

Optimization for data science

Constrained Optimization and Standard Problems

R. Flamary

Master Data Science, Institut Polytechnique de Paris

November 12, 2024



INSTITUT
POLYTECHNIQUE
DE PARIS



Full course overview

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

Current course overview

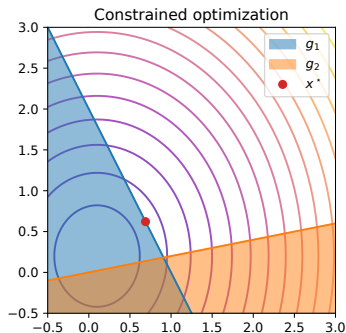
1. Introduction to optimization	4
2. Smooth optimization : Gradient descent	4
3. Smooth Optimization : Quadratic problems	4
4. Non-smooth optimization : Proximal methods	4
5. Stochastic Gradient Descent	4
6. Constrained Optimization and Standard Problems	4
6.1 Constrained optimization	6
6.1.1 Lagrangian of a constrained optimization problem	
6.1.2 Karush-Kuhn-Tucker (KKT) optimality conditions	
6.1.3 Interior point solvers	
6.2 Linear Program (LP)	26
6.2.1 Problem formulation	
6.2.2 Examples of LP in ML	
6.2.3 Simplex Algorithm	
6.3 Quadratic Program (QP)	41
6.3.1 Problem formulation	
6.3.2 Examples of QP in ML	
6.3.3 Active set (AS) and Sequential Minimal Optimization (SMO)	
6.4 Other optimization problems	53
6.4.1 Integer Programming (IP)	
6.4.2 Quadratically constrained QP and Cone Programming	
6.4.3 Semi-Definite Programming (SDP)	
6.5 Conclusion	62
7. Coordinate descent	63
8. Newton and quasi-newton methods	63
9. Beyond convex optimization	63

Constrained optimization and standard problems

Optimization problem (standard form)

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

- ▶ F is convex and differentiable
- ▶ h_j and g_i are differentiable and define convex constraints.
- ▶ The problem is convex.



Standard problems for specialized solvers

- ▶ Linear Programming (LP)
- ▶ Quadratic Programming (QP)
- ▶ Mixed Integer Programming (MIP)
- ▶ Quadratically Constrained QP, Second Order Cone Programming, etc.

Constrained optimization and standard problems

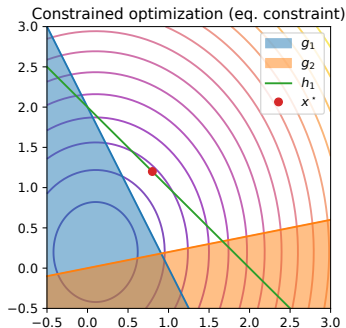
Optimization problem (standard form)

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

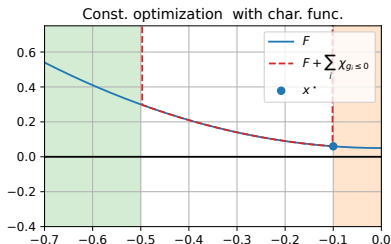
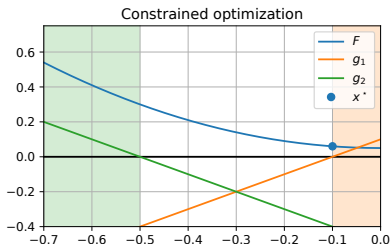
- ▶ F is convex and differentiable
- ▶ h_j and g_i are differentiable and define convex constraints.
- ▶ The problem is convex.

Standard problems for specialized solvers

- ▶ Linear Programming (LP)
- ▶ Quadratic Programming (QP)
- ▶ Mixed Integer Programming (MIP)
- ▶ Quadratically Constrained QP, Second Order Cone Programming, etc.



From indicator to Lagrangian (1)



Constrained optimization problem

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & F(\mathbf{x}) \\ \text{with} \quad & g_i(\mathbf{x}) \leq 0 \quad \forall i = 1, \dots, q. \end{aligned} \quad (2)$$

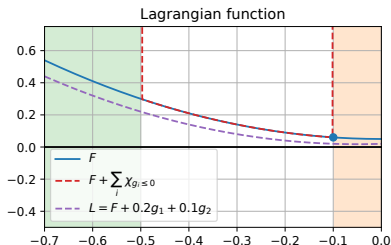
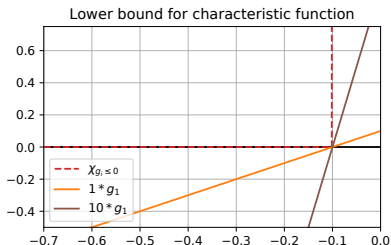
- ▶ The problem is equivalent to

$$\min_{\mathbf{x} \in \mathbb{R}^n} F(\mathbf{x}) + \sum_i \chi_{\leq 0}(g_i(\mathbf{x}))$$

where $\chi_{\leq 0}$ is the characteristic function of the negative values ($+\infty$ for positive value and 0 everywhere else).

- ▶ The problem is not differentiable but can sometimes be solved using :
 - ▶ Projected Gradient Descent (PGD) for simple constraints.
 - ▶ Proximal splitting methods for more complex constraints (needs prox.op.).

From indicator to Lagrangian (2)



Lagrangian (details in [Knowles, 2010])

$$\min_{\mathbf{x} \in \mathbb{R}^n} F(\mathbf{x}) + \sum_i \chi_{\leq 0}(g_i(\mathbf{x}))$$

- ▶ Let us define the Lagrangian function $\mathcal{L}(\mathbf{x}, \mathbf{u})$ such that $\mathbf{u} \geq 0$ and

$$\mathcal{L}(\mathbf{x}, \mathbf{u}) = F(\mathbf{x}) + \sum_{i=1}^k u_i g_i(\mathbf{x})$$

- ▶ For $\mathbf{u} \geq 0$ we have

$$F(\mathbf{x}) + \sum_i \chi_{\leq 0}(g_i(\mathbf{x})) = \max_{\mathbf{u} \geq 0} \mathcal{L}(\mathbf{x}, \mathbf{u}) \geq \mathcal{L}(\mathbf{x}, \mathbf{u})$$

Because $\max_{u \geq 0} uv = \chi_{\leq 0}(v)$ that is is $+\infty$ for $v > 0$ and 0 for $v \leq 0$.

Lagrangian for standard constrained optimization

Optimization problem (standard form)

$$\begin{aligned} & \min_{\mathbf{x} \in \mathbb{R}^n} && F(\mathbf{x}) \\ & \text{with} && h_j(\mathbf{x}) = 0 \quad \forall j = 1, \dots, p \\ & \text{and} && g_i(\mathbf{x}) \leq 0 \quad \forall i = 1, \dots, q. \end{aligned} \tag{3}$$

Lagrangian of the optimization problem

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

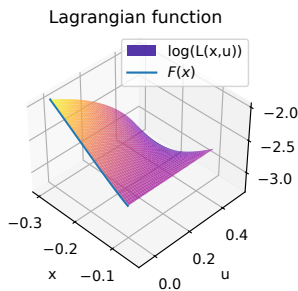
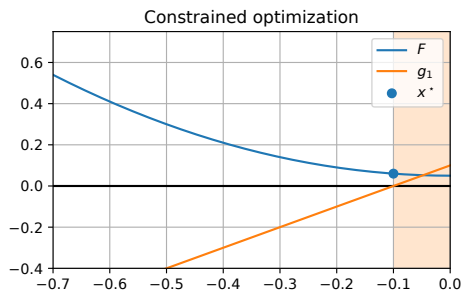
$$\mathcal{L}(\mathbf{x}, \mathbf{u}, \mathbf{v}) = F(\mathbf{x}) + \sum_{i=1}^k u_i g_i(\mathbf{x}) + \sum_{j=1}^m v_j h_j(\mathbf{x}) \tag{4}$$

- ▶ Inequality constraints are multiplied by positive dual variables $u_i \geq 0$.
- ▶ Equality constraints are multiplied by dual variables v_j (any sign).
- ▶ The constrained optimization problem can be reformulated as

$$\min_{\mathbf{x}} \max_{\mathbf{u} \geq 0, \mathbf{v}} \mathcal{L}(\mathbf{x}, \mathbf{u}, \mathbf{v})$$

Finding a saddle point of the problem solves the constrained problem.

Example of Lagrangian



Optimization problem

$$\min_{x \leq -0.1} F(x) = x^2 + 0.05$$

- ▶ Constraint $g(x) = x + 0.1 \leq 0$.
- ▶ The Lagrangian is

$$\mathcal{L}(x, u) = x^2 + 0.05 + u(x + 0.1)$$

with $u \geq 0$.

Lagrange dual function

Lagrange dual function

The Lagrange dual function D of the problem is

$$D(\mathbf{u}, \mathbf{v}) = \inf_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \mathbf{u}, \mathbf{v}) \quad (5)$$

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

Lower bound

For all $\mathbf{u} \geq \mathbf{0}$, \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 $\mathbf{u} \geq \mathbf{0}$.

Exercise 1: Lagrange dual

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

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 :

$$x^* =$$

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

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

Check that the function is concave

Exercise 1: Lagrange dual

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

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, u) = (x + 1)^2 - ux$$

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

$$x^* =$$

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

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

Check that the function is concave

Exercise 1: Lagrange dual

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

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, u) = (x + 1)^2 - ux$$

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

$$x^* = \frac{u}{2} - 1$$

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

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

Check that the function is concave

Exercise 1: Lagrange dual

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

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, u) = (x + 1)^2 - ux$$

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

$$x^* = \frac{u}{2} - 1$$

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

$$D(u) = \mathcal{L}(x^*, u) = -\frac{u^2}{4} - u + 4$$

Check that the function is concave

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

$$F(\mathbf{x}) - D(\mathbf{u}, \mathbf{v}) \geq 0 \quad (6)$$

- ▶ One property of the duality gap is that

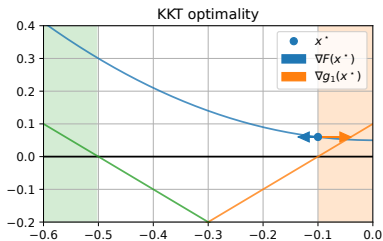
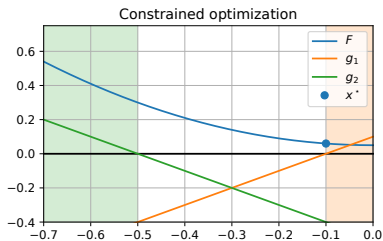
$$F(\mathbf{x}) - F^* \leq F(\mathbf{x}) - D(\mathbf{u}, \mathbf{v})$$

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

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

then strong duality holds.

Karush-Kuhn-Tucker (KKT) conditions



Optimization problems and Lagrangian

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

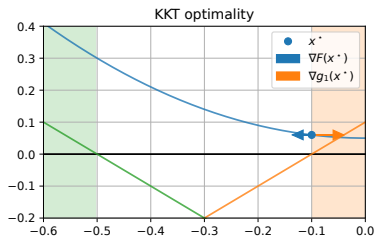
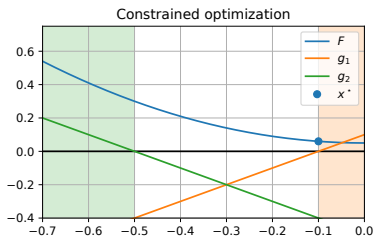
$$\begin{aligned} \max_{\mathbf{u} \in \mathbb{R}^q, \mathbf{v} \in \mathbb{R}^p} \quad & D(\mathbf{u}, \mathbf{v}) \\ \text{with} \quad & \mathbf{u} \geq \mathbf{0} \end{aligned}$$

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

Karush-Kuhn-Tucker (KKT) conditions

- $\nabla_{\mathbf{x}} F(\mathbf{x}) + \sum_i u_i \nabla_{\mathbf{x}} g_i(\mathbf{x}) + \sum_j v_j \nabla_{\mathbf{x}} h_j(\mathbf{x}) = 0$ Stationarity
- $g_i(\mathbf{x}) \leq 0, h_j(\mathbf{x}) = 0, \quad \forall i, \forall j$ Primal feasibility
- $u_i \geq 0 \quad \forall i$ Dual feasibility
- $u_i g_i(\mathbf{x}) = 0 \quad \forall i$ Complementarity

Solution and optimality conditions



Solution of the optimization problem

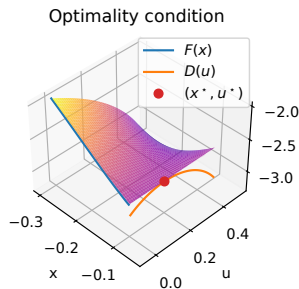
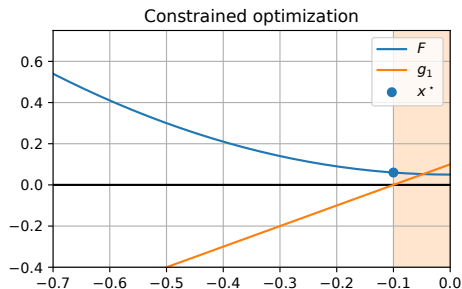
For a problem with strong duality (satisfying Slater's conditions) the two following statements are equivalent:

- ▶ x^* and u^*, v^* are solutions of the primal and dual problems.
- ▶ x^* and u^*, v^* satisfy the KKT conditions.

Finding a solution (sometimes)

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

Example of KKT optimality conditions



Optimization problem

$$\min_{x \leq -0.1} F(x) = x^2 + 0.05$$

- ▶ The Lagrangian is $\mathcal{L}(x, u) = x^2 + 0.05 + u(x + 0.1)$ with $u \geq 0$.
- ▶ $\nabla_x \mathcal{L}(x, u) = 2x + u = 0$ gives $x^* = -\frac{u}{2}$.
- ▶ $D(u) = \mathcal{L}(x^*, u) = -\frac{u^2}{4} + 0.1u + 0.05$.
- ▶ Optimal values are $(x^*, u^*) = (-0.1, 0.2)$.

Exercise 2: KKT conditions

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

1. Express the Lagrangian of the problem above :

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

2. Express the KKT of the problem:

2.1

2.2

2.3

2.4

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

$$D(v) =$$

$$v_i^* =$$

$$x_i^* =$$

Exercise 2: KKT conditions

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

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, v) = \frac{1}{2} \|\mathbf{x}\|^2 + v \left(\sum_{i=1}^n x_i - 1 \right)$$

2. Express the KKT of the problem:

2.1

2.2

2.3

2.4

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

$$D(v) =$$

$$v_i^* =$$

$$x_i^* =$$

Exercise 2: KKT conditions

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

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, v) = \frac{1}{2} \|\mathbf{x}\|^2 + v \left(\sum_{i=1}^n x_i - 1 \right)$$

2. Express the KKT of the problem:

2.1 $x_i + v = 0, \quad \forall i$

2.2 $\sum_{i=1}^n x_i - 1 = 0$

2.3 None

2.4 None

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

$$D(v) =$$

$$v_i^* =$$

$$x_i^* =$$

Exercise 2: KKT conditions

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

1. Express the Lagrangian of the problem above :

$$\mathcal{L}(x, v) = \frac{1}{2} \|\mathbf{x}\|^2 + v \left(\sum_{i=1}^n x_i - 1 \right)$$

2. Express the KKT of the problem:

2.1 $x_i + v = 0, \quad \forall i$

2.2 $\sum_{i=1}^n x_i - 1 = 0$

2.3 None

2.4 None

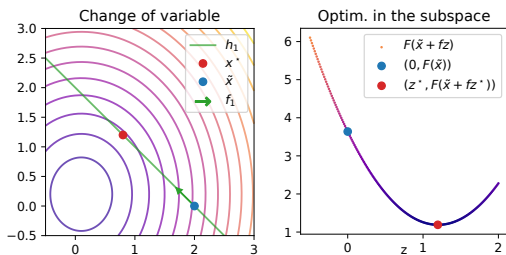
3. Deduce from 1 and 2 above the optimal v^* by maximizing $D(v)$ then \mathbf{x}^* :

$$D(v) = -\frac{nv^2}{2} - v$$

$$v_i^* = -\frac{1}{n}, \forall i \quad x_i^* = \frac{1}{n}, \forall i$$

Linear equality constraints (change of variables)

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & F(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{Ax} = \mathbf{b} \end{aligned} \quad (7)$$



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

$$\{\mathbf{x} | \mathbf{Ax} = \mathbf{b}\} = \{\mathbf{Fz} + \hat{\mathbf{x}} | \mathbf{z} \in \mathbb{R}^{n-p}\}$$

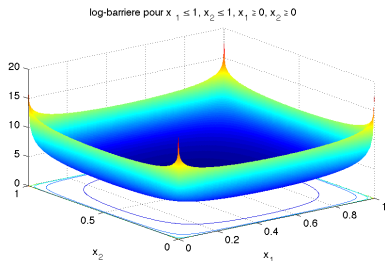
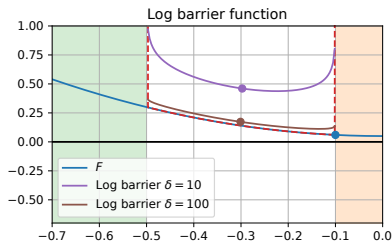
where $\hat{\mathbf{x}}$ is a vector satisfying $\mathbf{A}\hat{\mathbf{x}} = \mathbf{b}$ and $\text{Im}(\mathbf{F}) = \text{Ker}(\mathbf{A})$.

- ▶ In Python one can compute \mathbf{F} with `scipy.linalg.null_space`.
- ▶ The equivalent unconstrained problem is then

$$\min_{\mathbf{z} \in \mathbb{R}^{n-p}} F(\mathbf{Fz} + \hat{\mathbf{x}}) \quad (8)$$

where we can recover the solution of (7) with $\mathbf{x}^* = \mathbf{Fz}^* + \hat{\mathbf{x}}$.

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 $\delta > 0$

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

Interior point solver

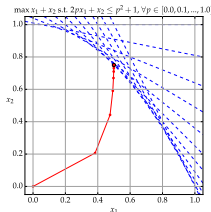
$$\mathbf{x}(\delta) = \arg \min_{\mathbf{x} \in \mathbb{R}^n} F(\mathbf{x}) + \frac{1}{\delta} \sum_{i=1}^q -\log(-g_i(\mathbf{x})) \quad (9)$$

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 : $\lim_{\delta \rightarrow \infty} \mathbf{x}(\delta) = \mathbf{x}^*$
- ▶ All iterations are inside the constraints.
- ▶ Converges provably in polynomial time for LP and QP.

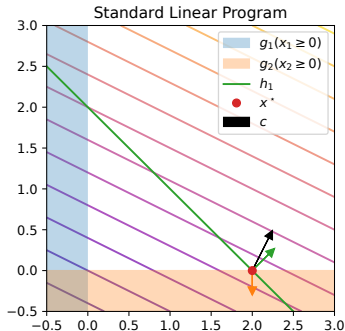
More details: [Boyd and Vandenberghe, 2004, Ch.11], [Nocedal and Wright, 2006, Ch. 19]

Linear Program (LP)

Linear program in standard form

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned} \quad (10)$$

- ▶ Linear objective with $\mathbf{c} \in \mathbb{R}^n$
- ▶ Linear equality constraints with $\mathbf{A} \in \mathbb{R}^{p \times n}, \mathbf{b} \in \mathbb{R}^p$
- ▶ Positivity inequality constraints.



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

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

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

Linear Program (LP)

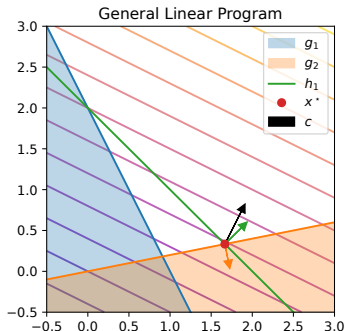
General formulation for LP

$$\min_{\mathbf{x} \in \mathbb{R}^n} \mathbf{c}^T \mathbf{x} \quad (11)$$

$$\text{s.t. } \mathbf{G}\mathbf{x} \leq \mathbf{h}$$

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

- ▶ 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 $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$:

Linear Program (LP)

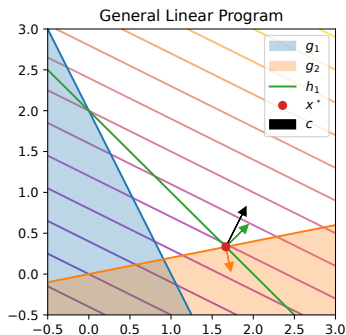
General formulation for LP

$$\min_{\mathbf{x} \in \mathbb{R}^n} \mathbf{c}^T \mathbf{x} \quad (11)$$

$$\text{s.t. } \mathbf{G}\mathbf{x} \leq \mathbf{h}$$

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

- ▶ 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 $\mathbf{x} \geq \mathbf{0}$: $\mathbf{G} = -\mathbf{I}_n$, $\mathbf{h} = \mathbf{0}$
- ▶ Simplex $\{\mathbf{x} | \mathbf{x} \geq \mathbf{0}, \sum_i x_i = 1\}$: $\mathbf{A} = [1, \dots, 1]$, $\mathbf{b} = [1]$, $\mathbf{G} = -\mathbf{I}_n$, $\mathbf{h} = \mathbf{0}$
- ▶ Box constraints $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$: $\mathbf{G} = \begin{bmatrix} \mathbf{I}_n \\ -\mathbf{I}_n \end{bmatrix}$, $\mathbf{h} = \begin{bmatrix} \mathbf{u} \\ \mathbf{l} \end{bmatrix}$

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

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

- ▶ 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 Kronecker 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 :

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

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 $\mathbf{A}_1 \mathbf{x} = \mathbf{a}$:

$$\mathbf{A}_1 = \mathbf{I}_n \otimes \mathbf{1}_{1,d}$$

The matrix can be expressed simply with the Kronecker product \otimes

2. Express the column-wise equality constraints $\sum_i X_{i,j} = s_j, \forall j$ and $\mathbf{A}_2 \mathbf{x} = \mathbf{s}$:

$$\mathbf{A}_2 = \mathbf{1}_{1,n} \otimes \mathbf{I}_d$$

3. Express all the matrices in the general LP :

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{a} \\ \mathbf{s} \end{bmatrix}, \quad \mathbf{G} = -\mathbf{I}_{nd}, \quad \mathbf{h} = \mathbf{0}_{nd}$$

Reduction from general to standard problem

Reformulation to standard LP with positive variables

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

- ▶ We express $\mathbf{x} = \mathbf{x}^+ - \mathbf{x}^-$ as a difference of positive variables.
- ▶ Problem on the right can be reformulated as standard LP.
- ▶ The positive variable $\mathbf{s} \geq \mathbf{0}$ is used to recover an equality constraint.
- ▶ The standard problem optimizes over $\tilde{\mathbf{x}} = [\mathbf{x}^{+\top}, \mathbf{x}^{-\top}, \mathbf{s}^\top]^\top \geq \mathbf{0}$.
- ▶ The matrix $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ can be recovered from \mathbf{A} , \mathbf{G} , \mathbf{b} and \mathbf{h} .
- ▶ 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

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

Primal VS Dual

- ▶ The problem permute their variables and constraints.
- ▶ When there is strict duality (problem has a solution) duality gap is 0:

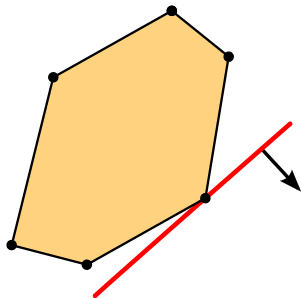
$$\mathbf{c}^\top \mathbf{x}^* = -\mathbf{b}^\top \mathbf{v}^*$$

- ▶ Finding \mathbf{x}^* from \mathbf{v}^* and vice versa:

1. Find components of \mathbf{x}^* equal to 0 from the equality $(\mathbf{A}^\top \mathbf{v}^* - \mathbf{c})^\top \mathbf{x}^* = 0$.
2. Solve the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ for the non-zero components of \mathbf{x}^* .

Solution of the standard LP

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



Property of the solution

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

Robust regression with Least Absolute Deviation

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

- ▶ 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 $|\mathbf{H}\mathbf{x} - \mathbf{y}| \leq \mathbf{f}$:

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

Robust regression with Least Absolute Deviation

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

- ▶ 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 $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T$:

$$\begin{aligned} \min_{\mathbf{w}, \mathbf{e}^+, \mathbf{e}^-} \quad & \mathbf{1}_n^T \mathbf{e}^+ + \mathbf{1}_n^T \mathbf{e}^- \\ \text{s.t.} \quad & \mathbf{X}\mathbf{w} - \mathbf{y} = \mathbf{e}^+ - \mathbf{e}^- \\ & \mathbf{e}^+ \geq \mathbf{0}_n, \mathbf{e}^- \geq \mathbf{0}_n \end{aligned}$$

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

$$\begin{aligned} \min_{\mathbf{w}, \mathbf{f}} \quad & \mathbf{1}_n^T \mathbf{f} \\ \text{s.t.} \quad & \mathbf{X}\mathbf{w} - \mathbf{y} \leq \mathbf{f}, \quad -\mathbf{X}\mathbf{w} + \mathbf{y} \leq \mathbf{f} \\ & \mathbf{f} \geq \mathbf{0}_n \end{aligned}$$

L1 Support Vector Machines

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{R}^d} \quad & \sum_{i=1}^n \max(0, 1 - y_i \mathbf{x}_i^T \mathbf{w}) \\ \text{s.t.} \quad & \|\mathbf{w}\|_1 \leq \beta \end{aligned} \quad (12)$$

- ▶ 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 :

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

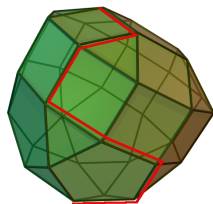
- ▶ The corresponding general LP problem with $\mathbf{x} = [\mathbf{w}^{+T}, \mathbf{w}^{-T}, \mathbf{f}]^T$ has the following matrices:

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

Simplex Algorithm

Main idea behind the simplex

- ▶ Initialize with a basic feasible solution $x^{(0)}$ (on a vertex or extreme point of the polytope).
- ▶ Update the solution to decrease the loss at each iteration.
- ▶ Use the sparsity of 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')`

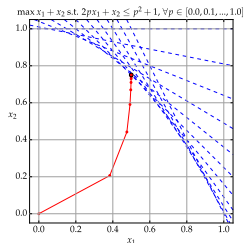
More details: [Vanderbei et al., 2015, part 1]

Interior point solver

Interior point method (IPM) for LP

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

- ▶ 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 for large problems.
- ▶ Use it on structured graph flow.

Interior point problem

- ▶ Better at early stopping.
- ▶ Usually faster on large problems.
- ▶ Most generic 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).

Wrappers available : <https://github.com/stephane-caron/lpsolvers>

Quadratic Program

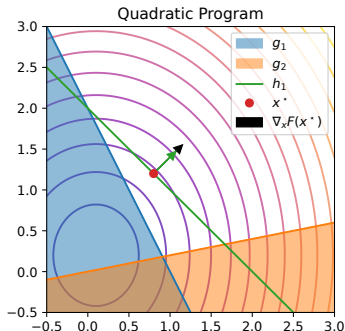
Optimization problem

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \quad (13) \\ \text{s.t.} \quad & \mathbf{G} \mathbf{x} \leq \mathbf{h} \\ & \mathbf{A} \mathbf{x} = \mathbf{b} \end{aligned}$$

- ▶ $\mathbf{Q} \in \mathbb{R}^{n \times n}$ 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 Σ .
- ▶ For a given portfolio \mathbf{x}
 - ▶ The expected gain (return) is : $\mathbf{p}^T \mathbf{x}$
 - ▶ The expected variance is : $\mathbf{x}^T \Sigma \mathbf{x}$
- ▶ The portfolio optimization can be expressed for a positive balance $b > 0$ as:

$$\min_{\mathbf{x} \in \mathbb{R}^n} \mathbf{x}^T \Sigma \mathbf{x} \quad (14)$$

$$\text{s.t. } \mathbf{1}_n^T \mathbf{x} = b \quad (15)$$

$$\mathbf{p}^T \mathbf{x} \geq r_{min} \quad (16)$$

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

Special Case : QP without constraints

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \quad (17)$$

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 : $\mathbf{Q}\mathbf{x}^* = -\mathbf{c}$
- ▶ If \mathbf{Q} is invertible and strictly positive definite then : $\mathbf{x}^* = -\mathbf{Q}^{-1}\mathbf{c}$
- ▶ 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 other courses).

Exercise 6: Least Square and Ridge

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

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

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

Special Case : QP without constraints

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \quad (17)$$

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 : $\mathbf{Q}\mathbf{x}^* = -\mathbf{c}$
- ▶ If \mathbf{Q} is invertible and strictly positive definite then : $\mathbf{x}^* = -\mathbf{Q}^{-1}\mathbf{c}$
- ▶ 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 other courses).

Exercise 6: Least Square and Ridge

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

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

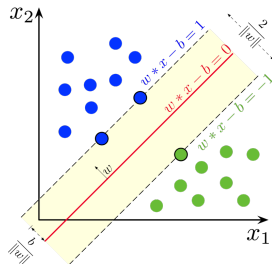
$$\mathbf{Q} = \mathbf{H}^T \mathbf{H}, \quad \mathbf{c} = -\mathbf{H}^T \mathbf{y}, \qquad \mathbf{Q} = \mathbf{H}^T \mathbf{H} + \lambda \mathbf{I}, \quad \mathbf{c} = -\mathbf{H}^T \mathbf{y},$$

Support Vector Machines (1)

Hard margin SVM [Cortes and Vapnik, 1995]

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{x}_i^T \mathbf{w} + b) \geq 1 \end{aligned} \quad (18)$$

- ▶ 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

$$\min_{\mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}} C \sum_i \max(0, 1 - y_i(\mathbf{x}_i^T \mathbf{w} + b)) + \frac{1}{2} \|\mathbf{w}\|^2 \quad (19)$$

- ▶ 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

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}, \mathbf{z} \in \mathbb{R}^n} \quad & C \sum_i z_i + \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{x}_i^T \mathbf{w} + b) \geq 1 - z_i, \quad \forall i \\ & \mathbf{z} \geq \mathbf{0} \end{aligned} \quad (20)$$

- ▶ 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

$$\begin{aligned} \min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \quad & \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{Q} \boldsymbol{\alpha} - \mathbf{1}_n^T \boldsymbol{\alpha} \\ \text{s.t.} \quad & \mathbf{y}^T \boldsymbol{\alpha} = 0 \\ & \mathbf{0}_n \leq \boldsymbol{\alpha} \leq C \mathbf{1}_n \end{aligned} \quad (21)$$

- ▶ 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 : $\mathbf{w}^* = \sum_i y_i \alpha_i^* \mathbf{x}_i$.
- ▶ b^* 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

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \lambda \sum_i |w_i| \quad (22)$$

- ▶ 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 $\mathbf{w} = \mathbf{w}^+ - \mathbf{w}^-$

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

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

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

Lasso estimator

$$\min_{\mathbf{w}} \quad \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \lambda \sum_i |w_i| \quad (22)$$

- ▶ 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 $\mathbf{w} = \mathbf{w}^+ - \mathbf{w}^-$

$$\begin{aligned} \min_{\mathbf{w}^+, \mathbf{w}^-} \quad & \frac{1}{2} \|\mathbf{X}(\mathbf{w}^+ - \mathbf{w}^-) - \mathbf{y}\|^2 + \lambda \sum_i w_i^+ + w_i^- \\ \text{s.t.} \quad & \mathbf{w}^+ \geq \mathbf{0}, \mathbf{w}^- \geq \mathbf{0} \end{aligned}$$

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

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

Lasso estimator

$$\min_{\mathbf{w}} \quad \frac{1}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \lambda \sum_i |w_i| \quad (22)$$

- ▶ 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 $\mathbf{w} = \mathbf{w}^+ - \mathbf{w}^-$

$$\begin{aligned} \min_{\mathbf{w}^+, \mathbf{w}^-} \quad & \frac{1}{2} \|\mathbf{X}(\mathbf{w}^+ - \mathbf{w}^-) - \mathbf{y}\|^2 + \lambda \sum_i w_i^+ + w_i^- \\ \text{s.t.} \quad & \mathbf{w}^+ \geq \mathbf{0}, \quad \mathbf{w}^- \geq \mathbf{0} \end{aligned}$$

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

$$\mathbf{Q} = \begin{bmatrix} \mathbf{X}^T \mathbf{X} & -\mathbf{X}^T \mathbf{X} \\ -\mathbf{X}^T \mathbf{X} & \mathbf{X}^T \mathbf{X} \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} -\mathbf{X}^T \mathbf{y} + \lambda \mathbf{1}_d \\ \mathbf{X}^T \mathbf{y} + \lambda \mathbf{1}_d \end{bmatrix}, \quad \mathbf{G} = -\mathbf{I}_{2d}, \quad \mathbf{h} = \mathbf{0}_{2d}$$

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 $\mathcal{A}(\mathbf{x}^*)$.
- ▶ 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)

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

Principle of SMO

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

SMO Algorithm

Initialize feasible α

1. Find two components α_i and α_j that violate KKT conditions.
2. Solve the QP on only those components in closed form (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, can be very fast on structured problems (sparsity, SVM)
- ▶ **SMO** State of the art solver for SVM.

QP Solvers in Python

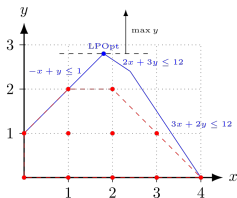
- ▶ **Numpy** (no constraints): (`np.linalg.solve` or `np.linalg.lstsq`).
- ▶ **quadprog** : Implements active set [Goldfarb and Idnani, 1983]
- ▶ **cvxopt** : Interior point solver for standard problems + wrapper for Mosek.
- ▶ **OSQP** : Operator splitting QP solver [Stellato et al., 2017].
- ▶ **Mosek**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

Benchmark available : <https://github.com/qpsolvers/qpbenchmark>

Integer Programming

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & F(\mathbf{x}) \\ \text{s.t.} \quad & h_j(\mathbf{x}) = 0 \quad \forall j = 1, \dots, p \\ & g_i(\mathbf{x}) \leq 0 \quad \forall i = 1, \dots, q. \end{aligned} \quad (23)$$

$\mathbf{x} \in \mathbb{Z}^n$



- ▶ Classical optimization problem with additional integer constraints $\mathbf{x} \in \mathbb{Z}^n$.
- ▶ Zero-one programming 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 (when 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

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

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:

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

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

Quadratically Constrained QP (QCQP)

Optimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} \quad \frac{1}{2} \mathbf{x}^T \mathbf{Q}_0 \mathbf{x} + \mathbf{c}_0^T \mathbf{x} \quad (24)$$

$$\text{s.t.} \quad \mathbf{x}^T \mathbf{Q}_i \mathbf{x} + \mathbf{c}_i^T \mathbf{x} \leq \mathbf{h}_i, \quad \forall i = 1, \dots, m$$

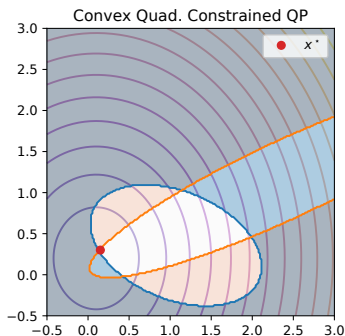
$$\mathbf{A} \mathbf{x} = \mathbf{b}$$

- ▶ If $\mathbf{Q}_0, \dots, \mathbf{Q}_m$ are positive definite then the problem is convex and can be solved with interior point.
- ▶ Nonconvex QCQP is NP-hard, because a constraint $x_i \in \{0, 1\}$ is recovered with:

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

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).



Quadratically Constrained QP (QCQP)

Optimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} \quad \frac{1}{2} \mathbf{x}^T \mathbf{Q}_0 \mathbf{x} + \mathbf{c}_0^T \mathbf{x} \quad (24)$$

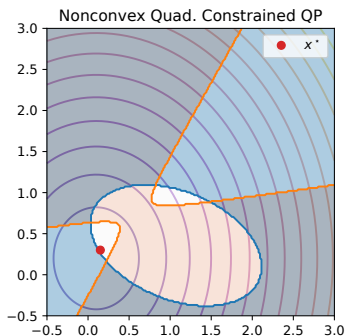
$$\text{s.t.} \quad \mathbf{x}^T \mathbf{Q}_i \mathbf{x} + \mathbf{c}_i^T \mathbf{x} \leq \mathbf{h}_i, \quad \forall i = 1, \dots, m$$
$$\mathbf{A} \mathbf{x} = \mathbf{b}$$

- ▶ If $\mathbf{Q}_0, \dots, \mathbf{Q}_m$ are positive definite then the problem is convex and can be solved with interior point.
- ▶ Nonconvex QCQP is NP-hard, because a constraint $x_i \in \{0, 1\}$ is recovered with:

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

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

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

- ▶ 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

$$\begin{aligned} \min_{\bar{\mathbf{x}}_k, \forall k, \mathbf{Z} \in \mathbb{R}^{N \times K}, \mathbf{D} \in \mathbb{R}^{N \times K}} \quad & \sum_{i,k} Z_{i,k} D_{i,k} & (25) \\ \text{s.t.} \quad & D_{i,k} = \|\mathbf{x}_i - \bar{\mathbf{x}}_k\|^2, \forall i, \forall k \\ & \mathbf{Z}\mathbf{1}_K = \mathbf{1}_N \\ & \mathbf{Z} \in \{0, 1\}^{N, K} \end{aligned}$$

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

Second Order Cone Programming (SOCP)

Optimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} \mathbf{c}^T \mathbf{x} \quad (26)$$

$$\text{s.t.} \quad \|\mathbf{A}_i \mathbf{x} - \mathbf{b}_i\|_2 \leq \mathbf{h}_i^T \mathbf{x} + d_i, \quad i = 1, \dots, m$$
$$\mathbf{A}_0 \mathbf{x} = \mathbf{b}_0$$

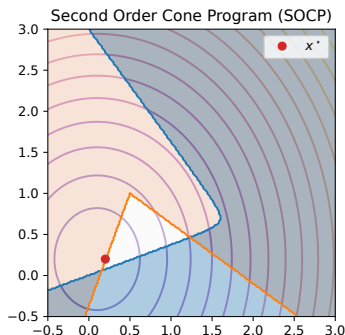
- ▶ The following constraint is called a Second order cone constraint:

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

- ▶ 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 (positive definite 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

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}, \mathbf{z} \in \mathbb{R}^n} \quad & C \sum_i z_i + \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{x}_i^T \mathbf{w} + b) \geq 1 - z_i + \gamma_i \left\| \boldsymbol{\Sigma}_i^{\frac{1}{2}} \mathbf{w} \right\|, \quad \forall i \\ & \mathbf{z} \geq \mathbf{0} \end{aligned} \tag{27}$$

- ▶ 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 $\boldsymbol{\Sigma}_i$ 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 problem above with $\gamma_i = \phi^{-1}(\kappa_i)$ where ϕ is the normal CDF.

Semi-Definite Programming

Optimization problem

$$\begin{aligned} \min_{\mathbf{X} \in \mathbb{S}^n} \quad & \langle \mathbf{X}, \mathbf{C} \rangle_{\mathbb{S}^n} \\ \text{s.t.} \quad & \langle \mathbf{X}, \mathbf{A}_i \rangle_{\mathbb{S}^n} = b_i, \quad i = 1, \dots, m \\ & \mathbf{X} \succeq 0 \end{aligned} \tag{28}$$

- ▶ \mathbb{S}^n is the set of $n \times n$ symmetric 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 $\mathbf{X} \succeq 0$ forces \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

Constrained optimization

- ▶ Constrained optimization is a very large field.
- ▶ Lagrangian and KKT conditions are the main tools to solve and check solutions.

Standard Problems (properties)

- ▶ Important properties
 - ▶ Linear or quadratic objective function.
 - ▶ Linear, quadratic or conic constraints.
 - ▶ Real or integer variables.
- ▶ Many existing generic solvers for those problems (commercial or free).

Solving DS and ML problems beyond gradient descent

- ▶ Constraints are important in many problems (fairness, robustness).
- ▶ You often have to model new ML problems depending on the constraints.
- ▶ Reformulation is key to use generic solvers (important skill).

References I



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, 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, 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, S. and Vandenberghe, L. (2004).

Convex optimization.







Cambridge university press.



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, C. and Vapnik, V. (1995).
Support-vector networks.
Machine learning, 20(3):273–297.
-  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, A., Chu, E., and Boyd, S. (2013).
ECOS: An SOCP solver for embedded systems.
In *European Control Conference (ECC)*, pages 3071–3076.
-  Goldfarb, D. and Idnani, A. (1983).
A numerically stable dual method for solving strictly convex quadratic programs.
Mathematical programming, 27(1):1–33.
-  Kantorovich, L. V. (1942).
On the translocation of masses.
In *Dokl. Akad. Nauk. USSR (NS)*, volume 37, pages 199–201.
-  Knowles, D. (2010).
Lagrangian duality for dummies.

References III



Nocedal, J. and Wright, S. (2006).

Numerical optimization.

Springer Science & Business Media.



Orlin, J. B. (1997).

A polynomial time primal network simplex algorithm for minimum cost flows.

Mathematical Programming, 78(2):109–129.



Park, J. and Boyd, S. (2017).

General heuristics for nonconvex quadratically constrained quadratic programming.

arXiv preprint arXiv:1703.07870.



Platt, J. (1998).

Sequential minimal optimization: A fast algorithm for training support vector machines.



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, 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, R. (1996).

Regression shrinkage and selection via the lasso.

Journal of the Royal Statistical Society: Series B (Methodological), 58(1):267–288.



Vandenberghe, L. (2010).

The cvxopt linear and quadratic cone program solvers.

Online: <http://cvxopt.org/documentation/coneprog.pdf>.



Vanderbei, R. J. et al. (2015).

Linear programming.

Springer.



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.