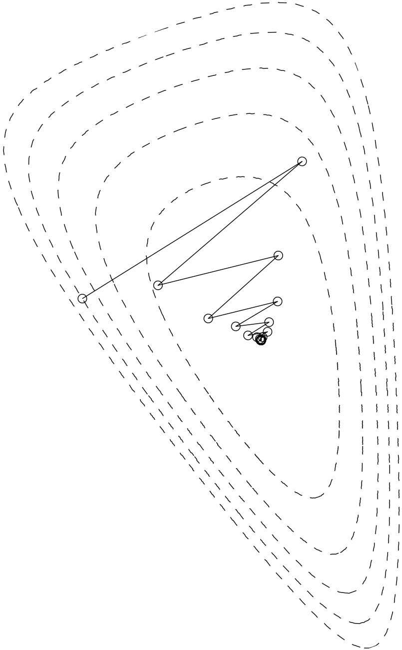


Optimization Algorithms

Convex Optimization

Convex, quasiconvex, unimodal, convex optimization problem, linear program (LP), standard form, simplex algorithm, LP-relaxation of integer linear programs, quadratic programming (QP), sequential quadratic programming

Marc Toussaint
Technical University of Berlin

Winter 2024/25

Function types

- A set $X \subseteq V$ is defined **convex** iff

$$\forall x, y \in X, a \in [0,1]: \; ax + (1-a)y \in X$$

- A function is defined **convex** iff

$$\forall x, y \in \mathbb{R}^n, a \in [0,1]: \; f(ax + (1-a)y) \le a\; f(x) + (1-a)\; f(y)$$

- A function is **quasiconvex** iff

$$\forall x, y \in \mathbb{R}^n, a \in [0,1]: \; f(ax + (1-a)y) \le \max\{f(x), f(y)\}$$

..alternatively, iff every sublevel set $\{x | f(x) \le \alpha\}$ is convex.

- We call a function **unimodal** iff it has only 1 local minimum, which is the global one convex $\subset$ quasiconvex $\subset$ unimodal $\subset$ general

Properties

- The sum of two confex functions $f_1(x) + f_2(x)$ is also convex
- A function $f \in \mathcal{C}^2$ convex $\quad\Leftrightarrow\quad \nabla^2 f(x)$ pos.-semidef. everywhere
- f convex $\Rightarrow$ sublevel sets $\{x : f(x) \le a\}$ are convex

Properties

- The sum of two confex functions $f_1(x) + f_2(x)$ is also convex
- A function $f \in \mathcal{C}^2$ convex $\Leftrightarrow$ $\nabla^2 f(x)$ pos.-semidef. everywhere
- f convex $\Rightarrow$ sublevel sets $\{x : f(x) \leq a\}$ are convex
- $l(\lambda) = \min_x L(x, \lambda)$ is concave! Point-wise minimization:
  - For each x, $L(x, \lambda)$ is linear in λ
  - Think of $L(x, \lambda)$ as a family of many linear functions
  - At each λ, pick the function with lowest value $\rightarrow$ concave
  - (Epigraph: The "region" $\{(x, y) : y \leq f(x)\}$ below a function; point-wise minimization $\leftrightarrow$ intersection of epigraphs.)

Convex Mathematical Program (CP)

- *Variant 1:* A mathematical program $\min_x \; f(x) \;$ s.t. $\; g(x) \le 0, \; h(x) = 0$ is convex iff f is convex and the feasible set is convex.

  Variant 2: A mathematical program $\min_x \; f(x) \;$ s.t. $\; g(x) \le 0, \; h(x) = 0$ is convex iff f and every g_i are convex and h is linear.

  - Variant 2 is the stronger and the default definition
  - In variant 1, only $\{x : h(x) = 0\}$ needs to be *linear*, and $\{x : g(x) \le 0\}$ needs to be convex

Linear and Quadratic Programs

- **Linear Program** (LP)

$$\min_x \; c^\top x \;\text{ s.t. }\; Gx \le h,\; Ax = b$$

LP in standard form

$$\min_x \; c^\top x \;\text{ s.t. }\; x \ge 0,\; Ax = b$$

- **Quadratic Program** (QP)

$$\min_x \; \frac{1}{2}x^\top Q x + c^\top x \;\text{ s.t. }\; Gx \le h,\; Ax = b$$

where Q is positive definite.

(This is different to a Quadratically Constraint Quadratic Programs (QCQP))

Transforming an LP problem into standard form

- LP problem:

$$\min_x \ c^\top x \ \text{ s.t. } \ Gx \le h, \ Ax = b$$

- Introduce **slack variables**:

$$\min_{x,\xi} \ c^\top x \ \text{ s.t. } \ Gx + \xi = h, \ Ax = b, \ \xi \ge 0$$

- Express $x = x^+ - x^-$ with $x^+, x^- \ge 0$:

$$\min_{x^+,x^-,\xi} \ c^\top(x^+ - x^-)$$

$$\text{s.t. } \ G(x^+ - x^-) + \xi = h, \ A(x^+ - x^-) = b, \ \xi \ge 0, \ x^+ \ge 0, \ x^- \ge 0$$

where $(x^+, x^-, \xi) \in \mathbb{R}^{2n+m}$

- Now this is conform with the standard form with $\tilde{x} = (x^+, x^-, \xi)$, $\tilde{A} = \begin{pmatrix} G & -G & \mathbf{I} \\ A & -A & 0 \end{pmatrix}$, $\tilde{b} = (h, b)$

$$\min_{\tilde{x}} \ c^\top \tilde{x} \ \text{ s.t. } \ \tilde{x} \ge 0, \ \tilde{A}\tilde{x} = \tilde{b}$$

- A **slack variable** is a variable that is added to an inequality constraint to transform it into an equality. Introducing a slack variable replaces an inequality constraint with an equality constraint and a non-negativity constraint on the slack variable (wikipedia)

Example LPs

See the exercises 4.8-4.20 of Boyd & Vandenberghe!

Example QP

- Support Vector Machines. Primal problem:

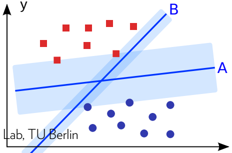
$$\min_{\beta,\xi} \|\beta\|^2 + C\sum_{i=1}^n \xi_i \quad \text{s.t.} \quad y_i(x_i^\top \beta) \geq 1 - \xi_i\ , \quad \xi_i \geq 0$$

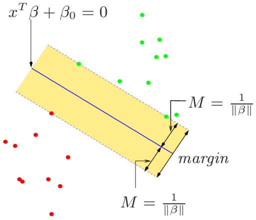
Dual problem:

$$l(\alpha,\mu) = \min_{\beta,\xi} L(\beta,\xi,\alpha,\mu) = -\frac{1}{4}\sum_{i=1}^n \sum_{i'=1}^n \alpha_i \alpha_{i'} y_i y_{i'} \hat{x}_i^\top \hat{x}_{i'} + \sum_{i=1}^n \alpha_i$$

$$\max_{\alpha,\mu} l(\alpha,\mu) \quad \text{s.t.} \quad 0 \leq \alpha_i \leq C$$

(See ML lecture 5:13 for a derivation.)





Finding the optimal discriminative function [from ML lecture]

- The constrained problem

$$\min_{\beta,\xi} \|\beta\|^2 + C \sum_{i=1}^n \xi_i \ \text{ s.t. }\ y_i(x_i^\top \beta) \ge 1 - \xi_i \ , \quad \xi_i \ge 0$$

  is a **quadratic program** and can be reformulated as the dual problem, with dual parameters α_i that indicate whether the constraint $y_i(x_i^\top \beta) \ge 1 - \xi_i$ is active. The dual problem is **convex**. SVM libraries use, e.g., CPLEX to solve this.

- For all inactive constraints ($y_i(x_i^\top \beta) \ge 1$) the data point (x_i, y_i) does not directly influence the solution β^*. Active points are support vectors.

[from ML lecture]

- Let (x, ξ) be the primal variables, (α, μ) the dual, we derive the dual problem:

$$\min_{\beta,\xi} \|\beta\|^2 + C \sum_{i=1}^n \xi_i \text{ s.t. } y_i(x_i^\top \beta) \geq 1 - \xi_i \;, \quad \xi_i \geq 0 \tag{1}$$

$$L(\beta, \xi, \alpha, \mu) = \|\beta\|^2 + C \sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i [y_i(x_i^\top \beta) - (1 - \xi_i)] - \sum_{i=1}^n \mu_i \xi_i \tag{2}$$

$$\partial_\beta L \overset{!}{=} 0 \quad \Rightarrow \quad 2\beta = \sum_{i=1}^n \alpha_i y_i x_i \tag{3}$$

$$\partial_\xi L \overset{!}{=} 0 \quad \Rightarrow \quad \forall_i : \; \alpha_i = C - \mu_i \tag{4}$$

$$l(\alpha, \mu) = \min_{\beta,\xi} L(\beta, \xi, \alpha, \mu) = -\tfrac{1}{4} \sum_{i=1}^n \sum_{i'=1}^n \alpha_i \alpha_{i'} y_i y_{i'} \hat{x}_i^\top \hat{x}_{i'} + \sum_{i=1}^n \alpha_i \tag{5}$$

$$\max_{\alpha,\mu} l(\alpha, \mu) \text{ s.t. } 0 \leq \alpha_i \leq C \tag{6}$$

- 2: Lagrangian (with negative Lagrange terms because of $\geq$ instead of $\leq$)
- 3: the optimal β^* depends only on $x_i y_i$ for which $\alpha_i > 0 \rightarrow$ support vectors
- 5: This assumes that $x_i = (1, \hat{x}_i)$ includes the constant feature 1 (so that the statistics become centered)
- 6: This is the dual problem. $\mu_i \geq 0$ implies $\alpha_i \leq C$
- Note: the dual problem only refers to $\hat{x}_i^\top \hat{x}_i \;\rightarrow$ **kernelization**

Algorithms for Convex Programming

- All the ones we discussed for non-linear optimization!
  - log barrier ("interior point method", "[central] path following")
  - augmented Lagrangian
  - primal-dual Newton

- The simplex algorithm, walking on the constraints

  (The emphasis in the notion of *interior* point methods is to distinguish from constraint walking methods.)

Simplex Algorithm

Simplex Algorithm

Georg Dantzig (1947)

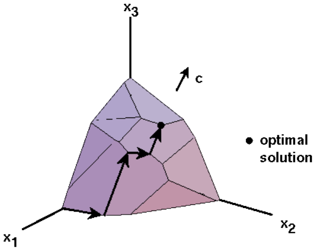
Note: Not to confuse with the Nelder-Mead method (downhill simplex method)

- Consider an LP

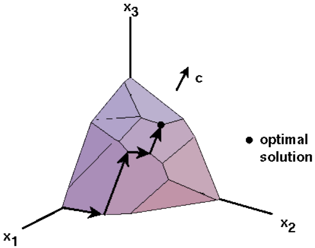
$$\min_x \; c^\top x \;\; \text{s.t.} \;\; Gx \le h, \; Ax = b$$

- Note that in a linear program an optimum is always located at a vertex

(If there are multiple optimal, at least one of them is at a vertex.)



Simplex Algorithm



- The Simplex Algorithm walks along the edges of the **polytope**, at every vertex choosing the edge that decreases $c^\top x$ most
- This either terminates at a vertex, or leads to an unconstrained edge ($-\infty$ optimum)

- In practise this procedure is done by “pivoting on the simplex tableaux”

Simplex Algorithm vs. Interior methods

- The simplex algorithm is often efficient, but in worst case exponential in n and m!
  (In high dimensions constraints may intersect and form edges and vertices in a combinatorial way.)

Simplex Algorithm vs. Interior methods

- The simplex algorithm is often efficient, but in worst case exponential in n and m!
  (In high dimensions constraints may intersect and form edges and vertices in a combinatorial way.)

- Sitting on an edge/face/vertex $\leftrightarrow$ hard decisions on which constraints are active
  - The simplex algorithm is sequentially making decisions on which constraints might be active – by walking through this combinatorial space.
- Interior point methods do exactly the opposite:
  - They "postpone" (or relax) hard decisions about active/non-active constraints,
  - approach the optimal vertex from the inside of the polytope; avoiding the polytope surface
  - have polynomial worst-case guaranteed

Simplex Algorithm vs. Interior methods

- The simplex algorithm is often efficient, but in worst case exponential in n and m!
  (In high dimensions constraints may intersect and form edges and vertices in a combinatorial way.)

- Sitting on an edge/face/vertex $\leftrightarrow$ hard decisions on which constraints are active
  - The simplex algorithm is sequentially making decisions on which constraints might be active – by walking through this combinatorial space.
- Interior point methods do exactly the opposite:
  - They "postpone" (or relax) hard decisions about active/non-active constraints,
  - approach the optimal vertex from the inside of the polytope; avoiding the polytope surface
  - have polynomial worst-case guaranteed
- Historically:
  - Before 50ies: Penalty and barrier methods methods were standard
  - From 50s: Simplex Algorithm
  - From 70s: More theoretical understanding, interior point methods (and more recently Augmented Lagrangian methods) again more popular

Sequential Quadratic Programming

Quadratic Programming

$$\min_x \frac{1}{2}x^\top Q x + c^\top x \text{ s.t. } Gx \le h,\ Ax = b$$

- Efficient Algorithms:
  - Interior point (log barrier)
  - Augmented Lagrangian

- Highly relevant applications:
  - Support Vector Machines
  - Similar types of max-margin modelling methods

Sequential Quadratic Programming (SQP)

- SQP is another standard method for **non-linear** programs
  - It can be understood as generalization of the Newton method to the constrained case:
  - The Newton method for $\min_x f(x)$ approximates f using 2nd-order Taylor, and computes the optimal step δ^* for this approximation
  - SQP approximates costs f and constraints g, h using Taylor, and then computes the optimal step δ^* for this approximation

Sequential Quadratic Programming (SQP)

- SQP is another standard method for **non-linear** programs
  - It can be understood as generalization of the Newton method to the constrained case:
  - The Newton method for $\min_x f(x)$ approximates f using 2nd-order Taylor, and computes the optimal step δ^* for this approximation
  - SQP approximates costs f and constraints g, h using Taylor, and then computes the optimal step δ^* for this approximation
- In each iteration we consider Taylor approximations:
  - 2nd order for: $f(x+\delta) \approx f(x) + \nabla f(x)^\top \delta + \frac{1}{2}\delta^\top H \delta$
  - 1st order for: $g(x+\delta) \approx g(x) + \nabla g(x)^\top \delta \ , \quad h(x+\delta) \approx h(x) + \nabla h(x)^\top \delta$
- Then we compute the optimal step δ^* solving the QP:

$$\min_\delta \ f(x) + \nabla f(x)^\top \delta + \frac{1}{2}\delta^\top \nabla^2 f(x) \delta \ \text{ s.t. } \ g(x) + \nabla g(x)^\top \delta \le 0 \ , \quad h(x) + \nabla h(x)^\top \delta = 0$$

Sequential Quadratic Programming (SQP)

- If f *were* a 2nd-order polynomial and g, h linear, then δ^* would jump directly to the optimum
- Otherwise, backtracking line search

Sequential Quadratic Programming (SQP)

- If f *were* a 2nd-order polynomial and g, h linear, then δ^* would jump directly to the optimum
- Otherwise, backtracking line search

- Note: Solving each QP to compute the search step δ^* requires a constrained solver, which itself might have two nested loops (e.g. using log-barrier or AugLag) $\rightarrow$ three nested loops
- **But:** To solve the QP-step, you need no queries of the original problem!
  $\rightarrow$ SQP can be query efficient. It invests in solving an approximate QP to minimize querying the original problem
- Potentially more prone to non-smoothness (local Taylor might be misleading)

Baseline methods for constrained optimization

- We now learnt about four baseline methods to tackle constrained optimization:
  - Log barrier method
  - Augmented Lagrangian method
  - Primal-dual Newton
  - Sequential Quadratic Programming

LP-relaxations of discrete problems*

Integer linear programming (ILP)

- An integer linear program (for simplicity binary) is

$$\min_x \; c^\top x \; \text{ s.t. } \; Ax = b, \; x_i \in \{0,1\}$$

- Examples:
  - Travelling Salesman: $\min_{x_{ij}} \sum_{ij} c_{ij} x_{ij}$ with $x_{ij} \in \{0,1\}$ and constraints $\forall_j : \sum_i x_{ij} = 1$ (columns sum to 1), $\forall_j : \sum_i x_{ji} = 1$, $\forall_{ij} : t_j - t_i \le n - 1 + n x_{ij}$ (where t_i are additional integer variables).
  - MaxSAT problem: In conjunctive normal form, each clause contributes an additional variable and a term in the objective function; each clause contributes a constraint
  - Search the web for *The Power of Semidefinite Programming Relaxations for MAXSAT*

LP relaxations of integer linear programs

- Instead of solving
$$\min_x c^\top x \;\text{ s.t. }\; Ax = b,\; x_i \in \{0,1\}$$
we solve
$$\min_x c^\top x \;\text{ s.t. }\; Ax = b,\; x \in [0,1]$$

- Clearly, the relaxed solution will be a *lower bound* on the integer solution (sometimes also called "outer bound" because $[0,1] \supset \{0,1\}$)

- Computing the relaxed solution is interesting
  - as an "approximation" or initialization to the integer problem
  - in cases where the optimal relaxed solution happens to be integer
  - for using the lower bound for **branch-and-bound** tree search over the discrete variable

Example*: MAP inference in MRFs

- Given integer random variables x_i, $i = 1, .., n$, a pairwise Markov Random Field (MRF) is defined as

$$f(x) = \sum_{(ij)\in E} f_{ij}(x_i, x_j) + \sum_i f_i(x_i)$$

where E denotes the set of edges. Problem: find $\max_x f(x)$.
(Note: any general (non-pairwise) MRF can be converted into a pair-wise one, blowing up the number of variables)

- Reformulate with indicator variables

$$b_i(x) = [x_i = x] \ , \quad b_{ij}(x, y) = [x_i = x] \ [x_j = y]$$

These are $nm + |E|m^2$ binary variables

- The indicator variables need to fulfil the constraints

$$b_i(x), b_{ij}(x, y) \in \{0, 1\}$$

$$\sum_x b_i(x) = 1 \qquad \text{because } x_i \text{ takes eactly one value}$$

$$\sum_y b_{ij}(x, y) = b_i(x) \qquad \text{consistency between indicators}$$

Example*: MAP inference in MRFs

- Finding $\max_x f(x)$ of a MRF is then equivalent to

$$\max_{b_i(x), b_{ij}(x,y)} \sum_{(ij)\in E} \sum_{x,y} b_{ij}(x,y)\ f_{ij}(x,y) + \sum_i \sum_x b_i(x)\ f_i(x)$$

such that

$$b_i(x), b_{ij}(x,y) \in \{0,1\}\ , \quad \sum_x b_i(x) = 1\ , \quad \sum_y b_{ij}(x,y) = b_i(x)$$

- The LP-relaxation replaces the constraint to be

$$b_i(x), b_{ij}(x,y) \in [0,1]\ , \quad \sum_x b_i(x) = 1\ , \quad \sum_y b_{ij}(x,y) = b_i(x)$$

This set of feasible b's is called **marginal polytope** (because it describes the a space of "probability distributions" that are marginally consistent (but not necessarily globally normalized!))

Example*: MAP inference in MRFs

- Solving the original MAP problem is NP-hard
  Solving the LP-relaxation is really efficient

- If the solution of the LP-relaxation turns out to be integer, we've solved the originally NP-hard problem!
  If not, the relaxed problem can be discretized to be a good initialization for discrete optimization

- For binary attractive MRFs (a common case) the solution will always be integer

Conclusions

- Convex Problems are an important special case
  - Convergence of backtracking line search $\leftarrow$ bounded Hessian $\rightarrow$ convexity
  - Some applications are convex
- Algorithms for convex programs are same as we discussed before
- Baseline methods for constrained optimization:
  - Log barrier method
  - Augmented Lagrangian method
  - Primal-dual Newton
  - Sequential Quadratic Programming