Optimization for machine learning

Constrained Optimization and Standard Optimization problems

R. Flamary

March 5, 2020

Full course overview

Course overview

Constraints and Lagrangian

Optimization problem

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

- F is convex and differentiable, h_j and g_i are differentiable and define convex constraints.

Lagragian of the optimization problem

We define the Lagrangian of the problem the function $\mathcal{L}$ such that :

$$\mathcal{L}(\mathbf{x},\mathbf{u},\mathbf{v}) = F(\mathbf{x}) + \sum_{i=1}^{k} u_i g_i(\mathbf{x}) + \sum_{j=1}^{m} v_j h_j(\mathbf{x}) \tag{2}$$

where $\mathbf{u}\in\mathbb{R}^k$ and $\mathbf{v}\in\mathbb{R}^m$ are the Lagrange multipliers of dual variables, with $u_i \geq 0$ (positivity constraints).

Lagrange dual function

Lagrange dual function

The Lagrange dual function D of the problem is

$$D(\mathbf{u}, \mathbf{v}) = \inf_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \mathbf{u}, \mathbf{v}) \tag{3}$$

- If F is not bounded below, $D = -\infty$.
- D is always concave (even when F is non-convex)

Lower bound

For all $\mathbf{u} \geq 0, \mathbf{v}$ and feasible $\mathbf{x}$ we have

$$F(\mathbf{x}) \geq \mathcal{L}(\mathbf{x}, \mathbf{u}, \mathbf{v}) \geq D(\mathbf{u}, \mathbf{v})$$

Proof:

$$\mathcal{L}(\mathbf{x}, \mathbf{u}, \mathbf{v}) = F(\mathbf{x}) + \sum_{i=1}^{k} \underbrace{u_i g_i(\mathbf{x})}_{\leq 0} + \sum_{j=1}^{m} \underbrace{v_j h_j(\mathbf{x})}_{=0} \leq F(\mathbf{x})$$

because $\mathbf{x}$ feasible $(g_i(\mathbf{x}) \leq 0, h_j(\mathbf{x}) = 0)$ and $\mathbf{u} \geq 0$.

Exercise 1: Lagrange dual

$$\min_{x, x \geq 0} \quad F(x) = (x-1)^2$$

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, u) =$$

2. Solve the infimum *w.r.t.* x for a given dual variable u:

$$x^\star =$$

3. Express the Lagrange Dual function $D(u)$:

$$D(u) =$$

Check that the function is concave

Lagrange Duality

Primal problem

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

- Optimal value $F^\star = F(\mathbf{x}^\star)$.

Dual problem

$$\begin{array}{ll} \max\limits_{\mathbf{u} \in \mathbb{R}^q, \mathbf{v} \in \mathbb{R}^p} & D(\mathbf{u}, \mathbf{v}) \\ \text{with} & \mathbf{u} \geq \mathbf{0} \end{array}$$

- The dual function is a lower bound on the optimal value of the primal.
- The dual problem is always convex.
- If an optimal value $D^\star$ is reached then we have what is called weak duality with

$$F^\star \geq D^\star$$

Duality Gap and Strong duality

Definition

For a feasible primal variable $\mathbf{x}$ and feasible dual variables $\mathbf{u}, \mathbf{v}$ we call **duality gap** the following positive value

$$F(\mathbf{x}) - D(\mathbf{u}, \mathbf{v}) \geq 0 \tag{4}$$

- One property of the duality gap is that

$$F(\mathbf{x}) - F^\star \leq F(\mathbf{x}) - D(\mathbf{u}, \mathbf{v})$$

- If the duality gap is 0 for a feasible triplet $\mathbf{x}^\star, \mathbf{u}^\star, \mathbf{v}^\star$ then $\mathbf{x}^\star$ is optimal for the primal and $\mathbf{u}^\star, \mathbf{v}^\star$ are optimal for the dual problem.
- If $F^\star = D^\star$ the problem is said to have **strong duality** .
- **Slater's constraint qualification**: if the primal problem is convex and there exists a feasible solution :

$$\exists \mathbf{x} \in \mathbb{R}^n, \; h_j(\mathbf{x}) = 0, \; g_i(\mathbf{x}) \leq 0 \; \forall i, j$$

then strong duality holds.

Karush-Kuhn-Tucker (KKT) conditions

Optimization problems and Lagrangian

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) \qquad\qquad \max_{\mathbf{u}\in\mathbb{R}^q,\mathbf{v}\in\mathbb{R}^p} \quad D(\mathbf{u},\mathbf{v})$$

$$\text{with} \quad h_j(\mathbf{x}) = 0 \quad \forall j = 1,\dots,p \qquad\qquad \text{with} \quad \mathbf{u} \geq \mathbf{0}$$

$$\text{and} \quad g_i(\mathbf{x}) \leq 0 \quad \forall i = 1,\dots,q.$$

$$\mathcal{L}(\mathbf{x},\mathbf{u},\mathbf{v}) = f(\mathbf{x}) + \sum_{i=1}^{k} u_i g_i(\mathbf{x}) + \sum_{j=1}^{m} v_j h_j(\mathbf{x}), \qquad \text{with } \mathbf{u} \geq 0$$

Karush-Kuhn-Tucker (KKT) conditions

1. $\nabla_\mathbf{x} F(\mathbf{x}) + \sum_i u_i \nabla_\mathbf{x} g_i(\mathbf{x}) + \sum_j v_j \nabla_\mathbf{x} h_j(\mathbf{x}) = 0$ — Stationarity
2. $g_i(\mathbf{x}) \leq 0, h_j(\mathbf{x}) = 0, \quad \forall i, \forall j$ — Primal feasibility
3. $u_i \geq 0 \; \forall i$ — Dual feasibility
4. $u_i g_i(\mathbf{x}) = 0 \; \forall i$ — Complementarity

Solution and optimality conditions

Solution of the optimization problem

For a problem with strong duality (satisfying Slater's conditions) the two following statements are equivalent:

- $\mathbf{x}^\star$ and $\mathbf{u}^\star, \mathbf{v}^\star$ are solutions of the primal and dual problems.
- $\mathbf{x}^\star$ and $\mathbf{u}^\star, \mathbf{v}^\star$ satisfy the KKT conditions.

Finding a solution (sometimes)

1. Express the Lagrangian.
2. Express the KKT conditions
3. Try to find an analytic solution for $\mathbf{x}^\star$ as function of $\mathbf{u}, \mathbf{v}$.
4. Express the dual problem and solve it if easier than primal.
5. Use KKT to recover the primal solution $\mathbf{x}^\star$

Exercise 2: KKT conditions

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}\|^2 \qquad \text{subject to } \sum_{i=1}^{n} x_i = 1$$

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, v) =$$

2. Express the KKT of the problem:
   2.1
   2.2

3. Deduce from 1 and 2 above the optimal $v^\star$ by maximizing $D(v)$ then $\mathbf{x}^\star$:

$$D(v) =$$

$$v_i^\star = \qquad\qquad x_i^\star =$$

Linear equality constraints

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) \tag{5}$$

$$\text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{b}$$

- With $\mathbf{A} \in \mathbb{R}^{p\times n}$ defining p linearly independent constraints.
- We can eliminate the equality constraints using basic linear algebra.

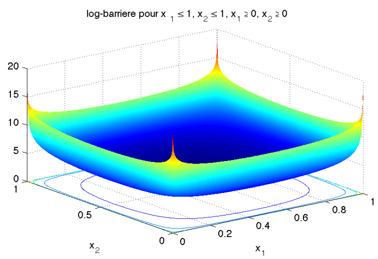
$$\{\mathbf{x}|\mathbf{A}\mathbf{x} = \mathbf{b}\} \quad = \quad \{\mathbf{F}\mathbf{z} + \hat{\mathbf{x}}|\mathbf{z} \in \mathbb{R}^{n-p}\}$$

  where $\hat{\mathbf{x}}$ is a vector satisfying $\mathbf{A}\hat{\mathbf{x}} = \mathbf{b}$ and $Im(\mathbf{F}) = Ker(\mathbf{A})$.
- In Python one can compute $\mathbf{F}$ with `scipy.linalg.null_space`.
- The equivalent unconstrained problem is then

$$\min_{\mathbf{z}\in\mathbb{R}^{n-p}} \quad F(\mathbf{F}\mathbf{z} + \hat{\mathbf{x}}) \tag{6}$$

  where we can recover the solution of (5) with $\mathbf{x}^\star = \mathbf{F}\mathbf{z}^\star + \hat{\mathbf{x}}$.

Log-Barrier function



Approximating the inequality constraints

- The **log-barrier** function is an approximation of the characteristic function χ.
- The hard constraints can then be replaced by the log-barrier with $\delta > 0$

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(x) \qquad \text{s.t.} \quad g_i(\mathbf{x}) \leq 0 \; \forall i \qquad \Rightarrow \qquad \min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) + \frac{1}{\delta}\sum_{i=1}^{q} -\log(-g_i(\mathbf{x}))$$

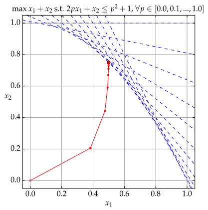
Interior point solver

$$\mathbf{x}(\delta) = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) + \frac{1}{\delta}\sum_{i=1}^{q} -\log(-g_i(\mathbf{x})) \tag{7}$$

Interior Point algorithm

Initialize with a feasible $\mathbf{x}$, and $\delta > 0, \mu > 1$

1. $\mathbf{x} = \mathbf{x}(\delta)$
2. $\delta = \mu\delta$
3. Go to 1. until convergence.



Properties of the algorithm

- Requires a solver for the inner problem : computing $\mathbf{x}(\delta)$
- Inner problem is unconstrained and smooth inside the domain.
- Converges to the solution of the constrained problem : $\lim_{\delta\rightarrow\infty} \mathbf{x}(\delta) = \mathbf{x}^\star$
- All iterations are inside the constraints.
- Converges provably in polynomial time for LP and QP.

More details: [Boyd and Vandenberghe, 2004, Ch.11], [Nocedal and Wright, 2006, Ch. 19]

Linear Program (LP)

Linear program in standard form

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{c}^\top\mathbf{x} \qquad \text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0 \tag{8}$$

- $\mathbf{c} \in \mathbb{R}^n$
- $\mathbf{A} \in \mathbb{R}^{p\times n}, \mathbf{b} \in \mathbb{R}^p$
- Other standard forms exist

- Linear objective function
- Linear constraints
- No inequality for standard LP.

Problem as a function of $\mathbf{A}\mathbf{x} = \mathbf{b}$

- Underdetermined $(p < d)$: more variables than equations.
- Determined $(p = d)$: as many equations than variables, a unique solution $\mathbf{x}^\star = \mathbf{A}^{-1}\mathbf{b}$ if $\mathbf{A}$ invertible.
- Overdetermined $(p > d)$: not feasible

We look at the case where $p < d$.

Linear Program (LP)

General formulation for LP

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{c}^T\mathbf{x} \qquad \text{s.t.} \quad \mathbf{G}\mathbf{x} \leq \mathbf{h}, \quad \mathbf{A}\mathbf{x} = \mathbf{b} \tag{9}$$

- Closer formulation to the constrained optimization (1).
- $\mathbf{A} \in \mathbb{R}^{p\times n}, \mathbf{b} \in \mathbb{R}^p$, and $\mathbf{G} \in \mathbb{R}^{q\times n}, \mathbf{h} \in \mathbb{R}^q$.
- Most standard solvers (open source and commercial) use this formulation.

Exercise 3: Classical constraints

Express the matrices and vectors from general LP above for the following constraints:

- Positivity $\mathbf{x} \geq \mathbf{0}$:
- Simplex $\{\mathbf{x} | \mathbf{x} \geq \mathbf{0}, \sum_i x_i = 1\}$:
- Box constraints $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$:

Example of LP : Optimal Transport (OT)

Definition of the problem

- n factories produce $a_i, \forall i$ amount of goods (vector $\mathbf{a}$).
- d stores need to sell $s_j, \forall j$ amount of goods ((vector $\mathbf{s}$, same total as $\mathbf{a}$)).
- There is a cost $C_{i,j}$ of transporting a unitary amount of good from factory i to store j.
- Find the optimal (cheapest) way to move all the goods between factories and stores. A solution of the problem is called a transport matrix.

Optimal transport problem

$$\min_{\mathbf{X}\in\mathbb{R}^{n\times d}} \quad \sum_{i=1,j=1}^{n,d} C_{i,j}X_{i,j}$$
$$\text{s.t.} \quad \sum_j X_{i,j} = a_i \quad \forall i, \sum_i X_{i,j} = s_j \quad \forall j$$
$$X_{i,j} \geq 0 \qquad \forall i,j$$

- Resource allocation problem .
- Proposed by [Kantorovich, 1942].
- Nobel prize in economy.
- Now used a lot in machine learning.

Exercise 4: OT expressed as general LP problem

We express the matrix $\mathbf{x}$ as the concatenation of the rows of the matrix $\mathbf{X}$:

$$\mathbf{x} = [X_{1,1}, X_{1,2}, X_{1,3}, \dots X_{n,d-1}, X_{n,d}]^T$$

The cost matrix $\mathbf{C}$ is also vectorized as $\mathbf{c}$.

1. Express the row-wise equality constraints $\sum_j x_{i,j} = a_i, \forall i$ and $\mathbf{A}_1\mathbf{x} = \mathbf{a}$:

   $$\mathbf{A}_1 =$$

   The matrix can be expressed simply with the Kroenecker product $\otimes$
2. Express the column-wise equality constraints $\sum_i x_{i,j} = s_j, \forall j$ and $\mathbf{A}_2\mathbf{x} = \mathbf{s}$:

   $$\mathbf{A}_2 =$$

3. Express all the matrices in the general LP :

   $$\mathbf{A} = \begin{bmatrix}\mathbf{A}_1\\ \mathbf{A}_2\end{bmatrix}, \quad \mathbf{b} = \qquad , \quad \mathbf{G} = \qquad , \quad \mathbf{h} =$$

Reduction from general to standard problem

Reformulation to standard LP with positive variables

$$\begin{aligned}\min_{\mathbf{x}\in\mathbb{R}^n} \quad & \mathbf{c}^T\mathbf{x}\\ \text{s.t.} \quad & \mathbf{G}\mathbf{x} \leq \mathbf{h}\\ & \mathbf{A}\mathbf{x} = \mathbf{b}\end{aligned} \quad \equiv \quad \begin{aligned}\min_{\mathbf{x}^+\in\mathbb{R}^n, \mathbf{x}^-\in\mathbb{R}^n, \mathbf{x}^+\in\mathbb{R}^q} \quad & \mathbf{c}^T\mathbf{x}^+ - \mathbf{c}^T\mathbf{x}^-\\ \text{s.t.} \quad & \mathbf{G}\mathbf{x} + \mathbf{s} = \mathbf{h}\\ & \mathbf{A}\mathbf{x}^+ - \mathbf{A}\mathbf{x}^- = \mathbf{b}\\ & \mathbf{x}^+ \geq 0, \mathbf{x}^- \geq 0, \mathbf{s} \geq 0\end{aligned}$$

- We express $\mathbf{x} = \mathbf{x}^+ - \mathbf{x}^-$ as a difference of positive variables.
- The positive variable $\mathbf{s} \geq \mathbf{0}$ is used to recover an equality constraint.
- Problem on the right can be reformulated as standard LP (only equality constraints and positivity)
- The two "tricks" above are classical tools for reformulation.

Primal and Dual problems

Primal LP

$$\begin{aligned}\min_{\mathbf{x}\in\mathbb{R}^n} \quad & \mathbf{c}^\top\mathbf{x}\\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}\\ & \mathbf{x} \geq 0\end{aligned}$$

Dual LP

$$\begin{aligned}\max_{\mathbf{v}\in\mathbb{R}^p} \quad & -\mathbf{b}^\top\mathbf{v}\\ \text{s.t.} \quad & -\mathbf{A}^T\mathbf{v} \leq \mathbf{c}\end{aligned}$$

Primal VS Dual

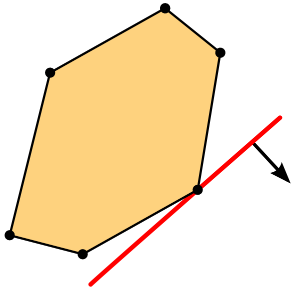
- The problem permute their variables and constraints.
- When there is strict duality (problem has a solution):

  $$\mathbf{c}^\top\mathbf{x}^\star = -\mathbf{b}^\top\mathbf{v}^\star$$

- Finding $\mathbf{x}^\star$ from $\mathbf{v}^\star$ and vice versa:
  1. Find which values of $\mathbf{x}^\star$ and 0 from the equality $(\mathbf{A}^T\mathbf{v}^\star - \mathbf{c})^T\mathbf{x}^\star = 0$.
  2. Solve the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ for the non-zero components of $\mathbf{x}^\star$.

Solution of the standard LP

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{c}^\top \mathbf{x}$$
$$\text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{b}$$
$$\mathbf{x} \geq \mathbf{0}$$



Property of the solution

- Problem is convex but possibly has an infinite number of solution (one side of the polytope).
- Solution $\mathbf{x}^\star$ is always on a border of the polytop describing the constraints.
- There is at most p ($\mathbf{A} \in \mathbb{R}^{p\times n}$) components of $\mathbf{x}^\star$ that are non-zero.
- Those non-zeros components are called **active variables**.

Robust regression with Least Absolute Deviation

$$\min_{\mathbf{w}\in\mathbb{R}^d} \quad \sum_{i=1}^{n} |y_i - \mathbf{x}_i^T \mathbf{w}|$$

- More robust to outliers than least squares but also less stable [Barrodale and Roberts, 1973].

Exercise 5: Reformulations as LP

1. Reformulate problem above as a LP with additional variables $\mathbf{e}^+ \geq \mathbf{0}, \mathbf{e}^- \geq \mathbf{0}$ such that $\mathbf{y} - \mathbf{X}\mathbf{w} = \mathbf{e}^+ - \mathbf{e}^-$ with $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T$:

$$\min_{\mathbf{w},\mathbf{e}^+,\mathbf{e}^-}$$

2. Reformulate problem above as a LP with additional variable $\mathbf{f} \geq \mathbf{0}_n$ such that $|\mathbf{H}\mathbf{x} - \mathbf{y}| \leq \mathbf{f}$:

$$\min_{\mathbf{w},\mathbf{f}}$$

L1 Support Vector Machines

$$\min_{\mathbf{w}\in\mathbb{R}^d} \quad \sum_{i=1}^{n} \max(0, 1 - y_i \mathbf{x}_i^T \mathbf{w}) \qquad (10)$$
$$\text{s.t.} \quad \|\mathbf{w}\|_1 \leq \beta$$

- Proposed in [Zhu et al., 2004], to promote sparsity in SVM (with the L1 norm).
- Problem above can be reformulated as the following optimization problem :

$$\min_{\mathbf{f},\mathbf{w}^+,\mathbf{w}^-} \quad \mathbf{1}_n^T \mathbf{f}$$
$$\text{s.t.} \quad \mathbf{1}_n - (\mathbf{y}\odot\mathbf{X})\mathbf{w}^+ + (\mathbf{y}\odot\mathbf{X})\mathbf{w}^- \leq \mathbf{f}$$
$$\mathbf{1}_d^T\mathbf{w}^+ + \mathbf{1}_d^T\mathbf{w}^- \leq \beta, \quad \mathbf{f} \geq \mathbf{0}, \quad \mathbf{w}^+ \geq \mathbf{0}, \quad \mathbf{w}^- \geq \mathbf{0}$$

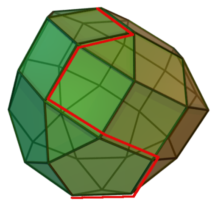
- The corresponding general LP problem with $\mathbf{x} = [\mathbf{w}^{+T}, \mathbf{w}^{-T}, \mathbf{f}]^T$ has the following matrices:

$$\mathbf{c} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1}_n \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} -(\mathbf{y}\odot\mathbf{X}) & (\mathbf{y}\odot\mathbf{X}) & -\mathbf{I}_n \\ \mathbf{1}_{1,d} & \mathbf{1}_{1,d} & \mathbf{0}_{1,n} \\ -\mathbf{I}_d & \mathbf{0}_{d,d} & \mathbf{0}_{d,n} \\ \mathbf{0}_{d,d} & -\mathbf{I}_d & \mathbf{0}_{d,n} \\ \mathbf{0}_{n,d} & \mathbf{0}_{n,d} & -\mathbf{I}_n \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} -\mathbf{1}_n \\ \beta \\ \mathbf{0}_d \\ \mathbf{0}_d \\ \mathbf{0}_n \end{bmatrix}$$

Simplex Algorithm

Main idea behind the simplex

- Initialize with a basic feasible solution $\mathbf{x}^{(0)}$ (on a vertex or extreme point of the polytope).
- Update the solution to decrease the loss at each iteration.
- Use the sparsity of $\mathbf{x}$ (add and remove active variables).



Simplex algorithm

- Invented by Dantzig around 1957.
- Solved the problem he thought was a homework exercise from his course.
- Standard algorithm for solving LP, very efficient for sparse problems but possibly non polynomial (worst case).
- On network flow problems, the adapted network simplex is proven to be polynomial [Orlin, 1997] (optimal transport).
- in Python : `scipy.optimize.linprog(method='simplex')`

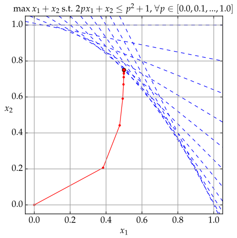
More details: [Vanderbei et al., 2015, part 1]

Interior point solver

Interior point method (IPM) for LP

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{c}^\top \mathbf{x} \qquad \text{s.t.} \quad \mathbf{G}\mathbf{x} \leq \mathbf{h} \qquad \Rightarrow \qquad \min_{\mathbf{x}\in\mathbb{R}^n} \quad \delta \mathbf{c}^\top \mathbf{x} + -\sum_{i=1}^{q} \log(\mathbf{g}_i^T \mathbf{x} - h_i)$$

- Classical solver for linear programs.
- Simplex searches on the corners of the polytope, IPM optimize inside it.
- Never against the constraints until numerical precision is achieved.
- Polynomial complexity for LP (better than simplex in theory).
- In Python: `scipy.optimize.linprog`



More details: [Boyd and Vandenberghe, 2004, Chapter 11], [Vanderbei et al., 2015, Part 3], [Nocedal and Wright, 2006, Chapter 14]

Solving a Linear Program

Simplex and variants

- Exact solutions.
- Can be slow of large problems.
- Use it on structured graph flow.

Interior point problem

- Better at early stopping.
- Usually faster on large problems.
- Most commercial solvers.

LP solvers in Python

- **Scipy** : `scipy.optimize.linprog` function (both simplex and interior points)
- **cvxopt** : Interior point solver for standard problems + wrapper for commercial solvers and GLPK [Vandenberghe, 2010].
- **Mosek** Commercial solver (free for academics) [Andersen and Andersen, 2000].
- **Gurobi** Commercial solver (free for academics).
- **CPLEX** Commercial solver (free for academics).

Benchmark available : `https://github.com/stephane-caron/lpsolvers`

Quadratic Program

Optimization problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{G}\mathbf{x} \leq \mathbf{h} \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned} \tag{11}$$

- $\mathbf{Q} \in \mathbb{R}n \times n$ is a symmetric positive definite matrix (convex QP).
- $\mathbf{A} \in \mathbb{R}^{p\times n}, \mathbf{b} \in \mathbb{R}^p$, and $\mathbf{G} \in \mathbb{R}^{q\times n}, \mathbf{h} \in \mathbb{R}^q$.
- Most standard solvers (open source and commercial) use this formulation.

Special cases

- Unconstrained : close form solution or iterative methods (Conjugate gradients)
- Box constraints $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$: projected gradient (see proximal methods).

QP Exemple: portfolio optimization

- Model proposed by Markowitz in 1952 (Nobel Prize in economy).
- $\mathbf{x}$ is a portfolio of n assets (or stock).
- The price change for each asset is modeled as random variables with expected price change $\mathbf{p}$ and covariance $\mathbf{\Sigma}$.
- For a given portfolio $\mathbf{x}$
  - The expected gain (return) is : $\mathbf{p}^T\mathbf{x}$
  - The expected variance is : $\mathbf{x}^T\mathbf{\Sigma}\mathbf{x}$
- The portfolio optimization can be expressed for a positive balance $b > 0$ as:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{x}^T\mathbf{\Sigma}\mathbf{x} \tag{12}$$

$$\text{s.t.} \quad \mathbf{1}_n^T\mathbf{x} = b \tag{13}$$

$$\mathbf{p}^T\mathbf{x} \geq r_{min} \tag{14}$$

where r_{min} is the minimal return of the portfolio.

Special Case : QP without constaints

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x}+\mathbf{c}^T\mathbf{x} \tag{15}$$

Unconstrained QP

- The gradient of the term above is $\nabla_\mathbf{x} = \frac{1}{2}(\mathbf{Q}+\mathbf{Q}^T)\mathbf{x}+\mathbf{c}$
- For symmetric matrix $\mathbf{Q}$ a solution respects : $\mathbf{Q}\mathbf{x}^\star = -\mathbf{c}$
- If $\mathbf{Q}$ is invertible and strictly positive definite then : $\mathbf{x}^\star = -\mathbf{Q}^{-1}\mathbf{c}$
- To solve the problem several approaches
  1. Solve the linear equations : np. linalg . solve with complexity$O(n^3)$
  2. Solve the linear equations with Conjugate Gradient or other gradient descent methods (see next course).

Exercise 6: Least Square

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|^2 \qquad\qquad \min_{\mathbf{x}\in\mathbb{R}^n} \quad \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x}+\mathbf{c}^T\mathbf{x}+\lambda\frac{1}{2}\|\mathbf{x}\|^2$$

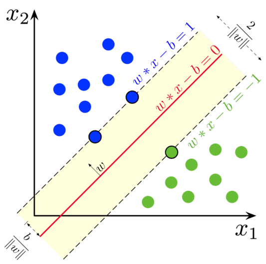
Recover the matrices $\mathbf{Q}$ and $\mathbf{c}$ of the equivalent QP for the problems above:

$\mathbf{Q} =$ $\qquad$ $\mathbf{c} =$ $\qquad$ $\mathbf{Q} =$ $\qquad$ $\mathbf{c} =$

Support Vector Machines (1)

Hard margin SVM [Cortes and Vapnik, 1995]

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1 \end{aligned} \tag{16}$$



- All samples $(\mathbf{x}_i, y_i)$ must be classified well with a margin of at least 1.
- Needs the data to be linearly separable.
- Minimizing the norm of $\mathbf{w}$ corresponds to maximizing the margin $\frac{2}{\mathbf{w}}$.

Soft margin SVM

$$\min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \quad C\sum_i \max(0, 1-y_i(\mathbf{x}_i^T\mathbf{w}+b)) + \frac{1}{2}\|\mathbf{w}\|^2 \tag{17}$$

- The margin constraints are relaxed with the Hinge loss.
- C is the weight of the data fitting term.
- Non differentiable convex problem.

Support Vector Machines (2)

Primal SVM

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}, \mathbf{z}\in\mathbb{R}^n} \quad & C\sum_i z_i + \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1 - z_i, \ \forall i \\ & \mathbf{z} \geq \mathbf{0} \end{aligned} \tag{18}$$

- We introduce the variables $z_i \geq 0$ such that $z_i = \max(0, 1-y_i(\mathbf{x}_i^T\mathbf{w}+b))$.

Dual SVM

$$\begin{aligned} \min_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad & \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{Q}\boldsymbol{\alpha} - \mathbf{1}_n^T\boldsymbol{\alpha} \\ \text{s.t.} \quad & \mathbf{y}^T\boldsymbol{\alpha} = 0 \\ & \mathbf{0}_n \leq \boldsymbol{\alpha} \leq C\mathbf{1}_n \end{aligned} \tag{19}$$

- QP ($Q_{i,j} = y_i y_j \mathbf{x}_i^T\mathbf{x}_j$) with box constraints and one linear constraint.
- Primal solution can be recovered with : $\mathbf{w}^\star = \sum_i y_i\alpha_i^\star\mathbf{x}_i$.
- $b^\star$ can be found on a support vector where inequality becomes equality.
- Most common formulation because allows the use of kernel for nonlinear classification ($Q_{i,j} = y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$)

Lasso estimator

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{X}\mathbf{w}-\mathbf{y}\|^2 + \lambda\sum_i |w_i| \tag{20}$$

- Classical approach to perform regression with variable selection [Tibshirani, 1996].
- Quadratic data fitting, L1 regularization term.
- Expressed either as additive term or constraint (equivalent problem).

Exercise 7: Lasso reformulation as QP

1. Reformulate the Lasso problem as a positive QP with $\mathbf{w} = \mathbf{w}^+ - \mathbf{w}^-$

$$\begin{aligned} \min_{\mathbf{w}^+, \mathbf{w}^-} \quad & \\ \text{s.t.} \quad & \mathbf{w}^+ \geq \mathbf{0}, \ \mathbf{w}^- \geq \mathbf{0} \end{aligned}$$

2. Express the matrices $\mathbf{Q}, \mathbf{c}, \mathbf{G}, \mathbf{h}$ for standard QP corresponding to the problem.

$\mathbf{Q} =$ $\qquad$ $\mathbf{c} =$ $\qquad$ $\mathbf{G} =$ $\qquad$ $\mathbf{h} =$

Active set Algorithm

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{G}\mathbf{x} \leq \mathbf{h} \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned}$$

Principle of active set method

- Search for the active constraints $\mathcal{A}(\mathbf{x}^\star)$.
- If the optimal active set is known the problem is an equality constrained QP.
- QP with equality constraint can be solved with : null space + unconstrained QP.
- QP version of the simplex (search on which constraints is the solution).
- Very efficient on some problems (positivity, bloc constraints, SVM).

Active set Method (simplified)

Initialize feasible $\mathbf{x}$, $\mathcal{A}(\mathbf{x}) = \{i | \mathbf{g}_i^T\mathbf{x} = h_i\}$ the active set of inequality constraints.

1. Solve subproblem with inequality constraints in $\mathcal{A}(\mathbf{x})$ forced to equality.
2. Update the active set using KKT conditions.

More details: [Nocedal and Wright, 2006, Sec. 16.5]

Sequential Minimal Optimization (SMO)

$$\begin{aligned} \min_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad & \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{Q}\boldsymbol{\alpha} - \mathbf{1}_n^T\boldsymbol{\alpha} \\ \text{s.t.} \quad & \mathbf{y}^T\boldsymbol{\alpha} = 0 \\ & \mathbf{0}_n \leq \boldsymbol{\alpha} \leq C\mathbf{1}_n \end{aligned}$$

Principle of SMO

- Proposed in [Platt, 1998] to solve large scale SVM.
- Coordinate descent algorithm taking into account $\mathbf{y}^T\boldsymbol{\alpha} = 0$.
- The choice of the coordinates to update is sensitive.
- Sate of the art solver for SVM [Chang and Lin, 2001] also use a cache for computing the kernel matrix.

SMO Algorithm

Initialize feasible $\boldsymbol{\alpha}$

1. Find two components α_i and α_j that violate KKT conditions.
2. Solve the QP on only those components (1D problem).

Solving a QP

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{G}\mathbf{x} \leq \mathbf{h} \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned}$$

Main Algorithms

- **Interior points** Efficient for large problems (commercial solvers).
- **Active set** General solver, an be very fast on structured problems (sparsity, SVM)
- **SMO** State of the art solver for SVM.

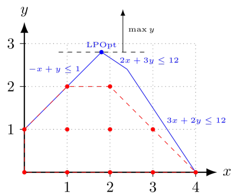
QP Solvers in Python

- **Numpy** (no constraints): (np. linalg . solve ornp. linalg . lstsq).
- **quadprog** : Implements active set [Goldfarb and Idnani, 1983]
- **cvxopt** : Interior point solver for standard problems + wrapper for Mosek.
- **OSQP** : Operator spliting QP solver [Stellato et al., 2017].
- **Mosek** : Commercial solver (free for academics) [Andersen and Andersen, 2000].
- **Gurobi** : Commercial solver (free for academics).

Benchmark available : `https://github.com/stephane-caron/qpsolvers`

Integer Programming

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) \\ \text{s.t.} \quad & h_j(\mathbf{x}) = 0 \quad \forall j = 1, \dots, p \\ & g_i(\mathbf{x}) \leq 0 \quad \forall i = 1, \dots, q. \\ & \mathbf{x} \in \mathbb{Z}^n \end{aligned} \qquad (21)$$



- Classical optimization problem with additional integer constraints $\mathbf{x} \in \mathbb{Z}^n$.
- Zero-one programing when variables can be only binary $\mathbf{x} \in \{0, 1\}^n$.
- **Mixed Integer Programming (MIP)** problems when only part of the variables are integer : $x_i \in \mathbb{Z} for i = 1, \dots, n_i$ with $n_i \leq n$.
- Problem is extremely hard to solve exactly (NP complete).

Algorithms

- Continuous relaxation (and then rounding, can work well on MILP).
- Cutting Plane Algorithm (relaxation + iteratively add linear constraints).
- Branch and bound (exact method using upper and lower bounds to split the space of solution).

MILP and MIQP

Mixed Integer LP (MILP)

$$\begin{aligned}\min_{\mathbf{x}\in\mathbb{R}^n} \quad & \mathbf{c}^T\mathbf{x}\\ \text{s.t.} \quad & \mathbf{x}\geq \mathbf{0}\\ & \mathbf{A}\mathbf{x}=\mathbf{b}\\ & x_i\in\mathbb{Z}, \forall i\in\{1,\dots,n_i\}\end{aligned}$$

Mixed Integer QP (MIQP)

$$\begin{aligned}\min_{\mathbf{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x}+\mathbf{c}^T\mathbf{x}\\ \text{s.t.} \quad & \mathbf{x}\geq \mathbf{0}\\ & \mathbf{A}\mathbf{x}=\mathbf{b}\\ & x_i\in\mathbb{Z}, \forall i\in\{1,\dots,n_i\}\end{aligned}$$

- Well studied MIP problems.
- For MILP, relaxation can be exact (total unimodularity of $\mathbf{A}$)
- Solved by Branch and Bound and cutting planes in general.

MIP solvers in Python

- **cvxpy** : General optimization (multiple wrappers) [Diamond and Boyd, 2016].
- **ECOS** : Embedded Conic Solver for MILP [Domahidi et al., 2013].
- **Mosek** : Commercial solver (free for academics) [Andersen and Andersen, 2000].
- **Gurobi** : Commercial solver (free for academics).

L0 sparse regression

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \frac{1}{2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|^2+\lambda\|\mathbf{x}\|_0$$

Problem above can be reformulated as a MIQP [Bourguignon et al., 2015].

- First we introduce a binary vector $\mathbf{z}\in\{0,1\}^n$.
- We suppose that $z_i=1$ if variable $\mathbf{x}_i\neq 0$ else $z_i=0$. This means that for a big enough M we have:

$$-M\mathbf{z}\leq\mathbf{x}\leq M\mathbf{z}$$

- We can express the L0 sparse regression as the following optimization problem:

$$\begin{aligned}\min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{z}\in\mathbb{R}^n} \quad & \frac{1}{2}\mathbf{x}^\top\mathbf{H}^T\mathbf{H}\mathbf{x}-(\mathbf{H}^T\mathbf{y})^T\mathbf{x}+\lambda\mathbf{1}_n^T\mathbf{z}\\ \text{s.t.} \quad & -M\mathbf{z}\leq\mathbf{x}\leq M\mathbf{z}\\ & \mathbf{z}\in\{0,1\}^n\end{aligned}$$

Other formulations corresponds to constrained expression but all use the "big M" trick.

Quadratically Constrained QP (QCQP)

Optimization problem

$$\begin{aligned}\min_{\mathbf{x}\in\mathbb{R}^n} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{Q}_0\mathbf{x}+\mathbf{c}_0^T\mathbf{x}\\ \text{s.t.} \quad & \mathbf{x}^T\mathbf{Q}_i\mathbf{x}+\mathbf{c}_i^T\mathbf{x}\leq\mathbf{h}_i,\ \forall i=1,\dots,m\\ & \mathbf{A}\mathbf{x}=\mathbf{b}\end{aligned} \tag{22}$$

- If $\mathbf{Q}_0,\dots,\mathbf{Q}_m$ are positive definite then the problem is convex and can be solved with interior point.
- QCQP is NP-hard, it is easy to prove since a Zero-One integer program can be cast as a QCQP with the following constraints that force $x_i\in\{0,1\}$:

$$x_i(1-x_i)\geq 0 \quad \text{and} \quad x_i(1-x_i)\leq 0$$

- QCQP can sometimes be solved by relaxation (Semi-definite programming or second-order cone programming)

QCQP solvers in Python

- **cvxpy** : with nonconvex QCQP extension [Park and Boyd, 2017] .
- **Mosek** : Commercial solver (free for academics) [Andersen and Andersen, 2000].
- **Gurobi** : Commercial solver (free for academics).

K-means as MIQCQP

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

- The argmin for each sample can be replaced by a linear term with a matrix $\mathbf{Z}\in\{0,1\}^{N,K}$ modeling the clustering of the samples.
- We force a unique cluster selection with constraints

$$\mathbf{Z}\in\{0,1\}^{N,K}, \quad \mathbf{Z}\mathbf{1}_K=\mathbf{1}_N$$

- We introduce the distance variable as $D_{i,k}=\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2$
- The optimization problem above can be expressed as

$$\begin{aligned}\min_{\bar{\mathbf{x}}_k,\forall_k,\mathbf{Z}\in\mathbb{R}^{N\times K},\mathbf{D}\in\mathbb{R}^{N\times K}} \quad & \sum_{i,k} Z_{i,k}D_{i,k}\\ \text{s.t.} \quad & D_{i,k}=\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2,\ \forall_i,\forall k\\ & \mathbf{Z}\mathbf{1}_K=\mathbf{1}_N\\ & \mathbf{Z}\in\{0,1\}^{N,K}\end{aligned} \tag{23}$$

Warning: Never try to solve K-means with this formulation!

Second Order Cone Programming (SOCP)

Optimization problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \|\mathbf{A}_i\mathbf{x}-\mathbf{b}_i\|_2 \leq \mathbf{h}_i^T\mathbf{x}+d_i, \quad i=1,\dots,m \\ & \mathbf{A}_0\mathbf{x}=\mathbf{b}_0 \end{aligned} \tag{24}$$

- The following constraint is called a Second order cone constraint:

$$\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2 \leq \mathbf{h}^T\mathbf{x}+d$$

- When $\mathbf{h}_i=\mathbf{0}$, $\forall i$ the problem is a QCQP (one can square the norm).
- Other kind of cone constraints can be used (definte positive matrices).

SOCP solvers in Python

- **cvxopt** : Interior point solver [Vandenberghe, 2010].
- **cvxpy** : SOCP solver [Diamond and Boyd, 2016].
- **Mosek** : Commercial solver (free for academics) [Andersen and Andersen, 2000].
- **Gurobi** : Commercial solver (free for academics).

Robust Support Vector Machines

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}, \mathbf{z}\in\mathbb{R}^n} \quad & C\sum_i z_i + \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1 - z_i + \gamma_i \left\|\mathbf{\Sigma}_i^{\frac{1}{2}}\mathbf{w}\right\|, \ \forall i \\ & \mathbf{z}\geq\mathbf{0} \end{aligned} \tag{25}$$

- Proposed in [Shivaswamy et al., 2006] to handle uncertain and missing data.
- We suppose that we have uncertain data $(\mathbf{x}_i, y_i)$ and that the training sample $\mathbf{x}_i$ has a covariance matrix $\mathbf{\Sigma}_i$ to model its uncertainty.
- In this can one want to replace the hard margin constraint by a probabilistic variant

$$P(y_i(\mathbf{x}_i^T\mathbf{w}+b)\geq 1-z_i)\geq 1-\kappa_i$$

  were κ_i is small.
- When using the normal distribution on the training samples, one can recover the optimization porblem above with $\gamma_i=\phi^{-1}(\kappa_i)$ where ϕ is the normal CDF.

Semi-Definite Programming

Optimization problem

$$\begin{aligned} \min_{\mathbf{X}\in\mathbb{S}^n} \quad & \langle\mathbf{X},\mathbf{C}\rangle_{\mathbb{S}^n} \\ \text{s.t.} \quad & \langle\mathbf{X},\mathbf{A}_i\rangle_{\mathbb{S}^n} = b_i, \quad i=1,\dots,m \\ & \mathbf{X}\succeq 0 \end{aligned} \tag{26}$$

- $\mathbb{S}^n$ is the set of $n\times n$ symmeric matrices.
- $\langle\mathbf{X},\mathbf{C}\rangle_{\mathbb{S}^n}=\sum_{i,j}X_{i,j}C_{i,j}$ is the Frobenius scalar product between matrices.
- The constraint $\mathbf{X}\succeq 0$ force $\mathbf{X}$ to be semi-definite positive.
- Special case of cone programming (cone of positive semi-definite matrices).
- Can be solved efficiently with interior point solver.

SDP solvers in Python

- **cvxopt** : Interior point solver [Vandenberghe, 2010].
- **cvxpy** : SDP solver [Diamond and Boyd, 2016].
- **Mosek** : Commercial solver (free for academics) [Andersen and Andersen, 2000].
- **Gurobi** : Commercial solver (free for academics).

Conclusion

Standard Problems (properties)

- Linear or quadratic objective function.
- Linear, quadratic of conic constraints.
- Real of integer variables.

Approach

- Express the Lagrangian to find optimality conditions (KKT).
- Try to express your problem as a standard problems.
- Use generic solvers for first tests (small problems).
- Find variant of generic solver that works better for your problem.

Next part of the course

- **Smooth optimization** : Problems without constraints.
- **Non-smooth optimization** : Problems with non-smooth objectives and constraints.

References I

[Andersen and Andersen, 2000] Andersen, E. D. and Andersen, K. D. (2000).
The mosek interior point optimizer for linear programming: an implementation of the homogeneous algorithm.
In *High performance optimization*, pages 197–232. Springer.

[Barrodale and Roberts, 1973] Barrodale, I. and Roberts, F. D. (1973).
An improved algorithm for discrete l_1 linear approximation.
SIAM Journal on Numerical Analysis, 10(5):839–848.

[Bourguignon et al., 2015] Bourguignon, S., Ninin, J., Carfantan, H., and Mongeau, M. (2015).
Exact sparse approximation problems via mixed-integer programming: Formulations and computational performance.
IEEE Transactions on Signal Processing, 64(6):1405–1419.

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

[Chang and Lin, 2001] Chang, C.-C. and Lin, C.-J. (2001).
Libsvm: a library for support vector machines," 2001. software available at http://www. csie. ntu. edu. tw/˜ cjlin/libsvm.

References II

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995).
Support-vector networks.
Machine learning, 20(3):273–297.

[Diamond and Boyd, 2016] Diamond, S. and Boyd, S. (2016).
CVXPY: A Python-embedded modeling language for convex optimization.
Journal of Machine Learning Research, 17(83):1–5.

[Domahidi et al., 2013] Domahidi, A., Chu, E., and Boyd, S. (2013).
ECOS: An SOCP solver for embedded systems.
In *European Control Conference (ECC)*, pages 3071–3076.

[Goldfarb and Idnani, 1983] Goldfarb, D. and Idnani, A. (1983).
A numerically stable dual method for solving strictly convex quadratic programs.
Mathematical programming, 27(1):1–33.

[Kantorovich, 1942] Kantorovich, L. V. (1942).
On the translocation of masses.
In *Dokl. Akad. Nauk. USSR (NS)*, volume 37, pages 199–201.

References III

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

[Orlin, 1997] Orlin, J. B. (1997).
A polynomial time primal network simplex algorithm for minimum cost flows.
Mathematical Programming, 78(2):109–129.

[Park and Boyd, 2017] Park, J. and Boyd, S. (2017).
General heuristics for nonconvex quadratically constrained quadratic programming.
arXiv preprint arXiv:1703.07870.

[Platt, 1998] Platt, J. (1998).
Sequential minimal optimization: A fast algorithm for training support vector machines.

[Shivaswamy et al., 2006] Shivaswamy, P. K., Bhattacharyya, C., and Smola, A. J. (2006).
Second order cone programming approaches for handling missing and uncertain data.
Journal of Machine Learning Research, 7(Jul):1283–1314.

[Stellato et al., 2017] Stellato, B., Banjac, G., Goulart, P., Bemporad, A., and Boyd, S. (2017).
OSQP: An operator splitting solver for quadratic programs.
ArXiv e-prints.

References IV

[Tibshirani, 1996] Tibshirani, R. (1996).
Regression shrinkage and selection via the lasso.
Journal of the Royal Statistical Society: Series B (Methodological), 58(1):267–288.

[Vandenberghe, 2010] Vandenberghe, L. (2010).
The cvxopt linear and quadratic cone program solvers.
Online: http://cvxopt. org/documentation/coneprog. pdf.

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

[Zhu et al., 2004] Zhu, J., Rosset, S., Tibshirani, R., and Hastie, T. J. (2004).
1-norm support vector machines.
In *Advances in neural information processing systems*, pages 49–56.