comparison: Assume that the matrix $\boldsymbol{A}$ is generated as a random Gaussian matrix.

	# of measurements	# of iterations
Exhaustive Search	$2S+1$	$\binom{n}{S} = O\left(n^S\right)$
OMP	$O\left(S^2 \log n\right)$	S
SP, CoSaMP, IHT	$O\left(S \cdot \log \frac{n}{S}\right)$	Typically $O\left(\log S\right)$, at most S

Restricted Isometry Property (RIP)

Intuitively, RIP means near orthogonality of submatrices.

Definition 5.3.1 (Restricted isometry property (RIP) and restricted isometry constant (RIC)) *A matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is said to satisfy the RIP with parameters (K, δ), if for all $\mathcal{T} \subset [n]$ such that $|\mathcal{T}| \leq K$ and for all $\boldsymbol{q} \in \mathbb{R}^{|\mathcal{T}|}$, it holds that*

$$(1-\delta)\left\|\boldsymbol{q}\right\|_2^2 \leq \left\|\boldsymbol{A}_{\mathcal{T}}\boldsymbol{q}\right\|_2^2 \leq (1+\delta)\left\|\boldsymbol{q}\right\|_2^2.$$

The RIC δ_K is defined as the smallest constant δ for which the K-RIP holds, i.e.,

$$\delta_K = \inf\left\{\delta : \ (1-\delta)\left\|\boldsymbol{q}\right\|_2^2 \leq \left\|\boldsymbol{A}_{\mathcal{T}}\boldsymbol{q}\right\|_2^2 \leq (1+\delta)\left\|\boldsymbol{q}\right\|_2^2. \right.$$
$$\left. \forall\, |\mathcal{T}| \leq K, \ \forall \boldsymbol{q} \in \mathbb{R}^{|\mathcal{T}|}\right\}.$$

Remark 5.3.2 (RIP, Eigenvalues and Singular Values) Let $\boldsymbol{B} \in \mathbb{R}^{m \times K}$ be a tall matrix, i.e. $m \geq K$. Then the following statements are equivalent.

- For all $\boldsymbol{q} \in \mathbb{R}^K$,
$$(1-\delta_K)\left\|\boldsymbol{q}\right\|_2^2 \leq \left\|\boldsymbol{B}\boldsymbol{q}\right\|_2^2 \leq (1+\delta_K)\left\|\boldsymbol{q}\right\|_2^2.$$
- $$1-\delta_K \leq \lambda_{\min}\left(\boldsymbol{B}^T\boldsymbol{B}\right) \leq \lambda_{\max}\left(\boldsymbol{B}^T\boldsymbol{B}\right) \leq 1+\delta_K.$$
- $$\sqrt{1-\delta_K} \leq \sigma_{\min}\left(\boldsymbol{B}\right) \leq \sigma_{\max}\left(\boldsymbol{B}\right) \leq \sqrt{1+\delta_K}.$$

Proof of the remark.

$$\begin{aligned}\|\boldsymbol{B}\boldsymbol{q}\|_2^2 &= \|\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{q}\|_2^2 \\ &= \boldsymbol{q}^T\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{q} \\ &= \boldsymbol{q}^T\boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^T\boldsymbol{q} \\ &= \boldsymbol{q}'^T\boldsymbol{\Sigma}^2\boldsymbol{q}' \\ &= \sum_{i=1}^{K}\sigma_i^2 q_i'^2,\end{aligned}$$

where $\boldsymbol{q}' := \boldsymbol{V}^T\boldsymbol{q}$ satisfies $\|\boldsymbol{q}'\|_2^2 = \boldsymbol{q}^T\boldsymbol{V}\boldsymbol{V}^T\boldsymbol{q} = \|\boldsymbol{q}\|_2^2$. □

Theorem 5.3.1 (Monotonicity of RIC) *It holds that* $\delta_1 \le \delta_2 \le \delta_3 \le \cdots$ *(*$\delta_K \le \delta_{K'}$ *for all* $K \le K'$*).*

Proof. Let $\mathcal{Q}_K = \{\boldsymbol{q} \in \mathbb{R}^n : \|\boldsymbol{q}\|_0 \le K,\ \|\boldsymbol{q}\|_2 \le 1\}$. It is clear that $\mathcal{Q}_K \subset \mathcal{Q}_{K'}$ if $K \le K'$.

Then it holds that

$$\delta_K := \sup_{\boldsymbol{q}\in\mathcal{Q}_K}\left(\|\boldsymbol{A}\boldsymbol{q}\|_2^2 - 1\right) \le \sup_{\boldsymbol{q}\in\mathcal{Q}_{K'}}\left(\|\boldsymbol{A}\boldsymbol{q}\|_2^2 - 1\right) =: \delta_{K'}.$$

□

Theorem 5.3.2 (Near Orthogonality of Submatrices) *Let* $\mathcal{I}, \mathcal{J} \subset [n]$ *be two disjoint sets, i.e.,* $\mathcal{I} \cap \mathcal{J} = \phi$*. For all* $\boldsymbol{a} \in \mathbb{R}^{|\mathcal{I}|}$ *and* $\boldsymbol{b} \in \mathbb{R}^{|\mathcal{J}|}$*,*

$$|\langle \boldsymbol{A}_{\mathcal{I}}\boldsymbol{a}, \boldsymbol{A}_{\mathcal{J}}\boldsymbol{b}\rangle| \le \delta_{|\mathcal{I}|+|\mathcal{J}|}\,\|\boldsymbol{a}\|_2\,\|\boldsymbol{b}\|_2\,, \tag{5.3}$$

and

$$\left\|\boldsymbol{A}_{\mathcal{I}}^{\mathsf{T}}\boldsymbol{A}_{\mathcal{J}}\boldsymbol{b}\right\|_2 \le \delta_{|\mathcal{I}|+|\mathcal{J}|}\,\|\boldsymbol{b}\|_2\,. \tag{5.4}$$

Proof. (5.3) obviously holds when either $\boldsymbol{a}$ or $\boldsymbol{b}$ is zero. Assume $\boldsymbol{a} \neq \boldsymbol{0}$ and $\boldsymbol{b} \neq \boldsymbol{0}$. Define

$$\begin{aligned}\boldsymbol{a}' &= \boldsymbol{a}/\|\boldsymbol{a}\|_2\,, & \boldsymbol{b}' &= \boldsymbol{b}/\|\boldsymbol{b}\|_2\,, \\ \boldsymbol{x}' &= \boldsymbol{A}_{\mathcal{I}}\boldsymbol{a}', & \boldsymbol{y}' &= \boldsymbol{A}_{\mathcal{J}}\boldsymbol{b}'.\end{aligned}$$

Then RIP implies that

$$2\left(1-\delta_{|\mathcal{I}|+|\mathcal{J}|}\right) \le \|\boldsymbol{x}'+\boldsymbol{y}'\|_2^2 = \left\|\begin{bmatrix}\boldsymbol{A}_{\mathcal{I}} & \boldsymbol{A}_{\mathcal{J}}\end{bmatrix}\begin{bmatrix}\boldsymbol{a}' \\ \boldsymbol{b}'\end{bmatrix}\right\|_2^2 \le 2\left(1+\delta_{|\mathcal{I}|+|\mathcal{J}|}\right),$$

$$2\left(1-\delta_{|\mathcal{I}|+|\mathcal{J}|}\right) \le \|\boldsymbol{x}'-\boldsymbol{y}'\|_2^2 = \left\|\begin{bmatrix}\boldsymbol{A}_{\mathcal{I}} & \boldsymbol{A}_{\mathcal{J}}\end{bmatrix}\begin{bmatrix}\boldsymbol{a}' \\ -\boldsymbol{b}'\end{bmatrix}\right\|_2^2 \le 2\left(1+\delta_{|\mathcal{I}|+|\mathcal{J}|}\right),$$

Hence,

$$\langle x', y' \rangle = \frac{\|x' + y'\|_2^2 - \|x' - y'\|_2^2}{4} \leq \delta_{|\mathcal{I}|+|\mathcal{J}|}$$
$$-\langle x', y' \rangle = \frac{\|x' - y'\|_2^2 - \|x' + y'\|_2^2}{4} \leq \delta_{|\mathcal{I}|+|\mathcal{J}|}$$

Therefore,

$$\frac{|\langle A_{\mathcal{I}} a, A_{\mathcal{J}} b \rangle|}{\|a\|_2 \|b\|_2} = |\langle x', y' \rangle| \leq \delta_{|\mathcal{I}|+|\mathcal{J}|}.$$

From (5.3) to (5.4):

$$\begin{aligned} \left\|A_{\mathcal{I}}^{\mathsf{T}} A_{\mathcal{J}} b\right\|_2 &= \max_{q:\, \|q\|_2=1} \left|\langle q, A_{\mathcal{I}}^{\mathsf{T}} A_{\mathcal{J}} b \rangle\right| \\ &= \max_{q:\, \|q\|_2=1} \left|q^{\mathsf{T}} A_{\mathcal{I}}^{\mathsf{T}} A_{\mathcal{J}} b\right| \\ &\leq \max_{q:\, \|q\|_2=1} \delta_{|\mathcal{I}|+|\mathcal{J}|} \left\|q\right\|_2 \|b\|_2 \\ &= \delta_{|\mathcal{I}|+|\mathcal{J}|} \|b\|_2 \end{aligned}$$

□

Why RIP:

- Near orthogonality between columns:
  - $\left|\langle a_i, a_j \rangle\right|$ is small.
- Near orthogonality between submatrices:
  - $\sigma_{\max}\left(A_{\mathcal{I}}^{\mathsf{T}} A_{\mathcal{J}}\right) \leq \delta_{|\mathcal{I}|+|\mathcal{J}|}$ is small.
- Near-orthogonality of submatrices implies near-orthogonality of columns:
  - δ_2 is small.
- Near-orthogonality of columns does not mean near-orthogonality of submatrices.
  - Suppose that

$$A_{\mathcal{I}}^{\mathsf{T}} A_{\mathcal{J}} = \begin{bmatrix} \frac{1}{\ell} & \frac{1}{\ell} & \cdots & \frac{1}{\ell} \\ \frac{1}{\ell} & \frac{1}{\ell} & \cdots & \frac{1}{\ell} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{\ell} & \frac{1}{\ell} & \cdots & \frac{1}{\ell} \end{bmatrix} \in \mathbb{R}^{\ell \times \ell}$$

Then $\sigma\left(A_{\mathcal{I}}^{\mathsf{T}} A_{\mathcal{J}}\right) = 1, 0, \cdots, 0$.

Performance Guarantees: IHT

Theorem 5.3.3 *Suppose that A satisfies the RIP with $\delta_{3S} < 1/\sqrt{32}$, then the k^{th} iteration of IHT obeys*

$$\left\|x_0 - x^k\right\|_2 \leq 2^{-k} \|x_0\|_2 + 5 \|w\|_2 .$$

That is, IHT estimates $\boldsymbol{x}$ with accuracy

$$\left\|\boldsymbol{x}_0 - \boldsymbol{x}^k\right\|_2 \leq 6\left\|\boldsymbol{w}\right\|_2, \text{if } k > k^* = \left\lceil \log_2\left(\frac{\left\|\boldsymbol{x}_0\right\|_2}{\left\|\boldsymbol{w}\right\|_2}\right)\right\rceil. \tag{5.5}$$

Remark 5.3.3 (Optimality) No recovery method can perform fundamentally better.

Suppose that an oracle tells us the support $\mathcal{T}_0$ of $\boldsymbol{x}_0$. Then

$$\begin{cases} \hat{\boldsymbol{x}}_{\mathcal{T}_0} = \left(\boldsymbol{A}_{\mathcal{T}_0}^{\mathsf{T}}\boldsymbol{A}_{\mathcal{T}_0}\right)^{-1}\boldsymbol{A}_{\mathcal{T}_0}^{\mathsf{T}}\boldsymbol{y}, \\ \hat{\boldsymbol{x}}_{\mathcal{T}_0^c} = \boldsymbol{0}. \end{cases}$$

It is clear that $\hat{\boldsymbol{x}}_{\mathcal{T}_0^c} - \boldsymbol{x}_{0,\mathcal{T}_0^c} = \boldsymbol{0}$. At the same time,

$$\hat{\boldsymbol{x}}_{\mathcal{T}_0} - \boldsymbol{x}_{0,\mathcal{T}_0} = \left(\boldsymbol{A}_{\mathcal{T}_0}^{\mathsf{T}}\boldsymbol{A}_{\mathcal{T}_0}\right)^{-1}\boldsymbol{A}_{\mathcal{T}_0}^{\mathsf{T}}\boldsymbol{w}$$

By the RIP property,

$$\frac{1}{\sqrt{1+\delta_S}}\left\|\boldsymbol{w}\right\|_2 \leq \left\|\hat{\boldsymbol{x}} - \boldsymbol{x}_0\right\|_2 \leq \frac{1}{\sqrt{1-\delta_S}}\left\|\boldsymbol{w}\right\|_2.$$

Lemma 5.3.4 *Let $\boldsymbol{r}^k := \boldsymbol{x}_0 - \boldsymbol{x}^k$ ($\boldsymbol{r}^0 = \boldsymbol{x}_0$). Then*

$$\left\|\boldsymbol{r}^{k+1}\right\|_2 \leq \sqrt{8}\delta_{3S}\left\|\boldsymbol{r}^k\right\|_2 + 2\sqrt{1+\delta_S}\left\|\boldsymbol{w}\right\|_2.$$

In particular, if $\delta_{3S} < 1/\sqrt{32}$,

$$\left\|\boldsymbol{r}^{k+1}\right\|_2 \leq 0.5\left\|\boldsymbol{r}^k\right\|_2 + 2.17\left\|\boldsymbol{w}\right\|_2.$$

Back to the main result in Theorem 5.3.3:

$$\begin{aligned} \left\|\boldsymbol{r}^k\right\|_2 &\leq \frac{1}{2}\left\|\boldsymbol{r}^{k-1}\right\|_2 + 2.17\left\|\boldsymbol{w}\right\|_2 \\ &\leq \frac{1}{4}\left\|\boldsymbol{r}^{k-2}\right\|_2 + 2.17\left(1+\frac{1}{2}\right)\left\|\boldsymbol{w}\right\|_2 \\ &\cdots < \frac{1}{2^k}\left\|\boldsymbol{r}^0\right\|_2 + 4.34\left\|\boldsymbol{w}\right\|_2. \end{aligned}$$

Proof of Lemma 5.3.4. Recall that

$$\boldsymbol{x}^{k+1} = H_S\left(\boldsymbol{x}^k + \boldsymbol{A}^T\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^k\right)\right).$$

Define

$$\begin{aligned} \boldsymbol{a}^{k+1} &:= \boldsymbol{x}^k + \boldsymbol{A}^T\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^k\right) \\ &= \boldsymbol{x}_0 - \boldsymbol{x}_0 + \boldsymbol{x}^k + \boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{w} - \boldsymbol{A}\boldsymbol{x}^k\right) \\ &= \boldsymbol{x}_0 + \left(\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)\left(\boldsymbol{x}_0 - \boldsymbol{x}^k\right) + \boldsymbol{A}^T\boldsymbol{w} \\ &= \boldsymbol{x}_0 + \left(\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)\boldsymbol{r}^k + \boldsymbol{A}^T\boldsymbol{w}. \end{aligned} \tag{5.6}$$

Then

$$\begin{aligned} \boldsymbol{x}^{k+1} &= H_S\left(\boldsymbol{x}_0 + \left(\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)\boldsymbol{r}^k + \boldsymbol{A}^T\boldsymbol{w}\right) \\ &= \boldsymbol{x}_0 - \boldsymbol{r}^{k+1} \end{aligned}$$

We will show that $\boldsymbol{r}^{k+1}$ is small.

Let $\mathcal{T}_0 = \text{supp}\left(\boldsymbol{x}_0\right)$, $\mathcal{T}^k = \text{supp}\left(\boldsymbol{x}^k\right)$, and $\mathcal{B}^k = \mathcal{T}_0 \bigcup \mathcal{T}^k$. Note that

- $\boldsymbol{r}^k = \boldsymbol{x}_0 - \boldsymbol{x}^k$ is supported on $\mathcal{B}^k$.
- $\boldsymbol{r}^{k+1} = \boldsymbol{x}_0 - \boldsymbol{x}^{k+1}$ is supported on $\mathcal{B}^{k+1}$.

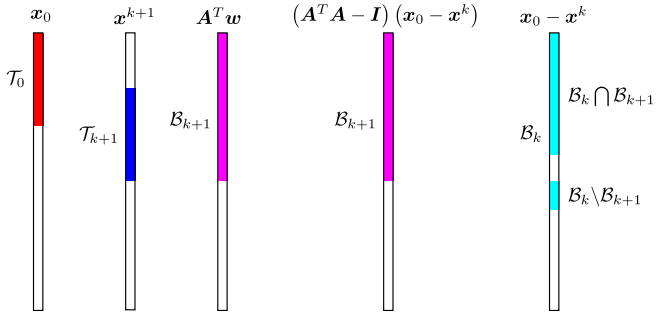


Figure 5.2: $\boldsymbol{x}^{k+1} = H_S\left(\boldsymbol{x}_0 + \left(\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)\boldsymbol{r}^k + \boldsymbol{A}^T\boldsymbol{w}\right)$.

We only need to focus on the set $\mathcal{B}^{k+1}$:

$$\begin{aligned} \left\|\boldsymbol{r}^{k+1}\right\|_2 &= \left\|\boldsymbol{x}_{0,\mathcal{B}^{k+1}} - \boldsymbol{x}^{k+1}_{\mathcal{B}^{k+1}}\right\|_2 \\ &= \left\|\boldsymbol{x}_{0,\mathcal{B}^{k+1}} - \boldsymbol{a}^{k+1}_{\mathcal{B}^{k+1}} + \boldsymbol{a}^{k+1}_{\mathcal{B}^{k+1}} - \boldsymbol{x}^{k+1}_{\mathcal{B}^{k+1}}\right\|_2 \\ &\overset{(a)}{\leq} \left\|\boldsymbol{x}_{0,\mathcal{B}^{k+1}} - \boldsymbol{a}^{k+1}_{\mathcal{B}^{k+1}}\right\|_2 + \left\|\boldsymbol{a}^{k+1}_{\mathcal{B}^{k+1}} - \boldsymbol{x}^{k+1}_{\mathcal{B}^{k+1}}\right\|_2 \\ &\overset{(b)}{\leq} 2\left\|\boldsymbol{x}_{0,\mathcal{B}^{k+1}} - \boldsymbol{a}^{k+1}_{\mathcal{B}^{k+1}}\right\|_2, \end{aligned} \tag{5.7}$$

where (a) has used triangle inequality, and (b) follows from that $\boldsymbol{x}^{k+1}_{\mathcal{B}^{k+1}}$ is the best s-term approximation to $\boldsymbol{a}^{k+1}_{\mathcal{B}^{k+1}}$.

We bound the noise term $\boldsymbol{A}^\mathsf{T}\boldsymbol{w}$:

$$\left\|\left(\boldsymbol{A}^\mathsf{T}\boldsymbol{w}\right)_{\mathcal{B}^{k+1}}\right\|_2 = \left\|\boldsymbol{A}^\mathsf{T}_{\mathcal{B}^{k+1}}\boldsymbol{w}\right\|_2 \leq \sqrt{1+\delta_{2S}}\left\|\boldsymbol{w}\right\|_2.$$

The other term can be bounded as follows.

$$\begin{aligned} &\left(\left(\boldsymbol{I} - \boldsymbol{A}^\mathsf{T}\boldsymbol{A}\right)\boldsymbol{r}^k\right)_{\mathcal{B}^{k+1}} \\ &= \boldsymbol{r}^k_{\mathcal{B}^{k+1}} - \boldsymbol{A}^\mathsf{T}_{\mathcal{B}^{k+1}}\boldsymbol{A}\boldsymbol{r}^k \\ &= \boldsymbol{r}^k_{\mathcal{B}^{k+1}} - \boldsymbol{A}^\mathsf{T}_{\mathcal{B}^{k+1}}\boldsymbol{A}_{\mathcal{B}^{k+1}}\boldsymbol{r}^k_{\mathcal{B}^{k+1}} - \boldsymbol{A}^\mathsf{T}_{\mathcal{B}^{k+1}}\boldsymbol{A}_{\mathcal{B}^k\backslash\mathcal{B}^{k+1}}\boldsymbol{r}^k_{\mathcal{B}^k\backslash\mathcal{B}^{k+1}} \\ &= \left(\boldsymbol{I} - \boldsymbol{A}^\mathsf{T}_{\mathcal{B}^{k+1}}\boldsymbol{A}_{\mathcal{B}^{k+1}}\right)\boldsymbol{r}^k_{\mathcal{B}^{k+1}} - \boldsymbol{A}^\mathsf{T}_{\mathcal{B}^{k+1}}\boldsymbol{A}_{\mathcal{B}^k\backslash\mathcal{B}^{k+1}}\boldsymbol{r}^k_{\mathcal{B}^k\backslash\mathcal{B}^{k+1}}. \end{aligned}$$

Hence

$$\begin{aligned} &\left\|\left(\left(\boldsymbol{I} - \boldsymbol{A}^\mathsf{T}\boldsymbol{A}\right)\boldsymbol{r}^k\right)_{\mathcal{B}^{k+1}}\right\|_2 \\ &\leq \delta_{2S}\left\|\boldsymbol{r}^k_{\mathcal{B}^{k+1}}\right\|_2 + \delta_{3S}\left\|\boldsymbol{r}^k_{\mathcal{B}^k\backslash\mathcal{B}^{k+1}}\right\|_2 \\ &\leq \sqrt{2}\delta_{3S}\left\|\boldsymbol{r}^k\right\|_2, \end{aligned}$$

where

- The 1st term follows from $\left|\mathcal{B}^{k+1}\right| \leq 2S$ and RIP.
- The 2nd term follows from Theorem 5.3.2.
- The last term uses $\delta_{2S} \leq \delta_{3S}$ (Theorem 5.3.1) and Cauchy-Schwartz Inequality

$$\begin{aligned}
&\left\|\boldsymbol{r}^k_{\mathcal{B}^{k+1}}\right\|_2 + \left\|\boldsymbol{r}^k_{\mathcal{B}^k \setminus \mathcal{B}^{k+1}}\right\|_2 \\
&\leq \sqrt{2}\left(\left\|\boldsymbol{r}^k_{\mathcal{B}^{k+1}}\right\|_2^2 + \left\|\boldsymbol{r}^k_{\mathcal{B}^k \setminus \mathcal{B}^{k+1}}\right\|_2^2\right)^{1/2} \\
&= \sqrt{2}\left\|\boldsymbol{r}^k_{\mathcal{B}^k \cup \mathcal{B}^{k+1}}\right\|_2 = \sqrt{2}\left\|\boldsymbol{r}^k\right\|_2 .
\end{aligned}$$

Finally,

$$\left\|\boldsymbol{r}^{k+1}\right\|_2 \leq 2\left\|\boldsymbol{x}_{0,\mathcal{B}^{k+1}} - \boldsymbol{a}^{k+1}_{\mathcal{B}^{k+1}}\right\|_2 \leq \sqrt{8}\delta_{3S}\left\|\boldsymbol{r}^k\right\|_2 + \sqrt{1+\delta_{3S}}\left\|\boldsymbol{w}\right\|_2 .$$

□

5.4 Majorization Minimization (MM)

Consider an unconstrained optimization problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}).$$

Majorization minimization (MM) algorithm [5] constructs a series of surrogate functions that are easier to solve and guarantee the decrease of the original objective function. The algorithm is described below.

5: https://en.wikipedia.org/wiki/MM_algorithm

Algorithm 5: Majorization Minimization (MM)

1: **Initialization**: Choose an $\boldsymbol{x}^0$.
2: **while** $\ell = 1, 2, \cdots$ until stopping criteria are satisfied **do**
3: **Majorization step**: construct a surrogate function $f(\cdot|\boldsymbol{x}^l)$ s.t.

$$g(\boldsymbol{x}|\boldsymbol{x}^l) \geq f(\boldsymbol{x}) + c^l,$$

where $g(\cdot|\boldsymbol{x}^l) - f(\boldsymbol{x})$ is minimized at $\boldsymbol{x}^l$.
4: **Minimization step**:

$$\boldsymbol{x}^{l+1} \in \arg\min_{\boldsymbol{x}} g(\boldsymbol{x}|\boldsymbol{x}^l).$$

5: **end while**

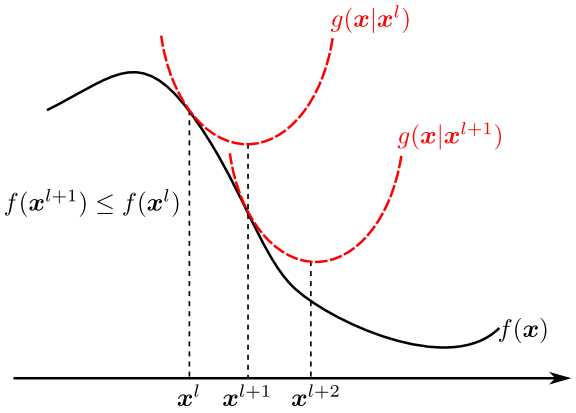


Figure 5.3: MM algorithm

Claim: the objective function decreases monotonically

$$f(\boldsymbol{x}^{l+1}) \overset{(a)}{\leq} g(\boldsymbol{x}^{l+1}|\boldsymbol{x}^l) - c^l \overset{(b)}{\leq} g(\boldsymbol{x}^l|\boldsymbol{x}^l) - c^l \overset{(c)}{\leq} f(\boldsymbol{x}^l),$$

where (a) and (c) follow from the majorization step ($g(\cdot|\boldsymbol{x}^l) - f(\boldsymbol{x})$ is minimized at $\boldsymbol{x}^l$), and (b) is due to the minimization step.

To understand MM algorithm, let us study the following example.

Example 5.4.1 Consider the unconstrained optimization problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}),$$

where

$$f(\boldsymbol{x}) = \delta\left(\|\boldsymbol{x}\|_0 \leq S\right) + \frac{\alpha}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \tag{5.8}$$

for some $\alpha > 0$.

As discussed in Section 5.1, this problem can be difficult to solve.

We construct a surrogate function as follows

$$\begin{aligned} g(\boldsymbol{x}|\boldsymbol{x}^l) &= f(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}^l\|_{\boldsymbol{M}}^2 \\ &= \delta\left(\|\boldsymbol{x}\|_0 \leq S\right) + \frac{\alpha}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}^l\|_{\boldsymbol{M}}^2, \end{aligned}$$

where

$$\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}^l\|_{\boldsymbol{M}}^2 := \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}^l\right)^{\mathsf{T}} \boldsymbol{M}\left(\boldsymbol{x} - \boldsymbol{x}^l\right),$$

and $\boldsymbol{M} \succeq 0$ is a properly chosen matrix.

In particular, choose a γ s.t. $0 < \gamma < 1/\sigma_{\max}^2(\boldsymbol{A})$ and define

$$\boldsymbol{M} := \frac{1}{\gamma}\boldsymbol{I} - \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}.$$

It is straightforward to verify that

▸

$$\boldsymbol{M} \succeq \sigma_{\max}^2(\boldsymbol{A})\boldsymbol{I} - \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} \succeq 0.$$

- $$g(\boldsymbol{x}|\boldsymbol{x}^l) = f(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}^l\|_{\boldsymbol{M}}^2 \geq f(\boldsymbol{x}),\ \forall \boldsymbol{x}$$
- $$g(\boldsymbol{x}^l|\boldsymbol{x}^l) = f(\boldsymbol{x}^l)$$

  and $g(\boldsymbol{x}|\boldsymbol{x}^l) - f(\boldsymbol{x})$ is minimized at $\boldsymbol{x}^l$.

With the above particular choice of $\boldsymbol{M} \geq 0$, it holds that

$$\begin{aligned}
g(\boldsymbol{x}|\boldsymbol{x}^l) &= \delta\left(\|\boldsymbol{x}\|_0 \leq S\right) + \frac{\alpha}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}^l\|_{\boldsymbol{M}}^2 \\
&= \delta\left(\|\boldsymbol{x}\|_0 \leq S\right) \\
&\quad + \frac{1}{2}\boldsymbol{x}^\mathsf{T}\boldsymbol{A}^\mathsf{T}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}^\mathsf{T}\boldsymbol{A}\boldsymbol{x} + \frac{1}{2}\boldsymbol{y}^\mathsf{T}\boldsymbol{y} \\
&\quad + \frac{1}{2}\boldsymbol{x}^\mathsf{T}\boldsymbol{M}\boldsymbol{x} - \boldsymbol{x}^{l,\mathsf{T}}\boldsymbol{M}\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^{l,\mathsf{T}}\boldsymbol{M}\boldsymbol{x}^l \\
&= \delta\left(\|\boldsymbol{x}\|_0 \leq S\right) + \frac{1}{2\gamma}\left\|\boldsymbol{x} - \gamma\left(\boldsymbol{A}^\mathsf{T}\boldsymbol{y} + \boldsymbol{M}\boldsymbol{x}^l\right)\right\|_2^2 - c \\
&= \delta\left(\|\boldsymbol{x}\|_0 \leq S\right) + \frac{1}{2\gamma}\left\|\boldsymbol{x} - \left(\boldsymbol{x}^l + \gamma\boldsymbol{A}^\mathsf{T}\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^l\right)\right)\right\|_2^2 - c,
\end{aligned}$$

where c is a constant independent of $\boldsymbol{x}$ and hence does not affect the minimization step. Now it is clear that this choice of $\boldsymbol{M}$ admits a closed form solution of the minimization step (using the proximal operator in Example 5.1.3). MM algorithm can solve the nonconvex and non-smooth (unconstrained) optimization problem in (5.8).

Example 5.4.2 Consider the unconstrained optimization problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}),$$

where

$$f(\boldsymbol{x}) = \|\boldsymbol{x}\|_0 + \frac{\alpha}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2$$

for some $\alpha > 0$.

5.5 Appendix

Lp Norm

Definition 5.5.1 (ℓ_p-norm) *For a real number $p \geq 1$, the ℓ_p-norm of $\boldsymbol{x} \in \mathbb{R}^n$ is given by*

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{\frac{1}{p}}.$$

Example 5.5.1

- ℓ_1-norm (Manhattan distance): $\|\boldsymbol{x}\|_1 = \sum |x_i|$.

- ℓ_2-norm (Euclidean norm): $\|\boldsymbol{x}\| = \sqrt{\sum x_i^2}$.
- ℓ_∞-norm: $\|\boldsymbol{x}\|_\infty = \max(|x_1|, \cdots, |x_n|)$.

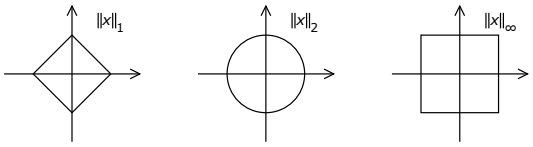


Figure 5.4: ℓ_p-norms

The Cauchy-Schwartz Inequality

Theorem 5.5.1 (The Hölder's inequality)

Let $p, q \in [1, \infty]$ *with* $1/p + 1/q = 1$.

For all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, *it holds that*

$$\sum_{i=1}^{n} |x_i \cdot y_i| \leq \|\boldsymbol{x}\|_p \left\|\boldsymbol{y}\right\|_q = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p} \cdot \left(\sum_{i=1}^{n} \left|y_i\right|^q\right)^{1/q}.$$

The equality holds iff $|x|^p$ *and* $\left|y\right|^q$ *are linear dependent, i.e.,* $\alpha |x_i|^p = \beta \left|y_i\right|^q$, $\forall i$. [6]

6: Hölder's inequality is a fundamental inequality for real analysis.

(Proof is omitted.)

Theorem 5.5.2 (The Cauchy-Schwartz Inequality)

A special case of the Hölder's inequality is when $p = q = 2$.

$$\sum_{i=1}^{n} |x_i \cdot y_i| \leq \|\boldsymbol{x}\|_2 \cdot \left\|\boldsymbol{y}\right\|_2.$$

In particular, for all $\boldsymbol{x} \in \mathbb{R}^n$,

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i| \leq \sqrt{n} \|\boldsymbol{x}\|_2,$$

where the equality holds iff $|x_i| = \left|x_j\right|$.

Convex Optimization: Introduction

6

6.1 Convexity 41
6.1.1 Definitions 41
6.1.2 Conditions of Convexity . 42
6.2 Unconstrained Convex Optimization 44
6.3 Line Search Methods . . . 45

6.1 Convexity

6.1.1 Definitions

Definition 6.1.1 *A* convex combination *is a linear combination of points where all coefficients are non-negative and sum to 1.*

More specifically, let $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_\ell \in \mathbb{R}^n$*. A convex combination of these points is of the form*

$$\sum_{i=1}^{\ell} \lambda_i \boldsymbol{x}_i,$$

where the real coefficients λ_i *satisfy* $\lambda_i \geq 0$ *and* $\sum_{i=1}^{n} \lambda_i = 1$*.*

Definition 6.1.2 *A set* $\mathcal{X}$ *is a* convex set *if and only if the convex combination of any two points in the set belongs to the set.*

That is, $\mathcal{X} \subseteq \mathbb{R}^n$ *is convex iff*

$$\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X},\ \forall \lambda \in [0,1],\ \lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2 \in \mathcal{X}.$$

Example 6.1.1 (Convex Sets)

- A *hyperplane*:

$$\mathcal{H} = \left\{\boldsymbol{x} :\ \boldsymbol{a}^T \boldsymbol{x} = b\right\}$$

- A *halfspace*:

$$\mathcal{H}_+ = \left\{\boldsymbol{x} :\ \boldsymbol{a}^T \boldsymbol{x} \leq b\right\}$$

- A *polyhedron*:

$$\mathcal{P} = \left\{\boldsymbol{x} :\ \boldsymbol{a}_j^T \boldsymbol{x} \leq b_j,\ j = 1, \cdots, m\right\}$$

- Intersections of convex sets are convex.

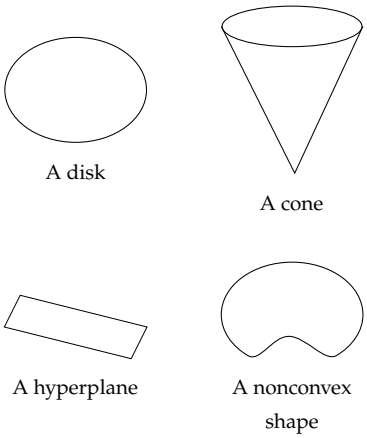


Figure 6.1: Convex and nonconvex sets

Definition 6.1.3 *The* domain *of a function* $f : \mathbb{R}^n \to \mathbb{R}$ *is defined as the set of the points where the function* f *is finite, i.e.,*

$$dom f = \left\{\boldsymbol{x} \in \mathbb{R}^n :\ \left|f(\boldsymbol{x})\right| < \infty\right\}.$$

For example, $\operatorname{dom} \log x = \mathbb{R}^+$.

Definition 6.1.4 (Convex functions) *A function* $f : \mathbb{R}^n \to \mathbb{R}$ *is* convex *if for any* $\boldsymbol{x}_1, \boldsymbol{x}_2 \in dom f \subseteq \mathbb{R}^n$, $\lambda \in [0,1]$*, it holds*

$$\lambda f(\boldsymbol{x}_1) + (1-\lambda) f(\boldsymbol{x}_2) \geq f(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2).$$

This definition implies that dom$f$ is convex. However, for simplicity we assume that dom $f = \mathbb{R}^n$ in this section.

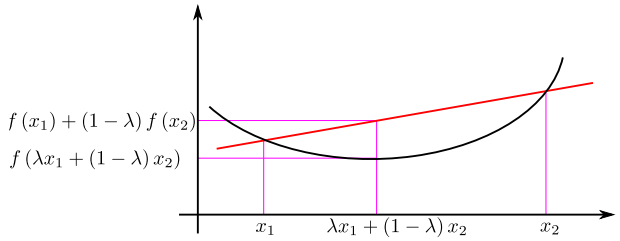


Figure 6.2: Convex functions

6.1.2 Conditions of Convexity

A useful mathematical tool to study conditions of convexity is Taylor's theorem.

Theorem 6.1.1 (Taylor's Theorem) *If f is continuously differentiable, then*

$$f(\boldsymbol{x}+\boldsymbol{p}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x}+t\boldsymbol{p})^{\mathsf{T}}\boldsymbol{p},$$

for some $t \in (0,1)$.

Furthermore, if f is twice continuously differentiable, we have

$$\nabla f(\boldsymbol{x}+\boldsymbol{p}) = \nabla f(\boldsymbol{x}) + \int_0^1 \nabla^2 f(\boldsymbol{x}+t\boldsymbol{p})\boldsymbol{p}\,dt,$$
$$f(\boldsymbol{x}+\boldsymbol{p}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{p} + \frac{1}{2}\boldsymbol{p}^{\mathsf{T}}\nabla^2 f(\boldsymbol{x}+t\boldsymbol{p})\boldsymbol{p}$$

for some $t \in (0,1)$.

With Taylor's theorem, we are able to state and prove the first-order and second-order conditions of convexity.

Theorem 6.1.2 (First-Order Condition of Convexity) *Suppose a function $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable. Then it is convex if and only if for all $\boldsymbol{x}, \boldsymbol{y} \in dom f$, it holds*

$$f\left(\boldsymbol{y}\right) \geq f\left(\boldsymbol{x}\right) + \nabla f\left(\boldsymbol{x}\right)^{\mathsf{T}}\left(\boldsymbol{y}-\boldsymbol{x}\right). \tag{6.1}$$

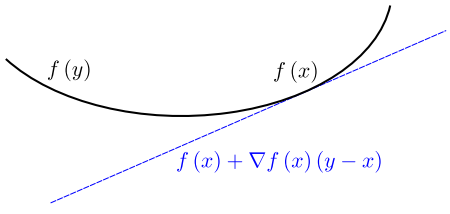


Figure 6.3: First-order condition of convexity

Proof. Necessity:

Assume first that f is convex and $\boldsymbol{x}, \boldsymbol{y} \in \text{dom}\left(f\right)$. Since dom $\left(f\right)$ is convex, $\boldsymbol{x} + t\left(\boldsymbol{y}-\boldsymbol{x}\right) \in \text{dom}\left(f\right)$ for all $0 < t \leq 1$. By convexity of f,

$$f\left(\boldsymbol{x} + t\left(\boldsymbol{y}-\boldsymbol{x}\right)\right) \leq (1-t)\, f\left(\boldsymbol{x}\right) + t f\left(\boldsymbol{y}\right).$$

Divide both sides by t. It holds

$$f\left(\boldsymbol{y}\right) \geq f\left(\boldsymbol{x}\right) + \frac{f\left(\boldsymbol{x} + t\left(\boldsymbol{y} - \boldsymbol{x}\right)\right) - f\left(\boldsymbol{x}\right)}{t}.$$

Take the limit as $t \to 0$ yields (6.1).

Sufficiency:

To show the other direction (sufficiency), assume that (6.1) holds. Choose any $\boldsymbol{x} \neq \boldsymbol{y}$ and $\lambda \in [0,1]$. Let $\boldsymbol{z} = \lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y}$. Applying (6.1) twice yields

$$\begin{aligned} f\left(\boldsymbol{x}\right) - f\left(\boldsymbol{z}\right) &\geq \nabla f\left(\boldsymbol{z}\right)^T \left(\boldsymbol{x} - \boldsymbol{z}\right), \\ f\left(\boldsymbol{y}\right) - f\left(\boldsymbol{z}\right) &\geq \nabla f\left(\boldsymbol{z}\right)^T \left(\boldsymbol{y} - \boldsymbol{z}\right). \end{aligned}$$

Multiply the first inequality by λ and the second by $1-\lambda$, and then add them together. It holds

$$\begin{aligned} &\lambda f\left(\boldsymbol{x}\right) + (1-\lambda) f\left(\boldsymbol{y}\right) - f\left(\boldsymbol{z}\right) \\ &\quad \geq \nabla f\left(\boldsymbol{z}\right)^T \left(\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y} - \boldsymbol{z}\right). \end{aligned}$$

By the definition of $\boldsymbol{z}$, the left side of the inequality is zero. Hence,

$$\lambda f\left(\boldsymbol{x}\right) + (1-\lambda) f\left(\boldsymbol{y}\right) \geq f\left(\boldsymbol{z}\right),$$

which proves that f is convex. □

Theorem 6.1.3 (Second-Order Condition of Convexity) *Suppose that f is twice differentiable. Then f is convex iff the Hessian of f is positive semidefinite (PSD), i.e., $\nabla^2 f(\boldsymbol{x}) \geq 0$.*

Proof. The proof relies on Taylor series:

$$f(\boldsymbol{x} + \boldsymbol{v}) = f(\boldsymbol{x}) + \nabla f(x)^T \boldsymbol{v} + \frac{1}{2}\boldsymbol{v}^T \nabla^2 f(\boldsymbol{x} + \theta \boldsymbol{v})\boldsymbol{v}$$

for some $\theta \in [0,1]$.

- If $\nabla^2 f \geq 0$, then $f(\boldsymbol{x}+\boldsymbol{v}) \geq f(\boldsymbol{x}) + \nabla f(x)^T \boldsymbol{v}$. By first-order condition for convexity, f is convex.
- If f is convex, then $f(\boldsymbol{x} + \boldsymbol{v}) \geq f(\boldsymbol{x}) + \nabla f(x)^T \boldsymbol{v}$ for all $\boldsymbol{v}$. Hence, $\boldsymbol{v}^T \nabla^2 f(\boldsymbol{x} + \theta \boldsymbol{v})\boldsymbol{v} \geq 0$. Let $\|\boldsymbol{v}\|_2 \to 0$. Then $\nabla^2 f(\boldsymbol{x}) \geq 0$.

□

Example 6.1.2 [1]

1. $f(x) = \log(x)$ is concave on its domain $x \in \mathbb{R}^+$.
2. $f(x) = x\log(x)$ is convex on its domain $x \in \mathbb{R}^+$.
3. Affine functions $f(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$ are both convex and concave.
4. Quadratic functions $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{x} + \boldsymbol{c}$ are convex if and only if $\boldsymbol{A} \geq 0$.
5. $f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2$ is convex.

1: In this list, the first five examples can be verified using second-order condition of convexity.

6. $f(\boldsymbol{x}) = \|\boldsymbol{x}\|_1 = \sum_i |x_i|$ is convex.
7. If $f(\boldsymbol{x})$ is convex, then $g(\boldsymbol{x}) := f(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b})$ is convex.
   - $f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x}\|_1$ is convex.
8. For large dimensional data processing, the frequently used function

$$\frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1,$$

   where $\lambda > 0$ is convex.

6.2 Unconstrained Convex Optimization

An unconstrained optimization problem is of the form

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}).$$

Definition 6.2.1 (global and local minimizer) *A point $\boldsymbol{x}^\star$ is a global minimizer if $f(\boldsymbol{x}^\star) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x}$.*

A point $\boldsymbol{x}^\star$ is a local minimizer if there is a neighbourhood $\mathcal{N}$ of $\boldsymbol{x}^\star$ such that $f(\boldsymbol{x}^\star) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{N}$. [2]

2: A neighbourhood of $\boldsymbol{x}^\star$ is simply an open set that contains $\boldsymbol{x}^\star$, for example, $\mathcal{B}(\boldsymbol{x}^\star, \epsilon) := \{x : \|x - \boldsymbol{x}^\star\|_2 < \epsilon\}$.

We are particularly interested in the case that f is convex due to some nice properties stated below.

Theorem 6.2.1 *Suppose that a feasible point $\boldsymbol{x}$ is locally optimal for a convex optimization problem. Then it is also globally optimal.*

Proof. Suppose that $\boldsymbol{x}$ is locally optimal but not globally optimal. Then there must exist a $\boldsymbol{y} \neq \boldsymbol{x}$ such that $f(\boldsymbol{y}) < f(\boldsymbol{x})$.

Consider a point $\boldsymbol{z}$ on the line segment between $\boldsymbol{x}$ and $\boldsymbol{y}$, i.e.,

$$\boldsymbol{z} = (1 - \lambda)\boldsymbol{x} + \lambda\boldsymbol{y}, \ \lambda \in (0, 1).$$

It is clear that

$$f(\boldsymbol{z}) \leq (1 - \lambda) f(\boldsymbol{x}) + \lambda f(\boldsymbol{y}) < f(\boldsymbol{x}),$$

where the first inequality follows from convexity of f, and the second (strict) inequality holds because $f(\boldsymbol{y}) < f(\boldsymbol{x})$. This contradicts with that $\boldsymbol{x}$ is locally optimal and proves the global optimality of $\boldsymbol{x}$. □

Theorem 6.2.2 (A Global Optimality Criterion) *Suppose that the objective function f in a convex optimization problem is differentiable, i.e.,*

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\mathsf{T}(\boldsymbol{y} - \boldsymbol{x}), \ \forall \boldsymbol{x}, \boldsymbol{y} \in dom(f).$$

Then an $\boldsymbol{x} \in dom\left(f\right)$ is optimal if and only if

$$\nabla f\left(\boldsymbol{x}\right)^{T}\left(\boldsymbol{y}-\boldsymbol{x}\right) \geq 0,\ \forall \boldsymbol{y} \in dom\left(f\right).$$

Or equivalently,

$$\nabla f(\boldsymbol{x}) = \mathbf{0}$$

when f is closed.

Proof. Sufficiency: Suppose the inequality holds. Then for all $\boldsymbol{y}$,

$$\begin{aligned} f\left(\boldsymbol{y}\right) &\geq f\left(\boldsymbol{x}\right) + \nabla f\left(\boldsymbol{x}\right)^{\mathsf{T}}\left(\boldsymbol{y}-\boldsymbol{x}\right) \\ &\geq f\left(\boldsymbol{x}\right). \end{aligned}$$

Hence, $\boldsymbol{x}$ is globally optimal.

Necessity: suppose $\boldsymbol{x}$ is optimal, but the inequality does not hold, i.e., for some $\boldsymbol{y}$ we have

$$\nabla f\left(\boldsymbol{x}\right)^{\mathsf{T}}\left(\boldsymbol{y}-\boldsymbol{x}\right) < 0.$$

Consider the point $\boldsymbol{z}\left(t\right) = t\boldsymbol{y} + \left(1-t\right)\boldsymbol{x},\ t \in [0,1]$. Then

$$\begin{aligned} \frac{d}{dt} f\left(\boldsymbol{z}\left(t\right)\right)\big|_{t=0} &= \nabla f\left(\boldsymbol{z}\left(0\right)\right)^{\mathsf{T}} \frac{d}{dt}\boldsymbol{z}\left(t\right)|_{t=0} \\ &= \nabla f\left(\boldsymbol{x}\right)^{\mathsf{T}}\left(\boldsymbol{y}-\boldsymbol{x}\right) < 0. \end{aligned}$$

This implies that for small $t > 0$, we have $f\left(\boldsymbol{z}\left(t\right)\right) < f\left(\boldsymbol{x}\right)$, which contradicts the optimality of $\boldsymbol{x}$. The necessity is therefore proved. □

Remark 6.2.1 The global optimal solution $\boldsymbol{x}$ may not be unique.

6.3 Line Search Methods

In each iteration of linear search methods, the algorithm chooses a direction $\boldsymbol{p}^{l}$ and searches along this direction for a lower function value. That is,

$$\boldsymbol{x}^{l+1} = \boldsymbol{x}^{l} + \alpha^{l}\boldsymbol{p}^{l},$$

where the search direction $\boldsymbol{p}^{l}$ and step size $\alpha^{l} > 0$ are appropriately chosen so that $f(\boldsymbol{x}^{l} + \alpha^{l}\boldsymbol{p}^{l}) < f(\boldsymbol{x}^{l})$.

Search Direction

Most line search methods require a search direction $\boldsymbol{p}^{l}$ to be a descent direction:

$$\boldsymbol{p}^{l,\mathsf{T}}\nabla f(\boldsymbol{x}^{l}) < 0.$$

Based on Taylor's theorem, it is straightforward to show that the function f will be reduced along this direction as long as the step size α_{k} is sufficiently small.

Typical choices of the search direction $\boldsymbol{p}^l$ are of the form

$$\boldsymbol{p}^l = -\boldsymbol{B}^l \nabla f(\boldsymbol{x}^l),$$

where $\boldsymbol{B}^l \succeq 0$ so that

$$\boldsymbol{p}^{l,\mathsf{T}} \nabla f(\boldsymbol{x}^l) = -\nabla f(\boldsymbol{x}^l)^\mathsf{T} \boldsymbol{B}^l \nabla f(\boldsymbol{x}^l) < 0.$$

Examples include

1. Steepest descent method: $\boldsymbol{B}^l = \boldsymbol{I}$.
2. Newton's method: $\boldsymbol{B}^l = \nabla^2 f(\boldsymbol{x}^l)$.
3. Quasi-Newton method: $\boldsymbol{B}^l$ is an approximation to the exact Hessian $\nabla^2 f(\boldsymbol{x}^l)$.

Step Size

Exact Line Search

The step size α^l is optimized at each iteration:

$$\alpha^l = \arg \min_{\alpha > 0} f(\boldsymbol{x}^l + \alpha \boldsymbol{p}^l).$$

In practice, it is often not possible to do this minimization exactly.

Backtracking Line Search

A more practical strategy is to choose an $\alpha^l > 0$ which gives *sufficient decrease* in the objective function f. For example, the popular *Armijo Condition* requires that

$$f(\boldsymbol{x}^l + \alpha^l \boldsymbol{p}^l) \leq f(\boldsymbol{x}^l) + c\alpha^l \boldsymbol{p}^{l,\mathsf{T}} \nabla f(\boldsymbol{x}^l),$$

for some constant $c \in (0, 1)$. In practice, c is chosen to be small, say $c = 10^{-4}$.

Armijo condition itself does not prevent tiny step sizes. To ensure reasonable progress of the algorithm, the *curvature condition* requires α^l to satisfy

$$\boldsymbol{p}^{l,\mathsf{T}} \nabla f(\boldsymbol{x}^l + \alpha^l \boldsymbol{p}^l) \geq c' \boldsymbol{p}^{l,\mathsf{T}} \nabla f(\boldsymbol{x}^l)$$

for some constant $c' \in (c, 1)$.

The combination of Armijo and curvature conditions is called *Wolfe Conditions*:

$$\begin{aligned} f(\boldsymbol{x}^l + \alpha^l \boldsymbol{p}^l) &\leq f(\boldsymbol{x}^l) + c\alpha^l \boldsymbol{p}^{l,\mathsf{T}} \nabla f(\boldsymbol{x}^l), \\ \boldsymbol{p}^{l,\mathsf{T}} \nabla f(\boldsymbol{x}^l + \alpha^l \boldsymbol{p}^l) &\geq c' \boldsymbol{p}^{l,\mathsf{T}} \nabla f(\boldsymbol{x}^l) \end{aligned}$$

with $0 < c < c' < 1$.

Sparse Inverse Problems: ℓ_1-minimization

7

7.1 Convex Relaxation 47
7.2 LASSO Solvers 50
7.2.1 Soft Thresholding Function 50
7.2.2 Cyclic Coordinate Descent 51
7.2.3 Iterative Shrinkage Thresholding Algorithm (ISTA) . 51
7.2.4 Majorization Minimization 52
7.2.5 Least Angle Regression (LAR) 52

7.1 Convex Relaxation

One popular approach to solve the sparse linear inverse problem is to replace the nonconvex objective functions mentioned in Chapter 5 with a convex one (convex relaxation).

Lp-norm

Definition 7.1.1 *Given a vector space $\mathcal{V}$ over the field $\mathbb{F}$ of complex (real) numbers, a norm on $\mathcal{V}$ is a function $p: \mathcal{V} \to \mathbb{R}$ with the following properties.*

For all $a \in \mathbb{F}$ and all $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{V}$,

1. $p(a\boldsymbol{v}) = |a|\, p(\boldsymbol{v})$, *(absolute scalability)*
2. $p(\boldsymbol{u} + \boldsymbol{v}) \leq p(\boldsymbol{u}) + p(\boldsymbol{v})$, *(triangle inequality)*
3. *if $p(\boldsymbol{v}) = 0$ then $\boldsymbol{v}$ is the zero vector. (separates points)*

Positivity follows from this definition.

Proof. By the first axiom, $p(\mathbf{0}) = 0$ and $p(-\boldsymbol{v}) = p(\boldsymbol{v})$. Then by triangle inequality,

$$0 \leq p(\boldsymbol{v}) + p(-\boldsymbol{v}) = 2p(\boldsymbol{v}) \implies 0 \leq p(\boldsymbol{v}).$$

□

Lemma 7.1.1 *A norm is a convex function.*

Proof. For any given $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ and $\lambda \in [0, 1]$, it holds that

$$\begin{aligned} \|\lambda\boldsymbol{u} + (1-\lambda)\boldsymbol{v}\| &\leq \|\lambda\boldsymbol{u}\| + \|(1-\lambda)\boldsymbol{v}\| \\ &= \lambda\|\boldsymbol{u}\| + (1-\lambda)\|\boldsymbol{v}\|, \end{aligned}$$

where we have used the triangle inequality and the absolute scalability. This establishes the convexity of the norm. □

Recall Definition 5.5.1 of ℓ_p-norm. For $p \geq 1$,

$$\|\boldsymbol{x}\|_p = \left(\sum_i |x_i|^p\right)^{1/p}.$$

In this definition, ℓ_p-norm is a proper norm iff $p \geq 1$.

The following picture draws the unit ℓ_p-ball.

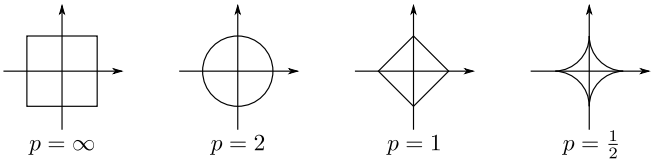


Figure 7.1: Unit ℓ_p ball

LASSO Problem

The most well-known convex relaxation for sparse linear inverse problem is LASSO (least absolute shrinkage and selection operator):

$$\min_{\boldsymbol{x}} \frac{1}{2} \left\| \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} \right\|_2^2 + \lambda \left\| \boldsymbol{x} \right\|_1 . \tag{7.1}$$

The first term is for data fidelity and the second one is an ℓ_1 regularization.

Figure 7.2 gives the geometric intuition why ℓ_1-norm promotes sparsity while ℓ_2-norm does not. Note that the set of feasible solutions of $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ is given by $\mathcal{X} = \boldsymbol{x}_0 + \mathcal{N}ull(\boldsymbol{A})$.

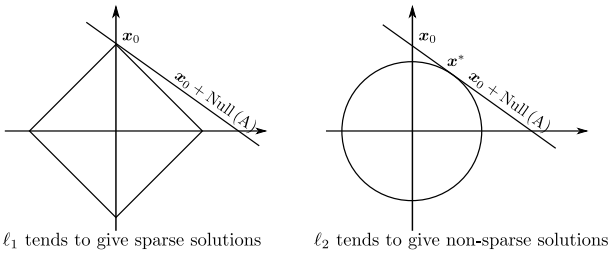


Figure 7.2: ℓ_1-norm promotes sparse solutions

Difficulty in Solving LASSO

The main difficulty in solving LASSO is that the objective function is not differentiable! Gradient descent method cannot be directly applied.

One naive approach is to define the so called subgradient and apply a subgradient line search method.

Definition 7.1.2 *Let $f : \mathcal{U} \to \mathbb{R}$ be a convex function defined on a convex open set $\mathcal{U} \subset \mathbb{R}^n$. A vector $\boldsymbol{v} \in \mathbb{R}^n$ is called a subgradient at a point $\boldsymbol{x} \in \mathcal{U}$ if*

$$f\left(\boldsymbol{y}\right) - f\left(\boldsymbol{x}\right) \geq \boldsymbol{v}^{\mathsf{T}} \left(\boldsymbol{y} - \boldsymbol{x}\right), \ \forall \boldsymbol{y} \in \mathcal{U}.$$

The set of all subgradients at $\boldsymbol{x}$ is called the subdifferential at $\boldsymbol{x}$ and is denoted $\partial f\left(\boldsymbol{x}\right)$.

Remark 7.1.1 If f is convex and its subdifferential at $\boldsymbol{x}$ contains exactly one subgradient, then f is differentiable at $\boldsymbol{x}$.

Example 7.1.1 Let $f(x) = |x|$. Then the subdifferential of f is given by

$$\partial f(x) = \partial |x| = \begin{cases} 1 & \text{if } x > 0, \\ [-1, 1] & \text{if } x = 0, \\ -1 & \text{if } x < 0. \end{cases}$$

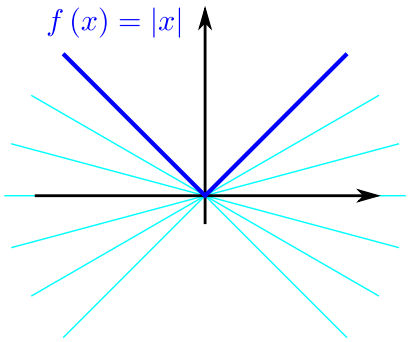


Figure 7.3: Subdifferential of $|x|$

With the definition of subgradient, one can implement a subgradient line search method. However,

- Subgradient line search does not necessarily lead to a decrease of the objective function.
- Subgradient line search does not necessarily converge to a global minimizer.

Example 7.1.2 Let $\boldsymbol{x} \in \mathbb{R}^2$ and

$$f(\boldsymbol{x}) = 3|x_1| + |x_2|.$$

Consider the point $\boldsymbol{x}^0 = [0, 1]^\mathsf{T}$. Show that

- $\boldsymbol{g}^1 = [0, 1]^\mathsf{T} \in \partial f(\boldsymbol{x})$.
- $\boldsymbol{g}^2 = [3, 1]^\mathsf{T} \in \partial f(\boldsymbol{x})$.

Calculate

$$\lim_{\tau > 0,\, \tau \to 0} \frac{f(\boldsymbol{x}^0 - \tau \boldsymbol{g}) - f(\boldsymbol{x}^0)}{\tau}$$

with $\boldsymbol{g} = \boldsymbol{g}^1$ and $\boldsymbol{g} = \boldsymbol{g}^2$ respectively. Comment on whether $-\boldsymbol{g}$ is always a descent direction or not.

Example 7.1.3 (Wolfe's Example) Let $\boldsymbol{x} \in \mathbb{R}^2$ and

$$f(\boldsymbol{x}) = \begin{cases} 5(9x_1^2 + 16x_2^2)^{1/2} & \text{if } x_1 > |x_2|, \\ 9x_1 + 16|x_2| & \text{if } x_1 \leq |x_2|. \end{cases}$$

Show that

$$\boldsymbol{x}^{\text{nd}} = [0, 0]^\mathsf{T}$$

is a non-differentiable point.

Suppose that $\boldsymbol{x}^0 = [16/9, 1]^\mathsf{T}$. Consider exact line search where

$$\boldsymbol{x}^{l+1} = \boldsymbol{x}^l - t^l \nabla f(\boldsymbol{x}^l),$$

where

$$t^l = \arg\min_t f(\boldsymbol{x}^l - t\nabla f(\boldsymbol{x}^l)).$$

Show that

1. $\boldsymbol{x}^l$ are all differentiable points;
2. $\boldsymbol{x}^l \to [0,0]^\mathsf{T}$ as $l \to \infty$, and the point $[0,0]^\mathsf{T}$ is not a global minimizer.

7.2 LASSO Solvers

7.2.1 Soft Thresholding Function

Proximal operator of ℓ_1-norm plays an essential role in developing practical algorithms to solve LASSO problems.

We first study the simple case — the scalar case.

Let

$$x^\star = \mathrm{Prox}_{\gamma|\cdot|}(y) = \arg\min_x |x| + \frac{1}{2\gamma}(x-y)^2.$$

It is clear that the objective function

$$f_\gamma(x) := |x| + \frac{1}{2\gamma}(x-y)^2$$

is convex. Set its subgradient at the global minimizer $x^\star$ to zero. One has

$$\partial_x f_\gamma(x^\star) = \partial_x |x^\star| + \frac{1}{\gamma}(x-y).$$

Hence,

$$\begin{aligned} x^\star &= y - \gamma \partial_x |x^\star| \\ &= \begin{cases} y-\gamma & \text{if } y \geq \gamma, \\ 0 & \text{if } -\gamma < y < \gamma, \\ y+\gamma & \text{if } y \leq -gamma, \end{cases} \\ &=: \eta(y;\gamma) = \eta_\gamma(y) \\ &= \mathrm{sign}(y)(|y|-\gamma)_+, \end{aligned} \tag{7.2}$$

where $(z)_+ = \max(z, 0)$.

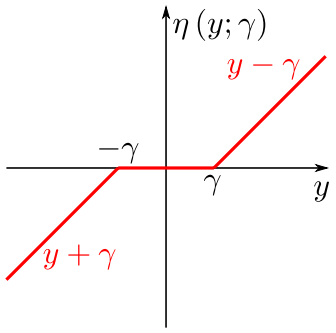


Figure 7.4: Soft thresholding function

The extension to vector case is straightforward. Let $\boldsymbol{x}^\star = \text{Prox}_{\gamma\|\cdot\|}(\boldsymbol{y})$. Then $x_i^\star = \eta(y_i;\gamma)$.

This closed form solution of the ℓ_1-norm proximal operator is called soft thresholding function.

7.2.2 Cyclic Coordinate Descent

Consider the LASSO problem

$$\min_{\boldsymbol{x}} \frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \lambda\left\|\boldsymbol{x}\right\|_1.$$

At each step of cyclic coordinate descent, we optimize the objective function with respect to x_i for some i by fixing all x_j, $j \neq i$. The objective function can be then written as

$$\begin{aligned}
&\frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \lambda\left\|\boldsymbol{x}\right\|_1 \\
&= \frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{a}_i x_i - \sum_{j\neq i}\boldsymbol{a}_j x_j\right\|_2^2 + \lambda|x_i| + \lambda\sum_{j\neq i}|x_j| \\
&= \frac{1}{2}\left\|\boldsymbol{r}_i - \boldsymbol{a}_i x_i\right\|_2^2 + \lambda|x_i| + c.
\end{aligned}$$

The global minimizer $x_i^\star$ is given by

$$x_i^\star = \eta(\boldsymbol{a}_i^\mathsf{T}\boldsymbol{r}_i;\lambda),$$

where

$$\begin{aligned}
\boldsymbol{r}_i &:= \boldsymbol{y} - \sum_{j\neq i}\boldsymbol{a}_j\hat{x}_j \\
&= \boldsymbol{y} - \sum_{j}\boldsymbol{a}_j\hat{x}_j + \boldsymbol{a}_i\hat{x}_i.
\end{aligned}$$

In practice, cyclic coordinate descent is slow as one needs to cyclically update all $\hat{x}_i$, $i = 1, 2, \cdots, N$, many times.

7.2.3 Iterative Shrinkage Thresholding Algorithm (ISTA)

The key idea of ISTA is a Taylor approximation to allow the direct application of proximal operator.

Let $f(\boldsymbol{x}) = \frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|_2^2$. Approximate the objective function locally by

$$\begin{aligned} & f(\boldsymbol{x}) + \lambda \left\|\boldsymbol{x}\right\|_1 \\ & \approx f(\boldsymbol{x}^l) + \left\langle \boldsymbol{x} - \boldsymbol{x}^l, \nabla f(\boldsymbol{x}^l) \right\rangle + \frac{1}{2\tau^l}\left\|\boldsymbol{x} - \boldsymbol{x}^l\right\|_2^2 + \lambda \left\|\boldsymbol{x}\right\|_1 \\ & = \frac{1}{2\tau^l}\left\|\boldsymbol{x} - \left(\boldsymbol{x}^l - \tau^l \nabla f(\boldsymbol{x}^l)\right)\right\|_2^2 + \lambda \left\|\boldsymbol{x}\right\|_1 + c. \end{aligned}$$

Note that

$$\nabla f(\boldsymbol{x}^l) = -\boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^l).$$

The ISTA iterations are given as

$$\boldsymbol{x}^{l+1} = \eta\left(\boldsymbol{x}^l + \tau^l \boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^l); \lambda\tau^l\right).$$

In practice, the step size τ^l is fixed as $\tau^l = \tau \in (0, 2/\sigma_{\max}(\boldsymbol{A}^\mathsf{T}\boldsymbol{A}))$.

7.2.4 Majorization Minimization

Apply majorization minimization (Section 5.4) to the LASSO problem. In each iteration, one has objective function

$$\begin{aligned} & \lambda \left\|\boldsymbol{x}\right\|_1 + \frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \frac{1}{2}\left\|\boldsymbol{x} - \boldsymbol{x}^l\right\|_M^2 \\ & = \lambda \left\|\boldsymbol{x}\right\|_1 + \frac{1}{2\gamma}\left\|\boldsymbol{x} - \left(\boldsymbol{x}^l + \gamma \boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^l)\right)\right\|_2^2 + c, \end{aligned}$$

where $\boldsymbol{M} = \frac{1}{\gamma}\boldsymbol{I} - \boldsymbol{A}^\mathsf{T}\boldsymbol{A} \succeq 0$ and $\gamma > 1/\sigma_{\max}(\boldsymbol{A}^\mathsf{T}\boldsymbol{A})$. Hence in each iteration of majorization minimization, proximal operator can be directly used for the update:

$$\begin{aligned} \boldsymbol{x}^{l+1} &= \mathrm{Prox}_{\lambda\gamma\|\cdot\|_1}\left(\boldsymbol{x}^l + \gamma \boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^l)\right) \\ &= \eta\left(\boldsymbol{x}^l + \gamma \boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^l); \lambda\gamma\right). \end{aligned}$$

7.2.5 Least Angle Regression (LAR)

Observation: Different $\lambda > 0$ results in different sparsity in the optimal solution

$$\boldsymbol{x}_\lambda := \arg\min_{\boldsymbol{x}} \frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \lambda \left\|\boldsymbol{x}\right\|_1.$$

- $\lambda = 0$:

$$\begin{aligned} \boldsymbol{x}_\lambda &= \arg\min_{\boldsymbol{x}} \left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|_2^2 \\ &= \boldsymbol{A}^\dagger \boldsymbol{y} \end{aligned}$$

  is not sparse.

- $\lambda \to \infty$:

$$\begin{aligned} \boldsymbol{x}_\lambda &= \arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 \\ &= \mathbf{0}. \end{aligned}$$

Least angle regression (LAR): Find a λ such that the optimal $\boldsymbol{x}$ has a specific sparsity.

The derivation of LAR is based on the optimal condition:

$$\boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_\lambda) = \lambda \text{sign}(\boldsymbol{x}_\lambda).$$

Let $\mathcal{I} := \text{supp}(\boldsymbol{x}_\lambda)$ and $\mathcal{I}^c := \{1, \cdots, N\} \backslash \text{supp}(\boldsymbol{x}_\lambda)$. It holds that

$$\begin{cases} \boldsymbol{x}_{\lambda,\mathcal{I}} = (\boldsymbol{A}_\mathcal{I}^T \boldsymbol{A}_\mathcal{I})^{-1} \left(\boldsymbol{A}_\mathcal{I}^T \boldsymbol{y} - \lambda \text{sign}(\boldsymbol{x}_{\lambda,\mathcal{I}})\right), \\ \boldsymbol{x}_{\lambda,\mathcal{I}^c} = \mathbf{0}. \end{cases}$$

That is, when the sign pattern of $\boldsymbol{x}_\lambda$ is fixed (so is $\mathcal{I}$), $\boldsymbol{x}_\lambda$ is linear in λ.

LAR: find the knots λ where $\text{sign}(\boldsymbol{x}_\lambda)$ changes.

$$\lambda \text{sign}(\boldsymbol{x}_\lambda) = \boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_\lambda) = \boldsymbol{A}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{A}_\mathcal{I}\boldsymbol{x}_{\lambda,\mathcal{I}}) = \boldsymbol{A}^\mathsf{T}\boldsymbol{r}_\lambda$$

- Keep track $\boldsymbol{x}_\lambda$ and $|\left\langle \boldsymbol{a}_j, \boldsymbol{r}_\lambda \right\rangle|$ until
  - Either $|\mathcal{I}|$ increases

$$\max_{j \notin \mathcal{I}} \left|\left\langle \boldsymbol{a}_j, \boldsymbol{r}_\lambda \right\rangle\right| = \lambda.$$

    Define $i = \arg\max_{j \notin \mathcal{I}} |\left\langle \boldsymbol{a}_j, \boldsymbol{r}_\lambda \right\rangle|$ and set $\mathcal{I} = \mathcal{I} \cup \{i\}$.
  - Or $|\mathcal{I}|$ decreases
    For some $i \in \mathcal{I}$,

$$|\langle \boldsymbol{a}_i, \boldsymbol{r}_\lambda \rangle| = \lambda \Rightarrow (\boldsymbol{x}_\lambda)_i = 0.$$

    Set $\mathcal{I} = \mathcal{I} \backslash \{i\}$.

Example: Find λ_0 such that

$$\begin{cases} \|\boldsymbol{x}_\lambda\|_0 = 0, & \forall \lambda > \lambda_0, \\ \|\boldsymbol{x}_\lambda\|_0 = 1, & \forall \lambda \in (\lambda_0 - \epsilon, \lambda_0), \end{cases}$$

for some $\epsilon > 0$.

Let $\text{supp}(\boldsymbol{x}_\lambda) = \{n\}$ for $\lambda \in (\lambda_0 - \epsilon, \lambda_0)$.

The optimality of $\boldsymbol{x}_\lambda$ suggests that

$$\begin{aligned} &\boldsymbol{A}^T(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_\lambda) = \lambda \text{sign}(\boldsymbol{x}_\lambda) \\ \Rightarrow &\boldsymbol{a}_n^T(\boldsymbol{y} - \boldsymbol{a}_n x_{\lambda,n}) = \lambda \text{sign}(x_{\lambda,n}) \\ \Rightarrow &x_{\lambda,n} = \boldsymbol{a}_n^T \boldsymbol{y} - \lambda \text{sign}(x_{\lambda,n}) \end{aligned}$$

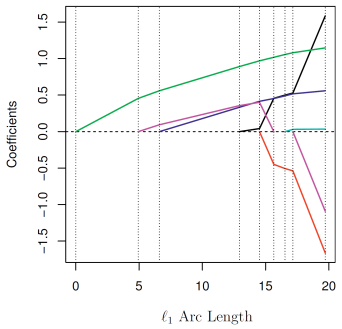


Figure 7.5: Typical behaviour of $\boldsymbol{x}_\lambda$. (Source: T. Hastie, et al., Statistical learning with sparsity: the lasso and generalizations. Chapman and Hall/CRC, 2015: page 120.)

Hence,

$$x_i^\star = \left(\left|\boldsymbol{a}_i^T \boldsymbol{y}\right| - \lambda\right) \operatorname{sign}(\boldsymbol{a}_i^T \boldsymbol{y}), \text{ and}$$
$$\lambda_0 = \max_i \ |\boldsymbol{a}_i^T \boldsymbol{y}|.$$

Bilinear Inverse Problems 8

8.1 Bilinear Inverse Problems 55
8.1.1 Matrix Norms 57
8.2 Examples 58
8.2.1 Low-Rank Matrix Completion 58
8.2.2 Blind Deconvolution . . . 60
8.2.3 Sparse PCA and Dictionary Learning 61
8.3 Methods 61
8.3.1 Alternating Optimization 61
8.3.2 Greedy Algorithms 62
8.3.3 Convex Relaxation 62
8.4 Appendix 63
8.4.1 Proof of Eckart-Young-Mirsky Theorem 63

8.1 Bilinear Inverse Problems

A linear map $\mathbb{L}$ satisfies

$$\mathbb{L}(\lambda_1 \boldsymbol{x}_1 + \lambda_2 \boldsymbol{x}_2) = \lambda_1 \mathbb{L}(\boldsymbol{x}_1) + \lambda_2 \mathbb{L}(\boldsymbol{x}_2)$$

for all $\lambda_1, \lambda_2 \in \mathbb{R}$ and $\boldsymbol{x}_1, \boldsymbol{x}_2$ in the domain of $\mathbb{L}$.

The general form of linear functions (domain of $\mathbb{L}$ is finite dimensional) is given by

$$\mathbb{L}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}.$$

A noisy linear system is typically written as

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}.$$

A linear inverse problem is given by

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2.$$

A bilinear map $\mathbb{B}$ is a map such that

$$\begin{aligned}\mathbb{B}(\lambda_1 \boldsymbol{p}_1 + \lambda_2 \boldsymbol{p}_2, \boldsymbol{q}) &= \lambda_1 \mathbb{B}(\boldsymbol{p}_1, \boldsymbol{q}) + \lambda_2 \mathbb{B}(\boldsymbol{p}_2, \boldsymbol{q}) \\ \mathbb{B}(\boldsymbol{p}, \lambda_1 \boldsymbol{q}_1 + \lambda_2 \boldsymbol{q}_2) &= \lambda_1 \mathbb{B}(\boldsymbol{p}, \boldsymbol{q}_1) + \lambda_2 \mathbb{B}(\boldsymbol{p}, \boldsymbol{q}_2)\end{aligned}$$

for all $\lambda_1, \lambda_2 \in \mathbb{R}$ and $\boldsymbol{p}, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{q}, \boldsymbol{q}_1, \boldsymbol{q}_2$, in the domain of $\mathbb{B}$.

The general form of bilinear map (domain of $\mathbb{B}$ is finite dimensional) is given by

$$\begin{aligned}\mathbb{B}(\boldsymbol{p}, \boldsymbol{q}) &= \left[f_1(\boldsymbol{p}, \boldsymbol{q}), \cdots, f_m(\boldsymbol{p}, \boldsymbol{q})\right]^\mathsf{T} \\ &= \left[\boldsymbol{p}^T \boldsymbol{A}_1 \boldsymbol{q}, \cdots, \boldsymbol{p}^T \boldsymbol{A}_m \boldsymbol{q}\right]^\mathsf{T},\end{aligned}$$

where $f_l(\boldsymbol{p}, \boldsymbol{q}) = \boldsymbol{p}^T \boldsymbol{A}_l \boldsymbol{q}$ are bilinear functions.

The above bilinear form can be written in a more convenient form. For this, we need the following definitions.

Definition 8.1.1 (Trace) *The trace of an $n \times n$ square matrix $\boldsymbol{A}$ is defined as the sum of diagonal entries, i.e.,*

$$tr(\boldsymbol{A}) = \sum_{i=1}^{n} a_{i,i}.$$

Definition 8.1.2 (Vectorization) *The vectorization of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a linear map which converts the matrix into a column vector by stacking*

the columns of the matrix $\boldsymbol{A}$ together:

$$\begin{aligned} vec(\boldsymbol{A}) &= [\boldsymbol{a}_{:,1}^{\mathsf{T}}, \cdots, \boldsymbol{a}_{:,n}^{\mathsf{T}}]^{\mathsf{T}} \\ &= [a_{1,1}, \cdots, a_{m,1}, \cdots, a_{1,n}, \cdots, a_{m,n}]^{\mathsf{T}}. \end{aligned}$$

Lemma 8.1.1 (Properties of Trace)

1. $$tr(\boldsymbol{A}) = tr(\boldsymbol{A}^{\mathsf{T}}).$$

2. $$tr(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{B}) = tr(\boldsymbol{B}\boldsymbol{A}^{\mathsf{T}}) = \sum_{i,j} a_{i,j} b_{j,i}.$$

3. $$tr(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{B}) = tr(\boldsymbol{A}\boldsymbol{B}^{\mathsf{T}}) = vec(\boldsymbol{A})^{\mathsf{T}} vec(\boldsymbol{B}).$$

Definition 8.1.3 (Frobenius Inner Product) *Given two matrices of the same size $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{C}^{n \times m}$, the Frobenius inner product is defined as*

$$\begin{aligned} \langle \boldsymbol{A}, \boldsymbol{B} \rangle &:= \sum_{i,j} \overline{a_{i,j}} b_{i,j} \\ &= tr(\boldsymbol{A}^{\mathsf{H}} \boldsymbol{B}), \end{aligned}$$

where the overline denotes the complex conjugate and the superscript H denotes Hermitian conjugate.

For two real matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}$, the Frobenius inner product becomes

$$\langle \boldsymbol{A}, \boldsymbol{B} \rangle = tr(\boldsymbol{A}^{\mathsf{T}} \boldsymbol{B}).$$

We write a bilinear map using trace.

$$\begin{aligned} \mathbb{B}(\boldsymbol{p}, \boldsymbol{q}) &= [\boldsymbol{p}^{\mathsf{T}} \boldsymbol{A}_1 \boldsymbol{q}, \cdots, \boldsymbol{p}^{\mathsf{T}} \boldsymbol{A}_m \boldsymbol{q}]^{\mathsf{T}} \\ &= [\mathrm{tr}(\boldsymbol{p}^{\mathsf{T}} \boldsymbol{A}_1 \boldsymbol{q}), \cdots, \mathrm{tr}(\boldsymbol{p}^{\mathsf{T}} \boldsymbol{A}_m \boldsymbol{q})]^{\mathsf{T}} \\ &\overset{(a)}{=} [\mathrm{tr}(\boldsymbol{A}_1 \boldsymbol{q}\boldsymbol{p}^{\mathsf{T}}), \cdots, \mathrm{tr}(\boldsymbol{A}_m \boldsymbol{q}\boldsymbol{p}^{\mathsf{T}})]^{\mathsf{T}} \\ &\overset{(b)}{=} [\langle \boldsymbol{A}_1^{\mathsf{T}}, \boldsymbol{q}\boldsymbol{p}^{\mathsf{T}} \rangle, \cdots, \langle \boldsymbol{A}_m^{\mathsf{T}}, \boldsymbol{q}\boldsymbol{p}^{\mathsf{T}} \rangle]^{\mathsf{T}} \\ &= \begin{bmatrix} \mathrm{vect}(\boldsymbol{A}_1)^{\mathsf{T}} \\ \vdots \\ \mathrm{vect}(\boldsymbol{A}_m)^{\mathsf{T}} \end{bmatrix} \mathrm{vect}(\boldsymbol{p}\boldsymbol{q}^{\mathsf{T}}) \\ &= \mathbb{A}\left(\mathrm{vect}(\boldsymbol{p}\boldsymbol{q}^{\mathsf{T}})\right), \end{aligned}$$

where (a) follows from Lemma 8.1.1-2 and (b) comes from Definition 8.1.3.

Hence, a noisy bilinear system can be expressed as

$$\begin{aligned} \boldsymbol{y} &= \mathbb{A}(\boldsymbol{p}\boldsymbol{q}^{T}) + \boldsymbol{w} \\ &= \boldsymbol{A}\mathrm{vect}\left(\boldsymbol{p}\boldsymbol{q}^{T}\right) + \boldsymbol{w}. \end{aligned}$$

A bilinear inverse problem can be written as

$$\min_{p,q} \frac{1}{2} \left\| y - A\mathrm{vect}\left(pq^T\right) \right\|_2^2$$

8.1.1 Matrix Norms

Frobenius Norm

$$\|A\|_F = \left(\sum_{i,j} |a_{i,j}|^2 \right)^{1/2} = \left(\sum_k \sigma_k^2 \right)^{1/2}.$$

Operator Norm

Operator norms of matrices are *induced* by vector norms.

Definition 8.1.4 (Operator Norm)

$$\|A\| = \sup\{\|Ax\| : \|x\| = 1\}.$$

Example 8.1.1

- $p = 1$:

$$\begin{aligned} \|A\|_1 &= \max_{1 \le j \le n} \sum_{i=1}^{m} |a_{i,j}| \\ &= \max_{1 \le j \le n} \|a_{:,j}\|_1. \end{aligned}$$

- $p = \infty$:

$$\begin{aligned} \|A\|_\infty &= \max_{1 \le i \le m} \sum_{j=1}^{n} |a_{i,j}| \\ &= \max_{1 \le i \le m} \|a_{i,:}\|_1. \end{aligned}$$

- $p = 2$:

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^\mathsf{T} A)} = \sigma_{\max}(A).$$

Remark 8.1.1 Note the difference between ℓ_2-norm and Frobenius norm of matrices.

$$\begin{aligned} \|A\|_2 &= \sigma_{\max}(A) \\ &\le \|A\|_F = \left(\sum_{i,j} |a_{i,j}|^2 \right)^{1/2} = \left(\sum_k \sigma_k^2 \right)^{1/2}, \end{aligned}$$

where the equality holds if and only if A has a rank at most one.

$L_{2,1}$-norm

$$\|\boldsymbol{A}\|_{2,1} := \sum_{j=1}^{n} \|\boldsymbol{a}_{:,j}\|_2 = \sum_{j=1}^{n} \left(\sum_{i=1}^{m} |a_{i,j}|^2 \right)^{1/2}$$

This norm has been widely used to promote sparsity in columns of a matrix.

Schatten Norms

The Schatten p-norm applies the p-norm to the singular value vector:

$$\|\boldsymbol{A}\|_p = \left(\sum_i \sigma_i^p(\boldsymbol{A}) \right)^{1/p}.$$

The most popular cases include

- $p = 1$: nuclear norm.

$$\|\boldsymbol{A}\|_* = \sum_i \sigma_i(\boldsymbol{A}). \tag{8.1}$$

  Nuclear norm is often used to prompt a low-rank solution.
- $p = 2$: Frobenius norm.

8.2 Examples

The first example is designed to show that bilinear inverse problems are inherently nonconvex.

Example 8.2.1 (A simple bilinear inverse problem) Consider the following bilinear inverse problem

$$f(u, v) = (1 - uv)^2.$$

- For a fixed u, $f(u, v)$ is convex in v.
- For a fixed v, $f(u, v)$ is convex in u.
- $f(u, v)$ is not convex in (u, v). To see this, consider two points $(u_1, v_1) = (0.1, 10)$ and $(u_2, v_2) = (10, 0.1)$.

See below for other examples of bilinear inverse problems.

8.2.1 Low-Rank Matrix Completion

Example 8.2.2 (Netflix Problem)

	Movie 1	Movie 2	Movie 3	Movie 4
Person 1		★★★★★	★★★☆☆	
Person 2	★★★★☆		★★★★★	
Person 3		★★★☆☆		★★★★☆
Person 4	★★★★★			★★★★★

Figure 8.1: Netflix problem https://en.wikipedia.org/wiki/Netflix_Prize

A low rank model: the full matrix is of low-rank.

The inverse problem is then given by

$$\min_{\boldsymbol{P}\in\mathbb{R}^{m,r},\boldsymbol{Q}\in\mathbb{R}^{n,r}} \frac{1}{2}\left\|\boldsymbol{Y}_\Omega - (\boldsymbol{P}\boldsymbol{Q}^T)_\Omega\right\|_F^2,$$
$$\equiv$$
$$\min_{\boldsymbol{X}:\ \mathrm{rank}(\boldsymbol{X})\le r} \frac{1}{2}\|\boldsymbol{Y}_\Omega - \boldsymbol{X}_\Omega\|_F^2$$

where Ω is the index set of revealed entries.

Example 8.2.3 (Simple Examples of Low Rank Matrix Completion)

1. Find a rank-1 matrix given the observations

$$\begin{bmatrix} 1 & ? & 3 \\ 2 & -2 & ? \\ ? & -1 & ? \end{bmatrix}$$

2. Find a rank-2 matrix given the observations

$$\begin{bmatrix} 1 & 2 & ? & ? \\ 1 & 2 & 0 & 4 \\ 1 & 0 & 2 & 2 \\ ? & ? & 2 & 2 \end{bmatrix}$$

Eckart-Young-Mirsky Theorem

Though bilinear inverse problem is nonconvex in general, we present a special case where a global minimizer can be easily found (via SVD).

Let $\boldsymbol{Y} \in \mathbb{R}^{m\times n}$ be a given matrix. Consider the following nonconvex optimization problem

$$\min_{\boldsymbol{X}:\ \mathrm{rank}(X)\le r} \|\boldsymbol{Y}-\boldsymbol{X}\|_F^2 \ \equiv \ \min_{\boldsymbol{P}\in\mathbb{R}^{m\times r},\boldsymbol{Q}\in\mathbb{R}^{n\times r}} \|\boldsymbol{Y}-\boldsymbol{P}\boldsymbol{Q}^T\|_F^2$$

Theorem 8.2.1 (The Eckart-Young-Mirsky Theorem) *Consider SVD of the matrix $\boldsymbol{Y} = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\mathsf{T}$. The best rank r approximation of $\boldsymbol{Y}$ is given by the truncated SVD*

$$\boldsymbol{X}_r = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\mathsf{T}.$$

Remark 8.2.1 (Weighted Low-Rank Approximation Problems) Consider the weighted low-rank approximation problems of the form

$$\min_{\boldsymbol{X}:\ \mathrm{rank}(X)\le r} \frac{1}{2}\mathrm{vec}(\boldsymbol{Y}-\boldsymbol{X})^\mathsf{T}\boldsymbol{W}\mathrm{vec}(\boldsymbol{Y}-\boldsymbol{X}).$$

The general weighted low-rank approximation problem does not admit an analytic solution. Global optimality of the solution is not guaranteed.

Remark 8.2.2 (Uniqueness of the Solution)

$$\min_{\boldsymbol{X}:\ \mathrm{rank}(X)\le r} \|\boldsymbol{Y}-\boldsymbol{X}\|_F^2 \ \equiv \ \min_{\boldsymbol{P}\in\mathbb{R}^{m\times r},\boldsymbol{Q}\in\mathbb{R}^{n\times r}} \|\boldsymbol{Y}-\boldsymbol{P}\boldsymbol{Q}^\mathsf{T}\|_F^2$$

- Optimal solutions $(\boldsymbol{P}^\star, \boldsymbol{Q}^\star)$ are not unique: scaling/rotation invariant.
  - If $(\boldsymbol{P}^\star, \boldsymbol{Q}^\star)$ is the optimal solution, so is $(\boldsymbol{P}^\star\boldsymbol{S}, \boldsymbol{Q}^\star\boldsymbol{S}^{-\mathsf{T}})$ for any invertible $\boldsymbol{S}\in\mathbb{R}^{r\times r}$. This is because

$$\boldsymbol{P}^\star\boldsymbol{Q}^{\star\mathsf{T}} = \boldsymbol{P}^\star(\boldsymbol{S}\boldsymbol{S}^{-1})\boldsymbol{Q}^{\star\mathsf{T}} = (\boldsymbol{P}^\star\boldsymbol{S})(\boldsymbol{Q}^\star\boldsymbol{S}^{-\mathsf{T}})^\mathsf{T}$$

  A solution $(\boldsymbol{P}^\star, \boldsymbol{Q}^\star)$ is optimal/unique up to scaling / rotation / permutation.
- $\boldsymbol{X}^\star = \boldsymbol{P}^\star\boldsymbol{Q}^{\star\mathsf{T}}$ is unique.

8.2.2 Blind Deconvolution

Example 8.2.4 (Blind Deconvolution) Assume that the observed data is a convolution of an unknown source $\boldsymbol{s}$ and an unknown channel $\boldsymbol{h}$, i.e.,

$$\boldsymbol{y} = \boldsymbol{s}\circledast\boldsymbol{h}:\ y[n] = \sum_{\ell=0}^{L} s[n-\ell]\,h[\ell]\,.$$

It is straightforward to verify that blind deconvolution problem is a bilinear inverse problem.

Figure 8.2 shows one simulation result from the seminal paper Ahmed, Recht, and Romberg, 2013.

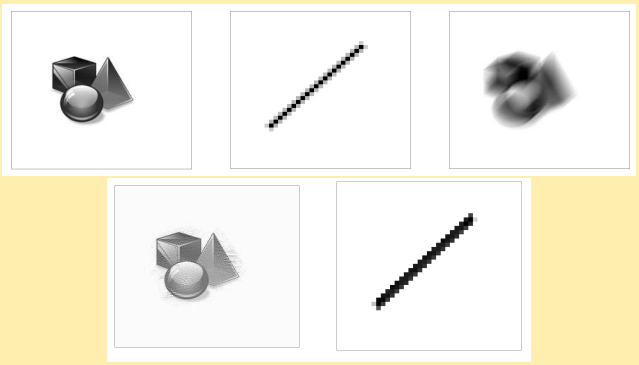


Figure 8.2: Blind deconvolution (Source: Ahmed, Recht, and Romberg, 2013). The top row from left to right: source, channel, and convolution of source and channel. The bottom row from left to right: estimated source and channel via solving the corresponding bilinear inverse problem.

Blind deconvolution can be formulated as a low-rank matrix recovery problem as follows. Define

$$\boldsymbol{X} := \boldsymbol{h}\boldsymbol{s}^T = \begin{bmatrix} h[0]s[0] & h[0]s[1] & h[0]s[2] & h[0]s[3] & \cdots \\ h[1]s[0] & h[1]s[1] & h[1]s[2] & h[1]s[3] & \cdots \\ h[2]s[0] & h[2]s[1] & h[2]s[2] & h[2]s[3] & \cdots \end{bmatrix}.$$

It is clear that the matrix $\boldsymbol{X}$ has rank one. The observations $\boldsymbol{y}$ can be

written as

$$\boldsymbol{y} = \begin{bmatrix} y[0] \\ y[1] \\ y[2] \\ y[3] \\ \vdots \end{bmatrix} = \mathbb{A}(\boldsymbol{X})$$
$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \operatorname{vect}(\boldsymbol{X}).$$

The inverse problem can be written as

$$\boldsymbol{X} : \operatorname*{rank}(\boldsymbol{X}) \leq 1 \min \frac{1}{2} \|\boldsymbol{y} - \mathbb{A}(\boldsymbol{X})\|_F^2.$$

8.2.3 Sparse PCA and Dictionary Learning

Example 8.2.5 (Sparse PCA and Dictionary Learning)

$$\underbrace{\boldsymbol{Y}}_{m \times n} = \boldsymbol{D}\boldsymbol{S}$$
$$= \underbrace{\begin{bmatrix} \boldsymbol{d}_1 & \cdots & \boldsymbol{d}_k \end{bmatrix}}_{m \times k} \underbrace{\boldsymbol{S}}_{k \times n}$$

Typical assumption: $\boldsymbol{S}$ is sparse.

- Sparse PCA
- Dictionary learning

The same formulation is applicable to blind source separation.[1]

1: https://en.wikipedia.org/wiki/Signal_separation

8.3 Methods

8.3.1 Alternating Optimization

$$\min_{\boldsymbol{P} \in \mathbb{R}^{m \times r}, \boldsymbol{Q} \in \mathbb{R}^{n \times r}} \left\| \boldsymbol{y} - \mathbb{A}(\boldsymbol{P}\boldsymbol{Q}^T) \right\|_2^2$$

Alternating projection: in each iteration

1. Fix $\boldsymbol{P}$, optimise $\boldsymbol{Q}$:

$$\boldsymbol{Q}^{l+1} = \arg\min_{\boldsymbol{Q}} \|\boldsymbol{y} - \mathbb{A}(\boldsymbol{P}^l \boldsymbol{Q}^T)\|_2^2$$
$$= \arg\min_{\boldsymbol{Q}} \|\boldsymbol{y} - \mathbb{A}_{\boldsymbol{P}^l}(\boldsymbol{Q}^T)\|_2^2$$

2. Fix $\boldsymbol{Q}$, optimise $\boldsymbol{P}$:

$$\begin{aligned}\boldsymbol{P}^{l+1} &= \arg\min_{\boldsymbol{P}} \|\boldsymbol{y} - \mathbb{A}(\boldsymbol{P}\boldsymbol{Q}^{l+1,T})\|_2^2 \\ &= \arg\min_{\boldsymbol{P}} \|\boldsymbol{y} - \mathbb{A}_{\boldsymbol{Q}^{l+1}}(\boldsymbol{P})\|_2^2\end{aligned}$$

Remark 8.3.1 1. Each step is to solve a least squares problem of which the global optimal solution can be computed.
2. The objective function is monotonically decreasing. Convergence is guaranteed.

8.3.2 Greedy Algorithms

OMP, SP, CoSaMP and IHT can be adapted to solve low-rank matrix recovery problems. In the following, we use IHT as an example.

For greedy algorithms, the optimization problem can be formulated as

$$\min_{\boldsymbol{X}:\ \mathrm{rank}(\boldsymbol{X})\le r} \|\boldsymbol{Y} - \mathbb{A}(\boldsymbol{X})\|_F^2.$$

The proximal operator of rank indicator function is given as

$$\begin{aligned}\boldsymbol{X}^\star &= \mathrm{Prox}_{\gamma\delta(\mathrm{rank}(\cdot)\le r)}(\boldsymbol{Z}) \\ &= \arg\min_{\boldsymbol{X}}\ \delta\left(\mathrm{rank}(\boldsymbol{X}) \le r\right) + \frac{1}{2\gamma}\|\boldsymbol{X} - \boldsymbol{Z}\|_F^2 \\ &= \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}} \\ &=: H_{\text{rank-}r}(\boldsymbol{Z}),\end{aligned}$$

where

$$\boldsymbol{Z} = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}}.$$

The IHT algorithm for low-rank matrix recovery is then given by

$$\boldsymbol{X}^{l+1} = H_{\text{rank-}r}\left(\boldsymbol{X}^l + \tau^l \mathbb{A}^*\left(\boldsymbol{y} - \mathbb{A}\left(\boldsymbol{X}^l\right)\right)\right),$$

where $\mathbb{A}^*$ is the adjoint operator of the linear map $\mathbb{A}$: if we represent the linear map $\mathbb{A}$ by a matrix $\boldsymbol{A}$, then $\mathbb{A}^* = \boldsymbol{A}^{\mathsf{T}}$ if $\boldsymbol{A}$ is a real matrix and $\mathbb{A}^* = \boldsymbol{A}^{\mathsf{H}}$ if $\boldsymbol{A}$ is a complex matrix.

8.3.3 Convex Relaxation

Nuclear norm

$$\|A\|_* = \sum_i \sigma_i(\boldsymbol{A}).$$

can be viewed as ℓ_1-norm of the singular value vector, promoting sparsity in singular value vector, i.e., low-rank solutions.

A Lasso type formulation to promote low-rank solutions:

$$\min_{X} \frac{1}{2} \left\| \boldsymbol{y} - \mathbb{A}\left(\boldsymbol{X}\right) \right\|_2^2 + \lambda \left\| \boldsymbol{X} \right\|_* .$$

An ISTA algorithm can be developed as follows. Note that

$$\partial \left\| \boldsymbol{X} \right\|_* = \sum_{i=1}^{\min(m,n)} \operatorname{sign}\left(\sigma_i\right) \boldsymbol{u}_i \boldsymbol{v}_i^T$$

The proximal operator of nuclear norm $\|\cdot\|_*$ is given by

$$\begin{aligned} \boldsymbol{X}^\star &= \operatorname{Prox}_{\gamma\|\cdot\|_*}(\boldsymbol{Z}) \\ &= \arg\min_{X} \ \|\boldsymbol{X}\|_* + \frac{1}{2\gamma}\|\boldsymbol{X}-\boldsymbol{Z}\|_F^2 \\ &= \sum_{i=1}^{\min(m,n)} \eta(\sigma_i;\gamma)\boldsymbol{u}_i\boldsymbol{v}_i^{\mathsf{T}}, \end{aligned}$$

where

- $\boldsymbol{Z} = \sum_{i=1}^{\min(m,n)} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}}$, and
- $\eta(\sigma_i;\gamma) = \max(0, \sigma_i - \gamma)$ is the soft thresholding function.[2]

2: See (7.2). Here, as by default singular values are non-negative, the form of soft-thresholding function can be simplified.

Let $f(\boldsymbol{X}) = \frac{1}{2}\left\|\boldsymbol{y} - \mathbb{A}(\boldsymbol{X})\right\|_2^2$. Note that

$$\begin{aligned} \nabla f(\boldsymbol{X}) &= -\mathbb{A}^*(\boldsymbol{y} - \mathbb{A}(\boldsymbol{X})) \\ &= -\boldsymbol{A}^{\mathsf{T}}\left(\boldsymbol{y} - \boldsymbol{A}\operatorname{vec}(\boldsymbol{X})\right), \end{aligned}$$

where the linear map $\mathbb{A}$ is represented by a matrix $\boldsymbol{A}$. Approximate the objective function locally by

$$\begin{aligned} & f(\boldsymbol{X}) + \lambda \left\|\boldsymbol{X}\right\|_* \\ & \approx f(\boldsymbol{X}^l) + \left\langle \boldsymbol{X} - \boldsymbol{X}^l, \nabla f(\boldsymbol{X}^l) \right\rangle + \frac{1}{2\tau^l}\left\|\boldsymbol{X} - \boldsymbol{X}^l\right\|_2^2 + \lambda \left\|\boldsymbol{X}\right\|_* \\ & = \frac{1}{2\tau^l}\left\|\boldsymbol{X} - \left(\boldsymbol{X}^l - \tau^l \nabla f(\boldsymbol{X}^l)\right)\right\|_2^2 + \lambda \left\|\boldsymbol{X}\right\|_* + c. \end{aligned}$$

Hence, the ISTA iterations are given as

$$\boldsymbol{X}^{l+1} = \eta_\sigma\left(\boldsymbol{X}^l + \tau^l \mathbb{A}^*\left(\boldsymbol{y} - \mathbb{A}(\boldsymbol{X}^l)\right); \lambda\tau^l\right),$$

where the notation η_σ emphasizes that the soft-thresholding function acts on the singular values of the matrix (rather than the elements of the matrix). In practice, the step size τ^l is fixed as $\tau^l = \tau \in (0, 2/\sigma_{\max}(\mathbb{A}^*\mathbb{A})) = (0, 2/\sigma_{\max}(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}))$.

8.4 Appendix

8.4.1 Proof of Eckart-Young-Mirsky Theorem

We first prove Eckart–Young–Mirsky theorem for spectral norm and then for Frobenius norm.

Consider singular value decomposition

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{T}} = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}}.$$

Consider the rank-k approximation generated by SVD given by

$$\boldsymbol{A}_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}}.$$

Consider any given (arbitrary) rank-k approximation

$$\boldsymbol{B}_k = \boldsymbol{P}\boldsymbol{Q}^{\mathsf{T}},$$

where both $\boldsymbol{P} \in \mathbb{R}^{m\times k}$ and $\boldsymbol{Q} \in \mathbb{R}^{n\times k}$ have k columns.

To prove Eckart–Young–Mirsky theorem for spectral norm, it suffices to show that

$$\|\boldsymbol{A} - \boldsymbol{A}_k\|_2 = \sigma_{k+1} \leq \|\boldsymbol{A} - \boldsymbol{B}_k\|_2.$$

Towards this end, note that $\boldsymbol{Q}$ has only k columns. There must exist a vector $\boldsymbol{w}$ such that

1. $\boldsymbol{Q}^{\mathsf{T}}\boldsymbol{w} = \boldsymbol{0}$;
2. $\boldsymbol{w} = \gamma_1 \boldsymbol{v}_1 + \cdots + \gamma_{k+1}\boldsymbol{v}_{k+1}$;
3. $\|\boldsymbol{w}\|_2 = 1$; This implies that $\|\boldsymbol{\gamma}\|_2^2 = \sum_{i=1}^{k+1} \gamma_i^2 = 1$.

Then

$$\begin{aligned}
\|\boldsymbol{A} - \boldsymbol{B}_k\|_2^2 &\geq \|(\boldsymbol{A} - \boldsymbol{B}_k)\boldsymbol{w}\|_2^2 \\
&= \|\boldsymbol{A}\boldsymbol{w}\|_2^2 = \gamma_1^2\sigma_1^2 + \cdots + \gamma_{k+1}^2\sigma_{k+1}^2 \\
&\geq \sigma_{k+1}^2,
\end{aligned}$$

which concludes the proof for the spectral norm.

For Frobenius norm, the goal is to establish

$$\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2 = \sum_{i=k+1}^{n} \sigma_i^2 \leq \|\boldsymbol{A} - \boldsymbol{B}_k\|_F^2.$$

Let $\boldsymbol{A}' = \boldsymbol{A} - \boldsymbol{B}_k$ and $\boldsymbol{A}'' = \boldsymbol{B}_k$. Let $\boldsymbol{A}'_i$ and $\boldsymbol{A}''_i$ be the rank-i approximation of Let $\boldsymbol{A}'$ and $\boldsymbol{A}''$ by using SVD, respectively. Then for any $i, j \geq 1$,

$$\begin{aligned}
\sigma_i(\boldsymbol{A}') + \sigma_j(\boldsymbol{A}'') &= \sigma_1(\boldsymbol{A}' - \boldsymbol{A}'_{i-1}) + \sigma_1(\boldsymbol{A}'' - \boldsymbol{A}''_{j-1}) \\
&\overset{(a)}{\geq} \sigma_1\left(\boldsymbol{A} - (\boldsymbol{A}'_{i-1} + \boldsymbol{A}''_{j-1})\right) \\
&\overset{(b)}{\geq} \sigma_1\left(\boldsymbol{A} - \boldsymbol{A}_{i+j-2}\right) \\
&= \sigma_{i+j-1}(\boldsymbol{A}),
\end{aligned}$$

where (a) comes from triangle inequality with the spectral norm:

$$\begin{aligned}
&\sigma_1\left(\boldsymbol{A} - (\boldsymbol{A}'_{i-1} + \boldsymbol{A}''_{j-1})\right) \\
&= \sigma_1\left(\boldsymbol{A}' - \boldsymbol{A}'_{i-1} + \boldsymbol{A}'' - \boldsymbol{A}''_{j-1}\right) \\
&\leq \sigma_1\left(\boldsymbol{A}' - \boldsymbol{A}'_{i-1}\right) + \sigma_1\left(\boldsymbol{A}'' - \boldsymbol{A}''_{j-1}\right),
\end{aligned}$$

and (b) holds by combining the fact that

$$\operatorname{rank}\left(\boldsymbol{A}'_{i-1} + \boldsymbol{A}''_{j-1}\right) \leq \operatorname{rank}\left(\boldsymbol{A}_{i+j-2}\right),$$

and Eckart–Young–Mirsky theorem for spectral norm. Since $\sigma_{k+1}(\boldsymbol{B}_k) = 0$, one has

$$\sigma_i(\boldsymbol{A} - \boldsymbol{B}_k) \geq \sigma_{k+i}(\boldsymbol{A}).$$

This leads to

$$\begin{aligned}
\|\boldsymbol{A} - \boldsymbol{B}_k\|_F^2 &= \sum_{i=1}^{n} \sigma_i(\boldsymbol{A} - \boldsymbol{B}_k)^2 \\
&\geq \sum_{i=k+1}^{n} \sigma_i(\boldsymbol{A})^2 \\
&= \|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2,
\end{aligned}$$

which concludes the proof.

Constrained Convex Optimization

9

9.1 General Form

A constrained optimization problem of the form

$$\begin{aligned} \min_{\boldsymbol{x}} \; & f(\boldsymbol{x}) \\ \text{subject to } & h_i(\boldsymbol{x}) \leq 0, \; i = 1, \cdots, m, \\ & \ell_i(\boldsymbol{x}) = 0, \; i = 1, \cdots, r, \end{aligned} \tag{9.1}$$

where

- the objective function f_0 is convex,
- h_i's are convex, and
- ℓ_i's are affine, i.e., in the form of $\boldsymbol{a}_i^T \boldsymbol{x} + b_i = 0$.

In the following, we show that convex h_i and affine ℓ_i lead to a *convex* feasible set.

9.1.1 Sublevel Sets

Definition 9.1.1 (Sublevel Sets, a.k.a. Lower Contour Sets) *The* α-sublevel set *of a function* $f : \mathbb{R}^n \to \mathbb{R}$ *is defined as*

$$\mathcal{C}_\alpha = \left\{ \boldsymbol{x} \in dom\left(f\right) : \; f(\boldsymbol{x}) \leq \alpha \right\}.$$

Lemma 9.1.1 *Sublevel sets of a convex function* f *are convex.*

Proof. We shall show that for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{C}_\alpha$, it holds $\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y} \in \mathcal{C}_\alpha$ for all $\lambda \in [0,1]$.

By the definition of $\mathcal{C}_\alpha$, $f(\boldsymbol{x}) \leq \alpha$ and $f\left(\boldsymbol{y}\right) \leq \alpha$. By the convexity of f,

$$f\left(\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y}\right) \leq \lambda f(\boldsymbol{x}) + (1-\lambda) f\left(\boldsymbol{y}\right) \leq \alpha,$$

which proves this lemma. □

The converse of Lemma 9.1.1 is not true. That sublevel sets of a function f are convex does not imply that f is convex.

Consider the general form of constrained convex optimization (9.1). That h_i is convex ensures that

$$\{\boldsymbol{x} : \; h_i(\boldsymbol{x}) \leq 0\}$$

is convex by Lemma 9.1.1. Further note that

$$\{\boldsymbol{x} : \; \ell_i(\boldsymbol{x}) = 0\}$$

9.1 General Form 66
9.1.1 Sublevel Sets 66
9.2 Duality 67
9.3 KKT Conditions 69
9.4 Examples 70

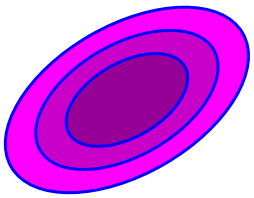


Figure 9.1: Sublevel sets of a convex function are convex

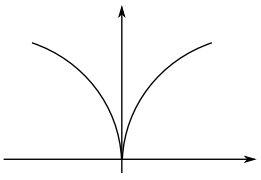


Figure 9.2: Converse of Lemma 9.1.1 is not true.

is convex. It can be concluded that

$$\left(\bigcap_i \{x : h_i(x) \le 0\}\right) \bigcap \left(\bigcap_j \{x : \ell_j(x) = 0\}\right)$$

is convex.

9.2 Duality

Consider a general optimization problem

$$\begin{aligned} &\min_x f(x) \\ &\text{s.t. } h_i(x) \le 0,\ i = 1, \cdots, m, \\ &\quad \ell_j(x) = 0,\ j = 1, \cdots, r. \end{aligned}$$

where the objective function f needs not to be convex. Of course we pay special attention to the convex case. Its Lagrangian is defined as follows.

Definition 9.2.1 (Lagrangian)

$$L(x, u, v) := f(x) + \sum_{i=1}^{m} u_i h_i(x) + \sum_{j=1}^{r} v_j \ell_j(x).$$

Here $u \in \mathbb{R}^m$, $v \in \mathbb{R}^r$, *and* $u \ge 0$.

The corresponding Lagrange dual function is defined as

Definition 9.2.2 (Lagrange Dual Function)

$$\begin{aligned} g(u, v) &:= \min_{x \in \mathbb{R}^n} L(x, u, v) \\ &= \min_{x \in \mathbb{R}^n} f(x) + \sum_{i=1}^{m} u_i h_i(x) + \sum_{j=1}^{r} v_j \ell_j(x). \end{aligned}$$

The following summaries some important properties of Lagrangian and Lagrange dual function.

- For every *feasible* x, $L(x, u, v) \le f(x)$:

$$\begin{aligned} L(x, u, v) &= f(x) + \underbrace{\sum_{i=1}^{m} u_i h_i(x)}_{\le 0} + \underbrace{\sum_{j=1}^{r} v_j \ell_j(x)}_{=0} \\ &\le f(x). \end{aligned}$$

- Let $\mathcal{X}$ denote the primal feasible set.

$$g(u, v) = \min_{x \in \mathbb{R}^n} L(x, u, v) \le \min_{x \in \mathcal{X}} L(x, u, v) \le f(x). \tag{9.2}$$

- $g(u, v)$ is concave in (u, v). (See the lemma below.)

Lemma 9.2.1 *The Lagrange dual function*

$$\begin{aligned} g(\boldsymbol{u},\boldsymbol{v}) &= \min_{\boldsymbol{x}\in\mathbb{R}^n} L(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}) \\ &= \min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) + \sum_{i=1}^{m} u_i h_i(\boldsymbol{x}) + \sum_{j=1}^{r} v_j \ell_j(\boldsymbol{x}) \end{aligned}$$

is concave in $(\boldsymbol{u},\boldsymbol{v})$.

We prove Lemma 9.2.1 by proving the following lemma.

Lemma 9.2.2

- *Let* $f_\alpha(\boldsymbol{x})$ *be concave. Then* $g(\boldsymbol{x}) = \inf_\alpha f_\alpha(\boldsymbol{x})$ *is concave.*
- *Let* $f_\alpha(\boldsymbol{x})$ *be convex. Then* $g(\boldsymbol{x}) = \sup_\alpha f_\alpha(\boldsymbol{x})$ *is convex.*

Figure 9.3: Pointwise Maximum

Proof. For any $\lambda \in [0,1]$,

$$\begin{aligned} g(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}) &= \inf_\alpha f_\alpha(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}) \\ &\geq \inf_\alpha \lambda f_\alpha(\boldsymbol{x}) + (1-\lambda) f_\alpha(\boldsymbol{y}) \\ &\geq \lambda \inf_\alpha f_\alpha(\boldsymbol{x}) + (1-\lambda) \inf_\alpha f_\alpha(\boldsymbol{y}). \end{aligned}$$

where the first inequality comes from concavity of f_α. □

Proof of Lemma 9.2.1. For any given $\boldsymbol{x}$, $L(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v})$ is linear in $(\boldsymbol{u},\boldsymbol{v})$, and hence concave in $(\boldsymbol{u},\boldsymbol{v})$. Lemma 9.2.1 is then a direct consequence of Lemma 9.2.2. □

Definition 9.2.3 (Lagrange Dual Problem) *Given the primal problem*

$$\begin{aligned} &\min_{\boldsymbol{x}} f(\boldsymbol{x}) \\ &s.t.\ h_i(\boldsymbol{x}) \leq 0,\ i = 1,\cdots,m, \\ &\quad \ell_j(\boldsymbol{x}) = 0,\ j = 1,\cdots,r, \end{aligned}$$

its Lagrange dual problem is

$$\begin{aligned} &\max_{\boldsymbol{u},\boldsymbol{v}} g(\boldsymbol{u},\boldsymbol{v}) \\ &s.t.\ \boldsymbol{u} \geq \boldsymbol{0}. \end{aligned}$$

- Weak duality: the dual optimal value $g^\star$ satisfies

$$f^\star \geq g^\star.$$

  This is a direct consequence of (9.2).
- Strong duality is referred to as the case that

$$f^\star = g^\star.$$

- Slater's condition: if the primal is a convex problem (i.e., f and g_i's are convex and ℓ_j's are affine), and there exists at least one strictly feasible $\boldsymbol{x} \in \mathbb{R}^n$ satisfying

$$h_i(\boldsymbol{x}) < 0, \ \forall i \in [m], \text{ and } \ell_j(\boldsymbol{x}) = 0, \ \forall j \in [r],$$

then strong duality holds. (Proof is omitted.)

9.3 KKT Conditions

Definition 9.3.1 (Karush-Kuhn-Tucker Conditions) *Given the optimization problem* (9.1)*, the Karush-Kuhn-Tucker (KKT) conditions for optimal* $\boldsymbol{x}^\star, \boldsymbol{u}^\star, \boldsymbol{v}^\star$ *are:*

- $\mathbf{0} \in \partial f(\boldsymbol{x}^\star) + \sum_{i=1}^m u_i^\star \partial h_i(\boldsymbol{x}^\star) + \sum_{j=1}^r v_j^\star \partial \ell_j(\boldsymbol{x}^\star)$. *(stationarity)*
- $u_i^\star h_i(\boldsymbol{x}^\star) = 0, \forall i$. *(complementary slackness)*
- $h_i(\boldsymbol{x}^\star) \le 0, \ell_j(\boldsymbol{x}^\star) = 0, \forall i, \forall j$. *(primal feasibility)*
- $u_i^\star \ge 0, \forall i$. *(dual feasibility)*

KKT conditions are

- Always sufficient.
- Necessary under strong duality.

Proof. We prove sufficiency first and then necessity.

1. (Sufficiency) If $\boldsymbol{x}^\star, \boldsymbol{u}^\star, \boldsymbol{v}^\star$ satisfy the KKT conditions, then

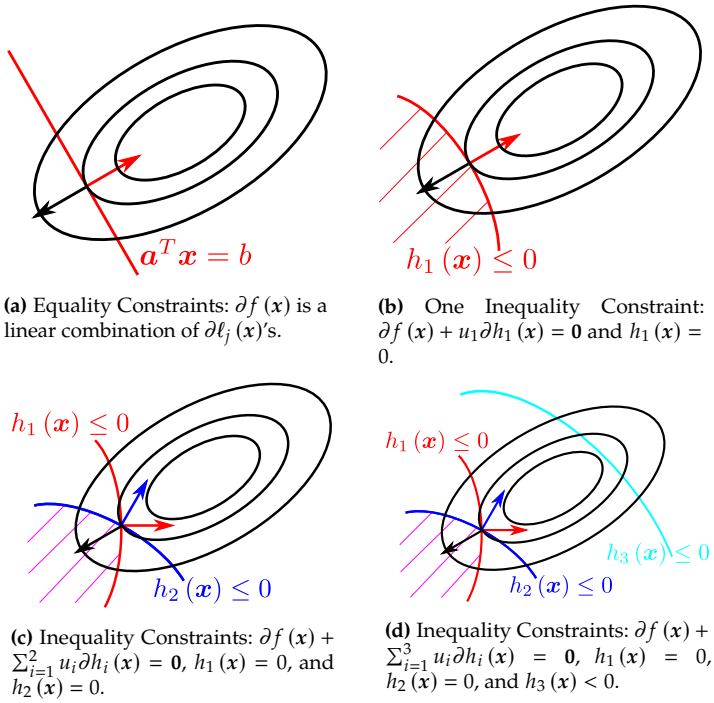
$$\begin{aligned} g(\boldsymbol{u}^\star, \boldsymbol{v}^\star) &= \min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{u}^\star, \boldsymbol{v}^\star) \\ &= f(\boldsymbol{x}^\star) + \sum_{i=1}^m u_i^\star h_i(\boldsymbol{x}^\star) + \sum_{j=1}^r v_i^\star \ell_i(\boldsymbol{x}^\star) \\ &= f(\boldsymbol{x}^\star), \end{aligned}$$

   where the 2nd equality follows from stationarity, and the 3rd one follows from complementary slackness. Recall (9.2). This equality suggests the duality gap is zero and $\boldsymbol{x}^\star$, $\boldsymbol{u}^\star$and $\boldsymbol{v}^\star$ are primal and dual optimal.
2. (Necessity) Suppose that the strong duality holds and that $\boldsymbol{x}^\star$ and $\boldsymbol{u}^\star, \boldsymbol{v}^\star$ are primal and dual solutions. One has

$$\begin{aligned} f(\boldsymbol{x}^\star) &= g(\boldsymbol{u}^\star, \boldsymbol{v}^\star) \\ &= \min_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) + \sum_{i=1}^m u_i^\star h_i(\boldsymbol{x}) + \sum_{j=1}^r v_j^\star \ell_j(\boldsymbol{x}) \\ &\le f(\boldsymbol{x}^\star) + \sum_{i=1}^m u_i^\star h_i(\boldsymbol{x}^\star) + \sum_{j=1}^r v_j^\star \ell_j(\boldsymbol{x}^\star) \\ &\le f(\boldsymbol{x}^\star). \end{aligned}$$

   In other words, all the inequalities are actually equalities. KKT conditions hold.

□



(a) Equality Constraints: $\partial f(x)$ is a linear combination of $\partial \ell_j(x)$'s.

(b) One Inequality Constraint: $\partial f(x) + u_1 \partial h_1(x) = \mathbf{0}$ and $h_1(x) = 0$.

(c) Inequality Constraints: $\partial f(x) + \sum_{i=1}^{2} u_i \partial h_i(x) = \mathbf{0}$, $h_1(x) = 0$, and $h_2(x) = 0$.

(d) Inequality Constraints: $\partial f(x) + \sum_{i=1}^{3} u_i \partial h_i(x) = \mathbf{0}$, $h_1(x) = 0$, $h_2(x) = 0$, and $h_3(x) < 0$.

Figure 9.4: Geometric Interpretation of KKT Conditions

9.4 Examples

Quadratic Programming with Equality Constraints

Consider a quadratic programming problem given by

$$\begin{aligned} \min_{x} \ & \frac{1}{2} x^{\mathsf{T}} A x + b^{\mathsf{T}} x \\ \text{s.t.} \ & Cx = d, \end{aligned}$$

where $A \succeq 0$.

The corresponding Lagrangian is given by

$$L(x, v) = \frac{1}{2} x^{\mathsf{T}} A x + b^{\mathsf{T}} x + \langle v, Cx - d \rangle.$$

By KKT conditions, x is the minimizer if and only if

$$\begin{bmatrix} A & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} -b \\ d \end{bmatrix},$$

where the first set of linear equations come from the stationarity and the second set follows from the primal feasibility. The optimal $x^\star$ can be obtained by solving the system of linear equations.

Water Filling Power Allocation for Wireless Communications

Suppose independent parallel channels in a wireless communication system. The corresponding channel capacity is given by

$$\sum_i \log\left(1+\frac{x_i}{\alpha_i}\right),$$

where $x_i \geq 0$ is the power allocated to the i-th channel and $\alpha_i > 0$ is the given noise power for the i-th channel. The task is to find the optimal power allocation to maximize the channel capacity under the total power constraint $\sum_i x_i = 1$.

Towards this goal, the following constrained convex optimization problem can be formulated.

$$\begin{aligned}\min_{\boldsymbol{x}} \; & -\sum_{i=1}^{n}\log\left(\alpha_i + x_i\right)\\ \text{s.t. } & \boldsymbol{x} \geq \mathbf{0},\ \mathbf{1}^T\boldsymbol{x} = 1.\end{aligned}$$

The corresponding Lagrangian is given by

$$L(\boldsymbol{x},\boldsymbol{u},v) = -\sum_{i=1}^{n}\log\left(\alpha_i + x_i\right) - \sum_i u_i x_i + v(\mathbf{1}^T\boldsymbol{x} - 1).$$

By KKT conditions, the optimal solution satisfies

- $-1/\left(\alpha_i + x_i^\star\right) - u_i^\star + v^\star = 0, \forall i.$
- $u_i^\star x_i^\star = 0, \forall i.$
- $\boldsymbol{x}^\star \geq 0, \mathbf{1}^T\boldsymbol{x}^\star = 1, \boldsymbol{u}^\star \geq 0.$

From the first two conditions, one has

$$\begin{aligned}&1/\left(\alpha_i + x_i^\star\right) \leq v^\star, \text{and}\\ &x_i^\star\left(v^\star - 1/\left(\alpha_i + x_i^\star\right)\right) = 0.\end{aligned}$$

Therefore, the optimal power allocation is given by

$$x_i^\star = \max\left(0, 1/v^\star - \alpha_i\right)$$

where $v^\star$ is chosen such that

$$\sum_{i=1}^{n}\max\left(0, 1/v^\star - \alpha_i\right) = 1.$$

Constrained ℓ_1-minimization

Consider the optimization problem

$$\begin{aligned}\min_{\boldsymbol{x}} \; & \|\boldsymbol{x}\|_1\\ \text{s.t. } & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}.\end{aligned}$$

The corresponding Lagrangian is given by

$$L(\boldsymbol{x}, \boldsymbol{v}) = \|\boldsymbol{x}\|_1 + \langle \boldsymbol{v}, \boldsymbol{A}\boldsymbol{x} - \boldsymbol{y} \rangle.$$

The corresponding Lagrange dual function is then given by

$$\begin{aligned} g(\boldsymbol{v}) &= \min_{\boldsymbol{x}} \ \|\boldsymbol{x}\|_1 + \langle \boldsymbol{v}, \boldsymbol{A}\boldsymbol{x} - \boldsymbol{y} \rangle \\ &= -\boldsymbol{v}^\mathsf{T}\boldsymbol{y} + \min_{\boldsymbol{x}} \ \|\boldsymbol{x}\|_1 + \langle \boldsymbol{A}^\mathsf{T}\boldsymbol{v}, \boldsymbol{x} \rangle. \end{aligned}$$

Note that

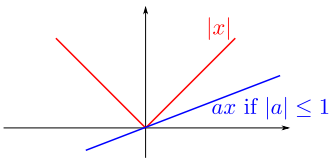
$$\min_{x_i} \ |x_i| + (\boldsymbol{A}^\mathsf{T}\boldsymbol{v})_i x_i = \begin{cases} 0 & \text{if } \left|(\boldsymbol{A}^\mathsf{T}\boldsymbol{v})_i\right| \leq 1 \\ -\infty & \text{if } \left|(\boldsymbol{A}^\mathsf{T}\boldsymbol{v})_i\right| > 1. \end{cases}$$

One has

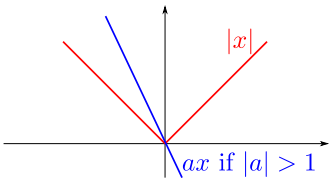
$$g(\boldsymbol{v}) = \begin{cases} -\boldsymbol{v}^\mathsf{T}\boldsymbol{y} & \text{if } \left\|\boldsymbol{A}^\mathsf{T}\boldsymbol{v}\right\|_\infty \leq 1 \\ -\infty & \text{if } \left\|\boldsymbol{A}^\mathsf{T}\boldsymbol{v}\right\|_\infty > 1 \end{cases}$$

The dual problem is given by

$$\begin{aligned} \max_{\boldsymbol{v}} \ & -\boldsymbol{v}^\mathsf{T}\boldsymbol{y} \\ \text{s.t.} \ & \left\|\boldsymbol{A}^\mathsf{T}\boldsymbol{v}\right\|_\infty \leq 1. \end{aligned}$$



(a) Case 1: $\min_x \ |x| + ax \ = 0$.



(b) Case 1: $\min_x \ |x| + ax \ = -\infty$.

Figure 9.5: An illustration of the dual function for constrained ℓ_1-minimization

10 ADMM

ADMM: Alternating Direction Method of Multipliers.

Key reference: Boyd, Stephen, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. "Distributed optimization and statistical learning via the alternating direction method of multipliers." Foundations and Trends in Machine learning 3, no. 1 (2011): 1-122.

10.1 Precursors 73
10.1.1 Dual Ascent Method . . . 73
10.1.2 Method of Multipliers . . 74
10.2 ADMM 75
10.2.1 Optimality Conditions . . 77

10.1 Precursors

10.1.1 Dual Ascent Method

Consider the convex optimization problem

$$\begin{aligned} &\min\ f(\boldsymbol{x}) \\ &\text{s.t. } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned} \tag{10.1}$$

Its Lagrangian is

$$L(\boldsymbol{x}, \boldsymbol{v}) = f(\boldsymbol{x}) + \boldsymbol{v}^{\mathsf{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}).$$

The dual function is given by

$$g(\boldsymbol{v}) = \inf_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{v}).$$

The dual problem is

$$\max\ g(\boldsymbol{v}).$$

Apply dual ascent method for the dual problem. One updates the dual variable via

$$\boldsymbol{v}^{k+1} = \boldsymbol{v}^k + \alpha^k \nabla g(\boldsymbol{v}^k).$$

From the definition of the dual function, one has

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \arg\min_{\boldsymbol{x}}\ L\left(\boldsymbol{x}, \boldsymbol{v}^k\right) \\ \boldsymbol{v}^{k+1} &= \boldsymbol{v}^k + \alpha^k \nabla_v L\left(\boldsymbol{x}^{k+1}, \boldsymbol{v}^k\right) \\ &= \boldsymbol{v}^k + \alpha^k\left(\boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{b}\right). \end{aligned}$$

With appropriate chosen α^k, $g\left(\boldsymbol{v}^{k+1}\right) > g\left(\boldsymbol{v}^k\right)$. Dual ascent method converges under certain assumptions.

However, the required assumptions do not hold in many applications. For example, when $f(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$, the $\boldsymbol{x}$-update step fails as Lagrangian L is unbounded below for some choices of $\boldsymbol{v}$.

10.1.2 Method of Multipliers

Instead of Lagrangian, we consider the augmented Lagrangian for (10.1) given by

$$L_\rho(\boldsymbol{x}, \boldsymbol{v}) = f(\boldsymbol{x}) + \boldsymbol{v}^\mathsf{T}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2,$$

where $\rho > 0$ is called penalty parameter.

Augmented Lagrangian can be viewed as Lagrangian of the problem

$$\begin{aligned} &\min f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 \\ &\text{s.t. } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \end{aligned}$$

which is equivalent to (10.1).

The associated dual function is given by

$$g_\rho(\boldsymbol{v}) = \inf_{\boldsymbol{x}} L_\rho(\boldsymbol{x}, \boldsymbol{v}).$$

Apply dual ascent method. The *method of multipliers* algorithm is given by

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \arg\min_{\boldsymbol{x}} L_\rho\left(\boldsymbol{x}, \boldsymbol{v}^k\right) \\ \boldsymbol{v}^{k+1} &= \boldsymbol{v}^k + \rho\left(\boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{b}\right). \end{aligned}$$

Note the fixed step size ρ.

The choice of the step size ρ can be motivated as follows. For simplicity, assume f is differentiable. The optimality conditions for (10.1) are primal and dual feasibility

$$\boldsymbol{A}\boldsymbol{x}^\star - \boldsymbol{b} = \boldsymbol{0}, \quad \nabla f\left(\boldsymbol{x}^\star\right) + \boldsymbol{A}^T\boldsymbol{v}^\star = \boldsymbol{0}.$$

As $\boldsymbol{x}^{k+1}$ minimizes $L_\rho\left(\boldsymbol{x}, \boldsymbol{v}^k\right)$, it holds that

$$\begin{aligned} \boldsymbol{0} &= \nabla_{\boldsymbol{x}} L_\rho\left(\boldsymbol{x}^{k+1}, \boldsymbol{v}^k\right) \\ &= \nabla_{\boldsymbol{x}} f\left(\boldsymbol{x}^{k+1}\right), + \boldsymbol{A}^T\left(\boldsymbol{v}^k + \rho\left(\boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{b}\right)\right) \\ &= \nabla_{\boldsymbol{x}} f\left(\boldsymbol{x}^{k+1}\right) + \boldsymbol{A}^T\boldsymbol{v}^{k+1}. \end{aligned}$$

Using ρ as step size, $\left(\boldsymbol{x}^{k+1}, \boldsymbol{v}^{k+1}\right)$ is dual feasible.

Compared with dual ascent method, the method of multipliers converges under much more general conditions. However, the $\boldsymbol{x}$-update step may be difficult to compute.

10.2 ADMM

ADMM solves problems in the form

$$\begin{aligned} &\min f(\boldsymbol{x}) + g(\boldsymbol{z}) \\ &\text{x.t. } \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c} \end{aligned}$$

The corresponding augmented Lagrangian is given by

$$\begin{aligned} L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{v}) &= f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{v}^\mathsf{T}(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}) \\ &\quad + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}\|_2^2 \\ &= f(\boldsymbol{x}) + g(\boldsymbol{z}) + \frac{\rho}{2}\left\|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c} + \boldsymbol{v}/\rho\right\|_2^2 \\ &\quad -\|\boldsymbol{v}\|_2^2/(2\rho) \end{aligned}$$

for some $\rho > 0$. ADMM is an iterative algorithm with iterations:

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \arg\min_{\boldsymbol{x}} L_\rho\left(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{v}^k\right), \\ \boldsymbol{z}^{k+1} &= \arg\min_{\boldsymbol{z}} L_\rho\left(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{v}^k\right), \\ \boldsymbol{v}^{k+1} &= \boldsymbol{v}^k + \rho\left(\boldsymbol{A}\boldsymbol{x}^{k+1} + \boldsymbol{B}\boldsymbol{z}^{k+1} - \boldsymbol{c}\right). \end{aligned}$$

Example 10.2.1 (Solving Lasso by ADMM) Consider the Lasso problem:

$$\min \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1.$$

An ADMM reformulation is given by

$$\begin{aligned} &\min \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{z}\|_1 \\ &\text{s.t. } \boldsymbol{x} = \boldsymbol{z}. \end{aligned}$$

The corresponding ADMM iterations are

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \frac{\rho}{2}\left\|\boldsymbol{x} - \boldsymbol{z}^k + \frac{\boldsymbol{v}^k}{\rho}\right\|_2^2 \\ \boldsymbol{z}^{k+1} &= \arg\min_{\boldsymbol{z}} \lambda\|\boldsymbol{z}\|_1 + \frac{\rho}{2}\left\|\boldsymbol{x}^{k+1} - \boldsymbol{z} + \frac{\boldsymbol{v}^k}{\rho}\right\|_2^2 \\ \boldsymbol{v}^{k+1} &= \boldsymbol{v}^k + \rho\left(\boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}\right). \end{aligned}$$

Remark 10.2.1

1. Each step of the above ADMM has a closed form solution.
2. ADMM is different from the method of multipliers. In the method

of multipliers, the variable update is

$$\begin{aligned}
&\left(\boldsymbol{x}^{k+1}, \boldsymbol{z}^{k+1}\right) \\
&= \arg\min_{\boldsymbol{x},\boldsymbol{z}} \; L_\rho\left(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{v}^k\right) \\
&= \arg\min_{\boldsymbol{x},\boldsymbol{z}} \; \frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \lambda\left\|\boldsymbol{z}\right\|_1 + \frac{\rho}{2}\left\|\boldsymbol{x}-\boldsymbol{z}+\frac{\boldsymbol{v}^k}{\rho}\right\|_2^2.
\end{aligned}$$

As $\boldsymbol{x}$ and $\boldsymbol{z}$ are not separate, there is no closed form for the variable update step.

3. The above ADMM converges for an arbitrary penalty constant $\rho > 0$. In ISTA (Section 7.2.3), the step size τ needs to be in the range $(0, 2/\sigma_{\max}(\boldsymbol{A}^\mathsf{T}\boldsymbol{A}))$ to guarantee a convergence.

Example 10.2.2 (Constrained Lasso Solved by ADMM) Consider the problem

$$\begin{aligned}
&\min \frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \lambda\left\|\boldsymbol{x}\right\|_1 \\
&\text{s.t. } \boldsymbol{B}\boldsymbol{x} \leq \boldsymbol{c}.
\end{aligned}$$

An ADMM reformulation is given by

$$\begin{aligned}
&\min \frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \lambda\left\|\boldsymbol{z}_1\right\|_1 + \delta_{\geq 0}\left(\boldsymbol{z}_2\right) \\
&\text{s.t. } \begin{bmatrix}\boldsymbol{I}\\ \boldsymbol{B}\end{bmatrix}\boldsymbol{x} + \begin{bmatrix}-\boldsymbol{I} & \\ & \boldsymbol{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{z}_1\\ \boldsymbol{z}_2\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{c}\end{bmatrix},
\end{aligned}$$

where

$$\delta_{\geq 0}(z) = \begin{cases} 0 & \text{if } z \geq 0, \\ \infty & \text{if } z < 0. \end{cases}$$

Write the above constraint as $\boldsymbol{B}'\boldsymbol{x} + \boldsymbol{D}'\boldsymbol{z} = \boldsymbol{c}'$. The ADMM iterations are

$$\begin{aligned}
\boldsymbol{x}^{k+1} &= \arg\min_{\boldsymbol{x}} \; \frac{1}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|_2^2 + \frac{\rho}{2}\left\|\boldsymbol{B}'\boldsymbol{x} + \boldsymbol{D}'\boldsymbol{z}^k - \boldsymbol{c}' + \frac{\boldsymbol{v}^k}{\rho}\right\|_2^2 \\
\boldsymbol{z}^{k+1} &= \arg\min_{\boldsymbol{z}_1,\boldsymbol{z}_2} \; \lambda\left\|\boldsymbol{z}_1\right\|_1 + \frac{\rho}{2}\left\|\boldsymbol{x}^{k+1} - \boldsymbol{z}_1 + \frac{\boldsymbol{v}_1^k}{\rho}\right\|_2^2 \\
&\qquad + \delta_{\geq 0}\left(\boldsymbol{z}_2\right) + \frac{\rho}{2}\left\|\boldsymbol{B}\boldsymbol{x}^{k+1} + \boldsymbol{z}_2 - \boldsymbol{c} + \frac{\boldsymbol{v}_2^k}{\rho}\right\|_2^2 \\
\boldsymbol{v}^{k+1} &= \boldsymbol{v}^k + \rho\left(\boldsymbol{B}'\boldsymbol{x}^{k+1} + \boldsymbol{D}'\boldsymbol{z}^{k+1} - \boldsymbol{c}'\right).
\end{aligned}$$

Each step admits a closed form.

10.2.1 Optimality Conditions

The necessary and sufficient conditions for optimality are primal feasibility

$$\boldsymbol{A}\boldsymbol{x}^\star + \boldsymbol{B}\boldsymbol{z}^\star - \boldsymbol{c} = \boldsymbol{0}. \tag{10.2}$$

and dual feasibility

$$\boldsymbol{0} \in \partial f\left(\boldsymbol{x}^\star\right) + \boldsymbol{A}^T\boldsymbol{v}^\star \tag{10.3}$$
$$\boldsymbol{0} \in \partial g\left(\boldsymbol{z}^\star\right) + \boldsymbol{B}^T\boldsymbol{v}^\star. \tag{10.4}$$

1. Consider primal feasibility. Define *primal residual* as

   $$r^{k+1} = \boldsymbol{A}\boldsymbol{x}^{k+1} + \boldsymbol{B}\boldsymbol{z}^{k+1} - \boldsymbol{c}.$$

   We expect r^k converges to zero.
2. Consider (10.4) in dual feasibility. It turns out that $\boldsymbol{z}^{k+1}$ and $\boldsymbol{v}^{k+1}$ always satisfy (10.4):

   $$\begin{aligned}\boldsymbol{0} &\in \partial g\left(\boldsymbol{z}^{k+1}\right) + \boldsymbol{B}^T\boldsymbol{v}^k + \rho\boldsymbol{B}^T\left(\boldsymbol{A}\boldsymbol{x}^{k+1} + \boldsymbol{B}\boldsymbol{z}^{k+1} - \boldsymbol{c}\right)\\ &= \partial g\left(\boldsymbol{z}^{k+1}\right) + \boldsymbol{B}^T\boldsymbol{v}^{k+1}.\end{aligned}$$

3. Consider (10.3) in dual feasibility. The situation about $\boldsymbol{x}^{k+1}$ is different. By definition $\boldsymbol{x}^{k+1}$ minimizes $L_\rho\left(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{v}^k\right)$. It holds that

   $$\begin{aligned}\boldsymbol{0} &\in \partial f\left(\boldsymbol{x}^{k+1}\right) + \boldsymbol{A}^T\boldsymbol{v}^k + \rho\boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{x}^{k+1} + \boldsymbol{B}\boldsymbol{z}^k - \boldsymbol{c}\right)\\ &= \partial f\left(\boldsymbol{x}^{k+1}\right) + \boldsymbol{A}^T\left(\boldsymbol{v}^k + \rho\boldsymbol{r}^{k+1} + \rho\boldsymbol{B}\left(\boldsymbol{z}^k - \boldsymbol{z}^{k+1}\right)\right)\\ &= \partial f\left(\boldsymbol{x}^{k+1}\right) + \boldsymbol{A}^T\boldsymbol{v}^{k+1} + \rho\boldsymbol{A}^T\boldsymbol{B}\left(\boldsymbol{z}^k - \boldsymbol{z}^{k+1}\right).\end{aligned}$$

   Or equivalently

   $$\rho\boldsymbol{A}^T\boldsymbol{B}\left(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k\right) \in \partial f\left(\boldsymbol{x}^{k+1}\right) + \boldsymbol{A}^T\boldsymbol{v}^{k+1}.$$

   Define *dual residual* as

   $$\boldsymbol{s}^{k+1} = \rho\boldsymbol{A}^T\boldsymbol{B}\left(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k\right).$$

Under mild conditions, ADMM converges:

- Primal residual convergence: $\boldsymbol{r}^k \to \boldsymbol{0}$ as $k \to \infty$.
- Objective convergence: $f\left(\boldsymbol{x}^k\right) + g\left(\boldsymbol{z}^k\right) \to \boldsymbol{p}^\star$ as $k \to \infty$.
- Dual variable convergence: $\boldsymbol{v}^k \to \boldsymbol{v}^\star$ as $k \to \infty$.

In practice, a reasonable criterion of terminating ADMM iterations is that the primal and dual residuals are small, i.e.,

$$\left\|\boldsymbol{r}^k\right\|_2 \le \epsilon^{\text{pri}}, \qquad \left\|\boldsymbol{s}^k\right\|_2 \le \epsilon^{\text{dual}}.$$

11 Stochastic Optimization

11.1 Motivation 78
11.2 Stochastic Least Squares . 78
11.2.1 Gradient Descent for LS . 78
11.2.2 Stochastic Gradient Descent for LS 79

11.1 Motivation

Throughout these notes, we have encountered a wide range of inference and recovery problems, all of which could be formulated as optimization problems. In the case of least squares estimation, treated in Chapter 3, the optimal solution to the optimization problem could be found in terms of a closed-form solution. In later chapters, as the optimization problems became more elaborate, such as for example in case of Lasso (treated in Chapter 7), the optimal solution was no longer available in closed-form. Instead, we pursued optimal solutions through iterative means, such as the Iterative Shrinkage Thresholding Algorithm (ISTA). We repeat the problems and solutions here for reference:

$$\hat{x} = \arg\min_x \|y - Ax\|_2^2 \Longrightarrow \hat{x} = \left(A^T A\right)^{-1} A^T y \tag{11.1}$$

$$\hat{x} = \arg\min_x \|y - Ax\|_2^2 + \lambda\|x\|_1 \Longrightarrow x^{l+1} = \text{Prox}_{\gamma\lambda\|\cdot\|_1}\left(x^l + \gamma A^T\left(y - Ax^l\right)\right) \tag{11.2}$$

In both cases, the expressions were *batch* solutions, meaning that the full batch of data, contained in the vector y and matrix A are leveraged either to compute the closed-form solution (11.1), or at every iteration (11.2), when evaluating the ISTA update. This is feasible when the amount of data, either in terms of problem dimension (i.e., the number of columns of A), or in terms of available samples (number of rows of A) is moderate when compared to the computational and storage resources. In many large-scale settings, the available data grows faster than computational resources, eventually reaching the point where we are no longer able to process the entire batch of data at once.

Motivated by these considerations, in this chapter, we resort to *stochastic* optimization techniques, which process smaller *mini-batches* of data at a time. We begin by deriving a stochastic version of least optimization, and will subsequently generalize to allow for arbitrary loss functions.

11.2 Stochastic Least Squares

11.2.1 Gradient Descent for LS

Let us revisit the least squares problem (11.1) and derive an alternative algorithm for pursuing its minimizer. We recall that the squared Euclidean norm of a vector is given by the sum of its elements squared, yielding the alternative expression:

$$\hat{x} = \arg\min_x \|y - Ax\|_2^2 = \arg\min_x \sum_{m=1}^{M} (y_m - a_m^T x)^2 \tag{11.3}$$

where we denote by a_m the *column* vector corresponding to the m-th row of the data matrix A. The minimizing argument in (11.3) is invariant to scaling of the cost, and hence we can normalize equivalently:

$$\hat{x} = \arg\min_x \frac{1}{M} \sum_{m=1}^{M} (y_m - a_m^T x)^2 \tag{11.4}$$

While a direct closed-form expression for $\hat{x}$ is available via (11.1), it will be instructive to derive a gradient based iterative algorithm and compare the computational cost. Using a constant step-size α (i.e., no line search), we find:

$$x^{l+1} = x^l - \alpha \nabla \left(\frac{1}{M} \sum_{m=1}^{M} (y_m - a_m^T x)^2 \right) = x^l + \frac{\alpha}{M} \sum_{m=1}^{M} a_m (y_m - a_m^T x) \tag{11.5}$$

Example 11.2.1 (Per-iteration complexity of Least-Squares Gradient Descent) We note that computing the gradient update in (11.5) involves the averaging of M smaller operations of the form $a_m(y_m - a_m^T x)$. Disregarding additions, evaluating $y_m - a_m^T x$ requires N multiplications, where N denotes the dimension of the feature space (and hence also of the parameter vector x). To find $a_m(y_m - a_m^T x)$, we require N more multiplications. All this is repeated M times to find the gradient $\frac{1}{M} \sum_{m=1}^{M} a_m(y_m - a_m^T x)$. In summary, we find:

$$\text{Per-iteration complexity of LS GD} \sim MN \tag{11.6}$$

We observe that the complexity of evaluating a single step of gradient descent, applied to the least mean squares cost (11.4) scales linearly in both the dimension N, and the size of the data set M. Of course this operation needs to be repeated a total of L times, where L denotes the total number of iterations. Hence:

$$\text{Total complexity of LS GD} \sim MNL \tag{11.7}$$

A natural question given the above consideration is then the following: Will the complexity (both per iteration, and in total) of a learning algorithm always scale at least linearly with the number of available samples? In either case, the answer would have far-reaching implications for large-scale data processing. Perhaps surprisingly, the answer is no. We will explore this in the context of least squares estimation first, and subsequently generalize.

11.2.2 Stochastic Gradient Descent for LS

A common strategy in engineering and statistics for dealing with intractable quantities of data is to sample subsets of the data and compute stochastic estimates of the quantity of interest. In the context of least

squares gradient descent, the quantity of interest is the gradient:

$$\nabla f(x) = \frac{1}{M} \sum_{m=1}^{M} a_m (y_m - a_m^T x) \tag{11.8}$$

Suppose now, at every iteration instead of evaluating the full gradient $\nabla f(x)$ by averaging M gradients of the form $a_m(y_m - a_m^T x)$, we sub-sample the total data set $\{x_m, y_m\}_{m=1}^{M}$. In particular, we will sample B times from $\{x_m, y_m\}_{m=1}^{M}$, obtaining the *mini-batch* $\{x_b^t, y_b^t\}_{b=1}^{B}$.