Optimization for Data Science

F. Rinaldi[1]

1 Università degli Studi di Padova | Dipartimento Matematica

Padova
2020

Outline

Optimization for Data Science

Interior Point (IP) Methods: Then and Now

- Class of approaches for solving linear and nonlinear programming problems.
- First IP algorithm proposed by Karmarkar (1984).
- In his seminal paper the author described a polynomial time algorithm for solving linear programs and made some strong claims about its performance in practice.
- The algorithm was controversial at the time of its introduction, but there have been many improvements both in theory and practice since then.
- Interior point methods are now considered to be better than simplex, especially on large LPs.
- The method, as we will see, can be used to solve general convex programs.

The Problem under Analysis

Problem Formulation

We consider problems having the following form:

$$\begin{aligned}&\min c^\top x\\&x \in C\end{aligned} \tag{1}$$

where C is a compact convex set with non-empty interior.

We notice that any convex problem

$$\begin{aligned}&\min f(x)\\&x \in C\end{aligned} \tag{2}$$

can be reformulated like Problem (1) by minimizing a linear function over the epigraph of the original objective f:

$$\begin{aligned}&\min_{x,y} y\\&x \in C\\&f(x) \leq y.\end{aligned} \tag{3}$$

The Barrier

- In interior point methods, we add to the original objective function a term $B(x)$ defined in the interior of C.
- This function, usually called *barrier function*, is continuous and tends to $+\infty$ as the point approaches the boundary of the feasible set.

Assumptions on Barrier Term

- the barrier should map the interior of the feasible space to the real space, that is $B(x) : int(C) \to \mathbb{R}$;
- since we include the barrier into the objective function, we need it to be smooth, i.e., continuously differentiable;
- it should be such that
$$B(x) \xrightarrow[x \to \partial C]{} +\infty;$$
- Hessian of B should be positive definite for each x.

Types of Barrier

- We consider a general feasible set of the form

$$C = \{x \in \mathbb{R}^n : \; g_i(x) \leq 0, \; i = 1, \ldots, p\},$$

  with $g_i(x), \; i = 1, \ldots, p$ convex functions.
- All functions we list are convex and satisfy the assumptions.

Logarithmic barrier

$$B(x) = -\sum_{i=1}^{p} \ln(-g_i(x)).$$

Inverse barrier

$$B(x) = -\sum_{i=1}^{p} \frac{1}{g_i(x)}.$$

A Method that uses Barriers

Barrier Method

it basically solves a sequence of problems of the following form

$$\min_{x \in \mathbb{R}^n} \; p\, c^\top x + B(x), \tag{4}$$

with $p \in \mathbb{R}^+$. The solution of this problem is usually indicated with $x^*(p)$.

- It is possible to prove that when $p \to +\infty$, we get

  $$x^*(p) \to x^*,$$

  with x^* optimal solution of the original problem.
- The sequence of $\{x^*(p)\}$ is usually called *central path*.
- IDEA: Move along the central path by “boosting” a fast locally convergent algorithm (denoted by $\mathcal{A}$).

Basic Scheme

- At each iteration k: choose a value p_k and use $\mathcal{A}$ initialized at $x^*(p_{k-1})$ to compute $x^*(p_k)$.
- The choice of p_k is crucial.
- p_k should be large in order to make as much progress as possible on the central path.
- The new point needs to be close enough to previous point in the central path so that it is in the basin of fast convergence for $\mathcal{A}$ when solving the problem with respect to p_k.

General Scheme

Algorithm 1 `General barrier method`

Choose a point $x_0 \in int(C)$ and $p_0 > 0$

For $k = 0, \ldots$

  Use $\mathcal{A}$ with starting point x_k to get

$$x^*(p_k) \in \underset{x \in \mathbb{R}^n}{\text{Argmin}} \; p_k\, c^\top x + B(x)$$

  If $x^*(p_k)$ satisfies some specific condition, then STOP

  Set $x_{k+1} = x^*(p_k)$

  Update p_{k+1}

End for

A Simple Barrier Method

- Since the barrier is defined only in the interior, the central path must be in the interior of the feasible set.
- In order to solve Barrier Problem (4), we can use Newton method.
- A basic interior-point approach is thus obtained by means of the *path-following scheme*.

Path-following scheme

Alternates between an updating of p_k and the calculation of an approximate solution of problem (4) by means of a single Newton step:

$$x_{k+1} = x_k - \nabla^2 B(x_k)^{-1}[p_k c + \nabla B(x_k)].$$

A classic updating rule for p_k (guaranteeing $p_k \to +\infty$) is

$$p_{k+1} = \left(1 + \frac{1}{13\sqrt{\nu}}\right) p_k,$$

with ν parameter depending on the barrier used.

Complexity of the Method

Theorem

The path-following scheme satisfies

$$c^\top x_k - c^\top x^* \leq \frac{2\nu}{p_0} e^{-\frac{k}{1+13\sqrt{\nu}}}.$$

- It is easy to see, by simple calculations, that computational complexity is $\mathcal{O}(M\sqrt{\nu}\ln(\nu/\varepsilon))$, where M is the cost of one Newton step and ν is some data-dependent scale factor.
- Cost of one Newton step is usually around $\mathcal{O}(n^2)$, with n dimension of the problem.
- In case C is polyhedral, it is possible to show that $\nu = n$.
- We thus have a polynomial time algorithm for linear programs
- REMARK: Interior-point methods guarantee better computational complexity than the the simplex!!!

Comparison with Information-based Approaches

- In the information-based complexity theory problem represented by an oracle (a black box)...no information on the instance!
- Information collected via sequential calls to the oracle, and the number of these calls sufficient to find an ε-solution basically represents the complexity of the method
- The information-based complexity does include neither the computational effort of the oracle, nor the arithmetic cost of processing the answers of the oracle by the method.
- Interior point methods from the very beginning possess *complete global information* (the data specifying the problem instance)
- Interior point is not as general as information-based methods!!!

Comments II

- When dealing with huge scale data, n is quite large!
- Operations like Hessian evaluation cannot be performed.
- Furthermore, the rate of the method described here depends on ν (which is related to the dimension of the problem).
- If we want to solve huge scale convex problems in data science, better choose *dimension-free* methods (i.e., methods whose rate does not depend on the input dimension n) that only use first order information.

Training of linear SVMs for Classification Problems

- Support Vector Machines (SVMs), first described by Vapnik in the Nineties.
- SVMs represent a very important class of machine learning tools.
- IDEA: When dealing with binary classification problems, build a hyperplane that maximizes the *separation margin* between the elements of the two classes.
- We then consider two sets A and B and assume that are linearly separable, i.e., there exists a hyperplane

$$H = \{x \in R^n : \; w^\top x + \theta = 0\}$$

such that

$$\begin{cases} w^\top x^i + \theta \geq 1 & x^i \in A \\ w^\top x^i + \theta \leq -1 & x^i \in B. \end{cases} \tag{5}$$

Separation Margin

Projection of a point x_0 over a hyperplane $H = \{x \in \mathbb{R}^n : \; a^\top x = b\}$

We have

$$\rho_H(x_0) = x_0 + \frac{b - a^\top x_0}{a^\top a} a$$

and

$$\|\rho_H(x_0) - x_0\| = \frac{|b - a^\top x_0|}{\|a\|}.$$

Separation Margin

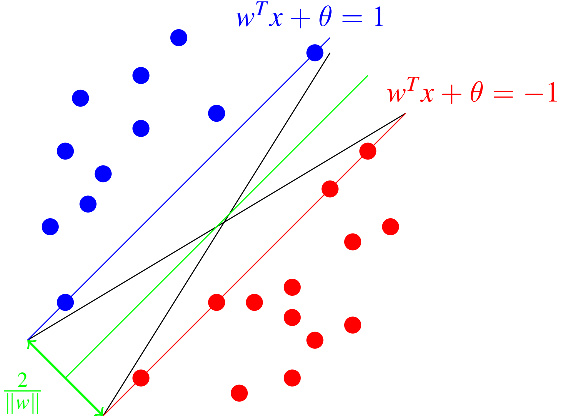
For a given hyperplane H, pair (w, θ) satisfying (5), we define as separation margin for H the minimum distance ρ between samples in $A \cup B$ and H:

$$\rho(w, \theta) = \min_{x^i \in A \cup B} \left\{ \frac{|w^\top x^i + \theta|}{\|w\|} \right\}.$$

Optimal Hyperplane

The optimal hyperplane $H(w^*, \theta^*)$ is the one with maximum separation margin.

Optimal Separating Hyperplane



Example of an Optimal Hyperplane

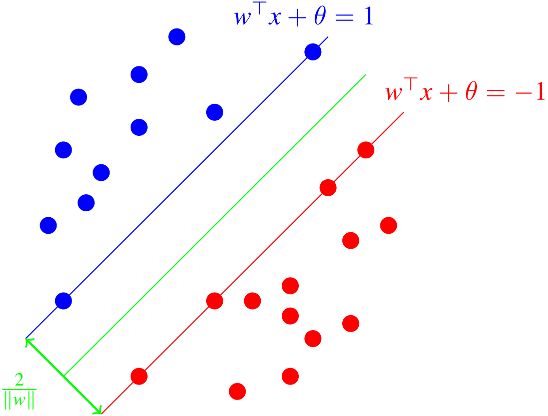


Figure: Optimal hyperplane for a classification problem.

Finding the Optimal Hyperplane

- It is possible to prove existence and uniqueness of the optimal hyperplane.
- Determining the optimal hyperplane is equivalent to solve the following problem:

$$
\begin{array}{ll}
\max\limits_{w,\ \theta} \rho(w, \theta) & \\
w^\top x^i + \theta \geq 1 & x^i \in A \\
w^\top x^i + \theta \leq -1 & x^i \in B.
\end{array}
$$

- It is possible to prove that this problem is equivalent to the following convex quadratic programming problem:

$$
\begin{array}{ll}
\min\limits_{w,\ \theta} \dfrac{1}{2}\|w\|^2 & \\
y^i(w^\top x^i + \theta) - 1 \geq 0 & i = 1, \ldots, P.
\end{array}
$$

with $y^i = 1$ if $x^i \in A$, $y^i = -1$ if $x^i \in B$ and $P = |A \cup B|$.

The Linearly Separable Case

SVM Training (Linearly Separable Case)

$$\begin{aligned} &\min_{w,\ \theta} \frac{1}{2}\|w\|^2 \\ &y^i(w^\top x^i + \theta) - 1 \geq 0 \qquad i = 1, \dots, P. \end{aligned}$$

with $y^i = 1$ if $x^i \in A$, $y^i = -1$ if $x^i \in B$ and $P = |A \cup B|$.

A More General Case

- When A and B are not linearly separable, the system (5) has no solutions.
- We thus introduce new variables $\xi^i \geq 0,\ i = 1, \dots, P$ and the system becomes:

$$\begin{cases} w^\top x^i + \theta \geq 1 - \xi^i & x^i \in A \\ w^\top x^i + \theta \leq -1 + \xi^i & x^i \in B \\ \xi^i \geq 0, \quad i = 1, \dots, P. \end{cases} \tag{6}$$

- Notice that when x^i is not correctly classified, variable ξ^i is greater than 1, thus the term

$$\sum_{i=1}^{P} \xi^i$$

is an upper bound over the training errors.

The Final Problem

SVM Training

$$
\begin{aligned}
&\min_{w,\ \theta,\ \xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{P}\xi^i \\
&y^i(w^\top x^i + \theta) - 1 \geq -\xi^i \qquad && i = 1,\ldots,P \\
&\xi^i \geq 0 && i = 1,\ldots,P.
\end{aligned}
$$

with $y^i = 1$ if $x^i \in A$, $y^i = -1$ if $x^i \in B$ and $P = |A \cup B|$.

SVMs for Regression Problems

- Consider the linear model
$$f(x; w, \theta) = w^\top x + \theta.$$
- Fix precision level ϵ, used to approximate the unknown function.
- Estimate is considered correct if the following condition is satisfied:
$$|y^i - w^\top x^i - \theta| \leq \epsilon.$$

- We use the loss function
$$|y - f(x; w, \theta)|_\epsilon = \max\{0, |y - f(x; w, \theta)| - \epsilon\}$$
and define the training error as follows (keep in mind that in this case $y^i \in \mathbb{R}$ for $i = 1, \dots, P$):
$$E = \sum_{i=1}^{P} |y^i - f(x^i; w, \theta)|_\epsilon.$$

Final Formulation of the Problem

- Training error is zero if the following system is satisfied:

$$\begin{cases} w^\top x^i + \theta - y^i \le \epsilon & i = 1, \dots, P \\ y^i - w^\top x^i - \theta \le \epsilon & i = 1, \dots, P. \end{cases} \tag{7}$$

SVM Training (Regression Problems)

It is possible to introduce new variables ξ^i and $\hat{\xi}^i$ and consider the following convex quadratic problem:

$$\begin{aligned} &\min_{w,\,\theta,\,\xi,\,\hat{\xi}} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{P}(\xi^i + \hat{\xi}^i) && \\ &w^\top x^i + \theta - y^i \le \epsilon + \xi^i && i = 1, \dots, P \\ &y^i - w^\top x^i - \theta \le \epsilon + \hat{\xi}^i && i = 1, \dots, P \\ &\xi^i, \hat{\xi}^i \ge 0 && i = 1, \dots, P. \end{aligned}$$

LP Problems with Random Costs

- Consider the linear program

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & c^\top x \\ \text{s.t.} \quad & Gx \le h \\ & Ax = b, \end{aligned}$$

- Assume that the cost vector c is a random vector with mean $\bar{c}$ and covariance

$$\mathbf{E}\left[(c-\bar{c})(c-\bar{c})^\top\right] = \Sigma.$$

- Assume that all the other parameters are deterministic.
- For a vector $x \in R^n$, cost $c^\top x$ is a random variable with mean

$$\mathbf{E}[c^\top x] = \bar{c}^\top x$$

and variance

$$\mathbf{var}(c^\top x) = \mathbf{E}[c^\top x - \mathbf{E}[c^\top x]]^2 = x^\top \Sigma x.$$

Problem Formulation

- Usually, we try to balance between expected value and variance.
- A classic model is given by combining the two terms:

$$\mathbf{E}[c^\top x] + \gamma \mathbf{var}(c^\top x),$$

with $\gamma \geq 0$ *risk-aversion* parameter.

- Keeping in mind that Σ is a positive semidefinite matrix, we get a convex quadratic program:

$$\begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & \bar{c}^\top x + \gamma x^\top \Sigma x \\ \text{s.t.} & Gx \leq h \\ & Ax = b. \end{array}$$

Portfolio optimization

- We have n available assets.
- We call x_i the quantity of money invested on the i-th asset during the considered period and with r_i the returns on the i-th asset.
- We have two different constraints:
  - Non-negativity for the variables (i.e., $x_i \geq 0$). Meaning that short selling (selling asset that we still don't own) is not allowed.
  - Budget constraint:

$$\sum_{i=1}^{n} x_i = B,$$

    the total amount of money invested needs to be equal to the budget B (B can be simply set to 1).

Stochastic Model for the Returns and Markowitz Model

- $r \in \mathbb{R}^n$ is a randomly generated vector with mean $\bar{r}$ and covariance Σ.
- Expected return will be
$$\bar{r}^\top x$$
and variance
$$x^\top \Sigma x.$$
- Classic portfolio problem, described by Markowitz (1952), is a convex quadratic programming problem:
$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \gamma x^\top \Sigma x - \bar{r}^\top x \\ s.t. \quad & e^\top x = 1 \\ & x \geq 0, \end{aligned}$$
with $\gamma > 0$ risk-aversion parameter.
- GOAL: Finding the set of assets that minimizes the variance (risk connected to the given portfolio) while maximizing the expected return (we obviously need to satisfy budget and non-negativity constraints).

PageRank

- The effectiveness of Google search engine largely relies (or used to rely) on its PageRank (named after Google's founder Larry Page) algorithm.
- IDEA: Quantitatively rank the importance of each page on the web.
- PageRank allows Google to present to the user the more important (and typically most relevant and helpful) pages first.

Web as a Direct Graph

- Consider the web of interest composed of n pages (each labelled with $k = 1, \ldots, n$).
- We can model this web as a directed graph.
- Pages are the nodes of the graph, and a directed edge exists pointing from node k_i to node k_j if the web page k_i contains a link to k_j.
- We call x_k, with $k = 1, \ldots, n$, the importance score of page k.
- A simple initial idea would be to assign the score to any page k according to the number of other web pages that link to the considered page (the so-called backlinks).

Example of Web

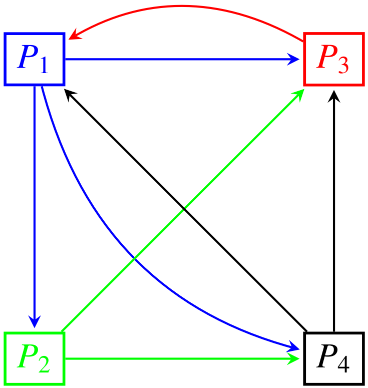


Figure: Small web example.

Scores in the Example

- In the example, the scores would be $x_1 = 2$, $x_2 = 1$, $x_3 = 3$, $x_4 = 2$.
- Page $k = 3$ appears to be the most relevant page, whereas page $k = 2$ is the least important.
- Using this approach, a page score can be interpreted as the number of "votes" that a page receives from other pages, where each incoming link is a vote.
- Is the web that simple? Of course not! :-)
- Relevance of a page typically depends on the relevance of the pages pointing to it.
- In other words, your page relevance should be higher if your page is pointed directly by Google.com rather than by Nobodycares.com.

Weighted Votes

- Votes should thus be weighted (not simply counted), and the weight should be related to the score of the pointing page itself.
- The actual scoring count goes then as follows: each page j has a score x_j and n_j outgoing links.
- As an assumption, we do not allow links from a page to itself, and we do not allow pages without outgoing links, therefore $n_j > 0$ for all j.
- The score x_j represents the total power of node j, which we need to split among the n_j outgoing links.
- Each outgoing link thus carries x_j/n_j units of vote.
- Let L_k denote the indices related to pages that point to page k (backlinks).
- Then, the score of page k is computed as follows:

$$\sum_{j \in L_k} \frac{x_j}{n_j}, \quad j = 1, \ldots, n.$$

Equations for the Small Web

- If we write this equations for our small web, we have

$$\begin{cases} x_1 = & & & x_3 + \frac{1}{2}x_4 \\ x_2 = \frac{1}{3}x_1 & & & \\ x_3 = \frac{1}{3}x_1 + \frac{1}{2}x_2 + & & & \frac{1}{2}x_4 \\ x_4 = \frac{1}{3}x_1 + \frac{1}{3}x_2 & & & \end{cases}$$

that we can be rewritten as follows

$$x = Ax,$$

with

$$A = \begin{pmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{pmatrix}.$$

PageRank Problem

- *Left stochastic matrix:* All components of the matrix are non-negative and each column sums up to 1.
- Matrix A, which is a *left stochastic matrix*, is the *link matrix* for the given web.
- It is actually possible to prove that 1 is eigenvalue for A and there exists an eigenvector with all components greater or equal than zero.
- Solving the PageRank problem corresponds to find the x that satisfies $x = Ax$, i.e., the eigenvector related to the eigenvalue 1.
- In our example a possible solution x is

$$x = \begin{pmatrix} 0.3871 \\ 0.1290 \\ 0.2903 \\ 0.1935 \end{pmatrix}.$$

- Page 1 is then the most relevant in our web.

Drawback of the Approach and use of the Google Matrix

- A problem we have when using this approach is that there might be multiple eigenvectors related to 1.
- Hence we consider a modified matrix

$$\hat{A} = (1 - \alpha)A + \alpha C,$$

where $\alpha \in (0, 1)$ and C is an $n \times n$ matrix with all entries equal to $1/n$.

- Classic choice is $\alpha = 0.15$.
- This modified matrix (usually called *Google matrix*) has only one eigenvector with corresponding eigenvalue equal to 1!

Meaning of C

C gives a surfer probability to jump randomly on any page in the web.

PageRank Formulation

A Useful Theoretical Result

It is possible to prove that the eigenvector x related to eigenvalue 1 satisfies the following conditions:

- x is unique;
- $x \geq 0$;
- $e^\top x = 1$.

- We can model PageRank as a constrained optimization problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \|Ax - x\|^2 \\ \text{s.t.} \quad & e^\top x = 1 \\ & x \geq 0. \end{aligned}$$

- An alternative unconstrained formulation, often used in practice, is the following:

$$\min_{x \in \mathbb{R}^n} \|Ax - x\|^2 + \gamma [e^\top x - 1]^2$$

with $\gamma > 0$ penalty parameter. In practice we add a term to the objective function that measures the violation of the equality constraint. Non-negativity constraints are simply dropped out in this case.

LASSO Problem

LASSO [Tibshirani, 1996]

Given the training set

$$T = \{(a^i, b^i),\ a^i \in \mathbb{R}^n,\ b^i \in \mathbb{R},\ i = 1, \ldots, m\}$$

the goal is finding a sparse linear model (i.e., a model with a small number of non-zero parameters) describing the data.

- This is a popular tool for sparse linear regression.
- This problem is strictly connected with the Basis Pursuit Denoising (BPDN) Problem in signal analysis.
- In this case, given a discrete-time input signal b, and a *dictionary*

  $$D = \{a_j \in \mathbb{R}^m \ : \ j = 1, \ldots, n\}$$

  of elementary discrete-time signals, usually called atoms, the goal is finding a sparse linear combination of the atoms that *approximate* the real signal.
- REMARK: It is just the approximated version of the compressive sensing problem!!!

Image Compression Example

- We have a real signal $\tilde{x}$ that has a sparse representation for a given $n \times n$ basis Ψ, that is
$$\tilde{x} = \Psi x$$
and x sparse.
- We call $b \in \mathbb{R}^m$ the measurement vector (compressed signal).
- This compressed signal is simply obtained by means of Φ $m \times n$ the so-called sampling matrix:
$$b = \Phi \tilde{x}.$$
- We then define a *reconstruction matrix* by multiplying Φ and Ψ, that is $A = \Phi\Psi$.
- As we will see, the available reconstruction tools use b and A to get the sparse representation x.

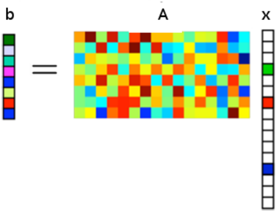


Figure: Image compression example.

LASSO/BPDN Formulation

Problem Formulation

LASSO/BPD problem can be formulated as follows:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \|Ax - b\|_2^2 \\ s.t. \quad & |supp(x)| \leq \tau \end{aligned} \tag{8}$$

The parameter τ controls the amount of shrinkage that is applied to the model (number of nonzero components in x).

- Constraint is represented by means of a nonconvex and discontinuous function.
- We then need to use an approximation to make problem tractable.
- Using the same idea already seen for the compressive sensing problem, we formulate LASSO/BPD problem as follows:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \|Ax - b\|_2^2 \\ s.t. \quad & \|x\|_1 \leq \tau. \end{aligned} \tag{9}$$

Thus we get a convex problem that can be easily handled in practice.