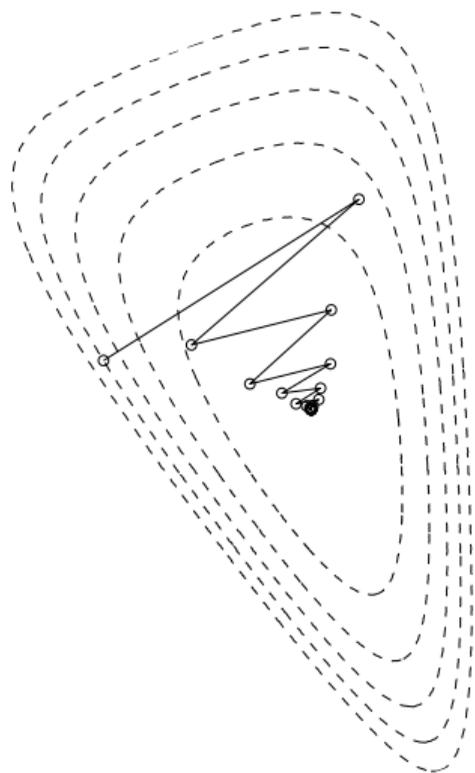


Optimization Algorithms

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 2020/21

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

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

- A function is **quasiconvex** iff

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

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

- We call a function **unimodal** iff it has only 1 local minimum, which is the global minimum

convex \subset quasiconvex \subset unimodal \subset general

Convex Program

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

Variant 2: A mathematical program $\min_x f(x)$ s.t. $g(x) \leq 0, h(x) = 0$ is convex if 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 $\{x : g(x) \leq 0\}$ needs to be convex

Linear and Quadratic Programs

- Linear Program (LP)

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

LP in standard form

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

- Quadratic Program (QP)

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

where Q is positive definite.

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

Transforming an LP problem into standard form

- LP problem:

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

- Introduce **slack variables**:

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

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

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

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

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

- Now this is conform with the standard form (replacing $(x^+, x^-, \xi) \equiv x$, etc)

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

- 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

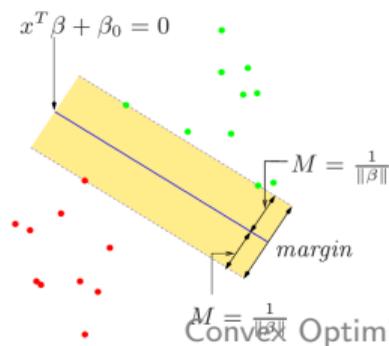
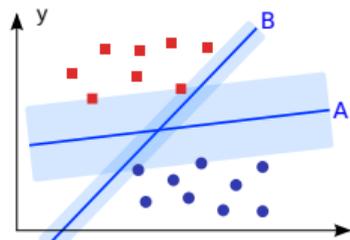
- Support Vector Machines. Primal problem:

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

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.)



Algorithms for Convex Programming

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

- 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

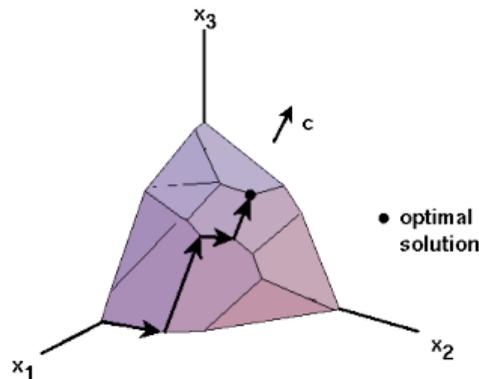
Georg Dantzig (1947)

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

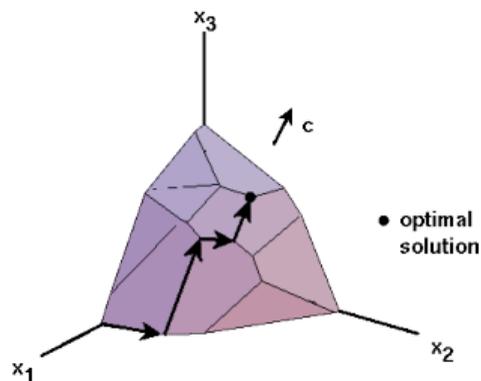
- We consider an LP in standard form

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

- 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 $c^T x$ 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
 - avoid the need to search through a combinatorial space of constraint activities
 - 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
 - avoid the need to search through a combinatorial space of constraint activities
 - 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

LP-relaxations of discrete problems

Integer linear programming (ILP)

- An integer linear program (for simplicity binary) is

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

- 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), $\forall_j : \sum_i x_{ji} = 1$, $\forall_{ij} : t_j - t_i \leq n - 1 + nx_{ij}$ (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

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

we solve

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

- 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 $x_i, i = 1, \dots, n$, 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

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

$$\sum_x b_i(x) = 1$$

because x_i takes exactly one value

$$\sum_y b_{ij}(x, y) = b_i(x)$$

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

Quadratic Programming

Quadratic Programming

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

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

- 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: $g(x + \delta) \approx g(x) + \nabla g(x)^\top \delta$, $h(x + \delta) \approx h(x) + \nabla h(x)^\top \delta$
- Then we compute the optimal step δ^* solving the QP:

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

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). In that case, SQP has three nested loops.

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

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
- We now learnt about five basic methods to tackle constrained optimization:
 - Log barrier method
 - Squared penalty method (approximate only)
 - Augmented Lagrangian method
 - Primal-dual Newton
 - Sequential Quadratic Programming