Constrained optimization

Antonio Frangioni

Department of Computer Science
University of Pisa
www.di.unipi.it/~frangio
frangio@di.unipi.it

Computational Mathematics for Learning and Data Analysis
Master in Computer Science – University of Pisa

Outline

Constrained optimization algorithms

- Algorithms for (P) $f_* = \min\{\, f(x) \;:\; x \in X \,\}$
- Standard form: $X = \{\, x \in \mathbb{R}^n \;:\; G(x) \leq 0\,,\; H(x) = 0 \,\}$
- Usually won't bother about $H(\cdot)$, just $G(x) \leq 0$ (**why?**)
- Only linear equalities $Ax = b$ almost dealt with already
- Mostly work with linear inequalities $Ax \leq b$
- Some references to nonlinear convex case $G(x) \leq 0$
- Basically ignore the nonlinear nonconvex case
  (you don't do a lot of that in Machine Learning) $\Longrightarrow$
  no equality constraints except if they are affine (**why?**)
- Important point: exploiting special structures in the constraints
  (only basic hints given, there is a lot more of that)

Outline

Quadratic problem with linear equality constraints

- Equality-constrained QP: (P) $\min\{ \frac{1}{2} x^T Q x + q x \ : \ A x = b \}$
  $A \in \mathbb{R}^{m \times n}$ with $\text{rank}(A) = m < n$, rows of A linearly independent (**why?**)
- $Q \succeq 0$, otherwise $v(P) = -\infty$ "almost always"

Exercise: derive conditions for $v(P) > -\infty$ when $Q \not\succeq 0$

- Just solve the KKT system (a.k.a. normal equations)

$$\begin{array}{ll} \text{(a)} & \\ \text{(b)} & \end{array} \quad \begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x \\ \mu \end{bmatrix} = \begin{bmatrix} -q \\ b \end{bmatrix} \quad \text{"only linear algebra"}$$

- Basic step in many cases $\Longrightarrow$ have to do that efficiently
- System symmetric but indefinite (lots of 0 eigenvalues)
- Clearly Federico's playground, let's just hint at some possible ways
- Just go and solve it by direct or iterative methods:
  - indefinite factorization of the matrix (may reduce sparsity)
  - Krylov-type iterative methods (GMRES, ...)
- Or try to exploit the large-scale structure (saddle-point system)

Solving the KKT system

- Reduced KKT: Q nonsingular $\Longrightarrow$ multiply (a) by AQ^{-1} + (b) $\Longrightarrow$
  $$[\, AQ^{-1}A^T \,]\mu = -b - AQ^{-1}q \wedge x = -Q^{-1}(\, A^T\mu + q \,)$$
  $0 \preceq AQ^{-1}A^T = M \in \mathbb{R}^{m\times m}$
- Null Space Methods: $A = [\, A_B \,,\, A_N \,]$, $x = [\, x_B \,,\, x_N \,]$, $det(A_B) \neq 0$
  $\Longrightarrow$ (b) $\equiv x_B = A_B^{-1}(\, b - A_N x_N \,) \Longrightarrow x = Dx_N + d$ with
  $$d = \begin{bmatrix} b \\ 0 \end{bmatrix} \;,\; D = \begin{bmatrix} -A_B^{-1}A_N \\ I \end{bmatrix} \in \mathbb{R}^{m\times n-m}$$
  basis of null space of A $\equiv AD = 0$
- Multiply (a) by $D^T \Longrightarrow D^TQx - D^TA^T\mu =$
  $D^TQ(\, Dx_N + d \,) = -D^Tq \Longrightarrow [\, D^TQD \,]x_N = -D^T(\, Qd + q \,)$
  $0 \preceq D^TQD = H \in \mathbb{R}^{n-m\times n-m}$ reduced hessian (and Q can be singular)
- Can be generalized to any basis of null space of A
- Both M and H can be very dense even if A, Q sparse
- Heuristics to permute rows to improve sparsity, proper choices of D, ...
- Iterative methods to solve the systems without forming M and H
  (Preconditioned Conjugate Method, appropriate preconditioners ...)

Outline

Projected gradient method

- Nonlinear problem with linear constraints: (P) $\min\{\, f(x) \,:\, Ax \leq b \,\}$
- Feasible solution x, $D_X(x) = \{\, d \in \mathbb{R}^n \,:\, A_{\mathcal{A}(x)} d \leq 0 \,\}$ (**check**)
- $-\nabla f(x) \in D_X(x) \Longrightarrow$ line search along $d = -\nabla f(x)$
- Special case: $\mathcal{A}(x) = \emptyset \Longrightarrow D_X(x) = \mathbb{R}^n$
- If not, first project $-\nabla f(x)$ upon $\partial D_X(x)$ and then do line search
- $\bar{A} = A_{\mathcal{A}(x)}$, $\partial D_X(x) = \{\, d \in \mathbb{R}^n \,:\, \bar{A} d = 0 \,\}$, projection:
  $\min\{\, \frac{1}{2} \| \nabla f(x) + d \|_2^2 = \frac{1}{2} d^T I d + \nabla f(x) d \,:\, \bar{A} d = 0 \,\} \Longrightarrow (I \succ 0)$
  $[\, \bar{A} \bar{A}^T \,] \mu = -\bar{A} \nabla f(x) \;\wedge\; d = -\bar{A}^T \mu - \nabla f(x)$ (reduced KKT, **check**)
- $\bar{A}$ full row rank $\Longrightarrow$ $\bar{A} \bar{A}^T$ nonsingular $\Longrightarrow$
  $\mu = -[\, \bar{A} \bar{A}^T \,]^{-1} \bar{A} \nabla f(x)$, $d = (\, I - \bar{A}^T [\, \bar{A} \bar{A}^T \,]^{-1} \bar{A} \,)(\, -\nabla f(x) \,)$
- $d = 0$ may happen: good if $\mu \geq 0$ (**why?**), un-good otherwise
- $d = 0$ surely happens if $\bar{A} \in \mathbb{R}^{n \times n}$ (x a vertex, $\mathcal{A}(x)$ a base)
- $\bar{A}$ full row rank not true in general (degenerate vertex . . .)

Projected gradient method

procedure $x = PGM\,(\,f\,,\,A\,,\,b\,,\,x\,,\,\varepsilon\,)\;\{$
  for(; ;) {
    $B \leftarrow$ maximal $\subseteq \mathcal{A}(x)$ s.t. $rank(A_B) = |\,B\,|$;
    for(; ;) {
      $d \leftarrow (\,I - A_B^T[\,A_B A_B^T\,]^{-1} A_B\,)(\,-\nabla f(x)\,)$;
      if($\langle\,\nabla f(x)\,,\,d\,\rangle \leq \varepsilon$) **then** {
        $\mu_B \leftarrow -[\,A_B A_B^T\,]^{-1} A_B \nabla f(x)$; $\mu_i \leftarrow 0\;\forall\, i \notin B$;
        if($\mu_B \geq 0$) **then return**;
        $h \leftarrow \min\{\, i \in B \;:\; \mu_i < 0\,\}$; $B \leftarrow B \setminus \{\, h\,\}$; **continue**;
        };
      $\bar{\alpha} \leftarrow \min\{\, \alpha_i = (\,b_i - A_i x\,)/A_i d \;:\; A_i d > 0\;,\; i \notin B\,\}$;
      if($\bar{\alpha} > 0$) **then break**;
      $k \leftarrow \min\{\, i \notin B \;:\; A_i d > 0 \;:\; \alpha_i = 0\,\}$; $B \leftarrow B \cup \{\, k\,\}$;
      };
    $\alpha \leftarrow$ Line_Search(f , x , d , $\bar{\alpha}$); $x \leftarrow x + \alpha d$;
    };
  }

- $\alpha_i = \max\{\,\alpha \;:\; A_i(\,x + \alpha d) \leq b_i\,\}$ if is $< \infty$ (**check**)
- f linear + some streamlining $\rightsquigarrow$ primal simplex method
- pesky part: handling of linear independence

Projected gradient method (cont.d)

- Maximal B easy to get via a greedy algorithm
- $d = 0 \wedge \mu \geq 0 \Longrightarrow x$ optimal (**why?**)
- $d = 0 \wedge \exists\, h \in B$ s.t. $\mu_h < 0 \Longrightarrow$
  $\exists\, x' \in \{\, x \in \mathbb{R}^n \,:\, A_{B\setminus\{h\}} x = b_{B\setminus\{h\}} \,,\; A_h x \leq b_h \,\}$ s.t. $f(x') < f(x)$
  $\Longrightarrow B \leftarrow B \setminus \{\, h \,\} +$ line search $=$ descent
  Proof: $\exists\, d$ s.t. $A_{B\setminus\{h\}} d = 0 \wedge A_h d < 0$ (**why?**) $\Longrightarrow$
  $\langle\, \nabla f(x) \,,\, d \,\rangle = \langle\, -\mu A_B \,,\, d \,\rangle = -\mu_h A_h d < 0$
- $d \neq 0$ descent direction: $H = I - A_B^T [\, A_B A_B^T \,]^{-1} A_B$ symmetric and idempotent $HH = H^T H = H$ (**check**) $\Longrightarrow \langle\, d \,,\, \nabla f(x) \,\rangle =$
  $\langle\, -H\nabla f(x) \,,\, \nabla f(x) \,\rangle = -\nabla f(x)^T H \nabla f(x) = -(\, H\nabla f(x) \,)^T H \nabla f(x) < 0$
  $\Longrightarrow$ globally convergent (**why? how?**) provided the inner loop ends
- Inner loop handles degenerate steps: B changes, x does not
- Inner loop explores $\neq B \subset \mathcal{A}(x)$ s.t. A_B full rank (if $A_{\mathcal{A}(x)}$ is not)
- Min entering/leaving $i \equiv$ Bland's anti-cycle rule $\Longrightarrow$ finite termination
  typically irrelevant in practice (constraints perturbation, ...)

Projected gradient method (cont.d)

- Could visit all 2^m possible different B (not good news)
  but it never even remotely happens in practice
- However, actually suffers from degeneracy (say, vertex $\bar{x}$ with many $\neq B$)
  $\Longrightarrow$ many (degenerate) iterations
- Important: B "changes little" at every iteration $\Longrightarrow$ exploit results of previous iteration to speed-up this one (e.g., factorization of $A_B A_B^T \ldots$)
- Algorithm works provided one has a feasible x to start with
- "Not difficult": feasibility of a system of linear inequalities
- Can be solved e.g. by simplex method ("phase 0" $\ldots$)
- Can be extended to $G(x) \leq 0$ (remain in $int\, D_X(x) \ldots$)

Exercise: propose a practical way to ensure that $d \in int\, D_X(x)$

- When $B \equiv$ optimal face $\approx$ unconstrained steepest descent $\Longrightarrow$ slow
- Projected (quasi-)Newton does not work
- Once optimal face identified, a faster approach would be preferable

Projected gradient method: special forms of constraints

- Important: always exploit all the structure of your problem
- Nonlinear problem with box constraints: (P) $\min\{ f(x) : l \leq x \leq u \}$

```
procedure x = BCPGM ( f , l , u , x , ε ) {
  for( ; ; ) {
    d = −∇f(x); ᾱ = ∞;
    foreach( i = 1 . . . n s.t. d_i ≠ 0 ) do
      if( d_i < 0 ) then if( x_i = l_i ) then d_i = 0 else ᾱ ← min{ ᾱ , (x_i − l_i)/d_i }
                    else if( x_i = u_i ) then d_i = 0 else ᾱ ← min{ ᾱ , (u_i − x_i)/d_i }
    if( ⟨ ∇f(x) , d ⟩ ≤ ε ) then return;
    α ← Line_Search( f , x , d , ᾱ ); x ← x + αd;
    };
  }
```

- Projection very cheap $+rank(A_B) = |B|$ always (**why?**) $+$ initial feasible x straightforward
- Other cases where projection is easy: simplex constraints $\sum_{i=1}^{n} x_i = 1$ $x_i \geq 0 \quad i = 1, \ldots, n$ ($+$ general, Cartesian product of disjoint simplices)

Exercise: propose a fast method to project on a simplex constraint (hint: the Lagrangian problem is very simple), bring it to $O(n \log n)$

Outline

Active-set method for Quadratic Programs

- QP with linear constraints: (P) $\min\{ \frac{1}{2} x^T Q x + q x \ : \ A x \leq b \}$
- If one knew $\mathcal{A}(x_*)$, then it would be "just linear algebra"
  (and if my granny had had wheels she'd been a wheelbarrow)
- "If you don't know it estimate it, but be ready to revise your estimate"

```
procedure x = ASMQP ( Q , q , A , b , x , ε ) {
  for( B ← A(x) ; ; ) {
    solve (P_B) min{ ½ x^T Q x + q x : A_B x = b_B } for ( x̄ , μ̄_B );
    if( A_i x̄ ≤ b_i ∀ i ∉ B ) then {
      if( μ_B ≥ 0 ) then return;
      h ← min{ i ∈ B : μ_i < 0 }; B ← B \ { h }; continue;
      };
    d ← x̄ − x; ᾱ ← min{ α_i = ( b_i − A_i x )/A_i d : A_i d > 0 , i ∉ B };
    x ← x + ᾱ d; B ← A(x);
    };
  }
```

- B = "active set" (doh!), current estimate of $\mathcal{A}(x_*)$
- Handling not-full-rank A_B "hidden" in "solve (P_B)"

Exercise: the code has a glaring omission: which one? Fix it.

Active-set method

- $A\bar{x} \leq b \wedge \mu_B \geq 0 \Longrightarrow \bar{x}$ optimal (**why?**)
- $A\bar{x} \leq b \wedge \exists\, h \in B$ s.t. $\mu_h < 0 \wedge A_B$ full rank $\Longrightarrow v(P_{B\setminus\{h\}}) < v(P_B)$
- $A\bar{x} \not\leq b \Longrightarrow \bar{\alpha} < 1$ (**check**) $\Longrightarrow \mathcal{A}(x + \bar{\alpha}d) \supsetneq B$ (**why?**)
- $A\bar{x} \not\leq b \Longrightarrow \bar{\alpha} \in \operatorname{argmin}\{\, f(\, x + \alpha d\,) \;:\; \alpha \in [\,0\,,\, \bar{\alpha}\,]\,\}$ (**why?**)
- Under mild conditions finitely terminates: once found the right B the problem is over . . . "just" have to search among 2^m possible ones
- Variant: $B \leftarrow B \cup$ any $\subseteq \mathcal{A}(x) \setminus B$ (B may change little), helps in exploiting results of previous iteration to speed-up this one
- Details depend on how KKT system is solved: update factorizations, use previous solution/direction to warm-start iterative approaches . . .
- Many different variants (direct/iterative, H/M): Federico's playground
- Fundamental for overall efficiency
- Can be extended to $f(x)$ generic; say, use quasi-Newton to solve (P_B)

Active set method: special forms of constraints

- Important: always exploit all the structure of your problem
- QP with box constraints: (P) $\min\{ \frac{1}{2} x^T Q x + q x \ : \ l \leq x \leq u \}$
- Active constraint $\equiv$ inactive variable (fixed), "$B \subseteq N = \{1, \ldots, n\}$"
- $B \equiv (L, U)$, $L \cap U = \emptyset$, $L \cup U \subset N$, $F = N \setminus (L \cup U) \Longrightarrow$
  $A_B x = b \equiv x = [x_L, x_F, x_U]$ with $x_L = l_L$, $x_U = l_U$, x_F "free"
- W.l.o.g. $l = 0$ (**why? how?**) $\Longrightarrow x = [0, x_F, u_U]$
- (P) $\min\{ \frac{1}{2} x_F^T Q_{FF} x_F + (q_F + u_U^T Q_{UF}) x_F \}$ $[+\frac{1}{2} x_U^T Q_{UU} x_U + q_U u_U]$
  (**check**) unconstrained and in a (possibly, much) smaller space
- Initial feasible x straightforward

Exercise: write the detailed pseudo-code of the active set method for box-constrained QP. Where is μ?

Exercise: extend the above to general box-constrained nonlinear problems ($f(x)$ generic)

Outline

Frank-Wolfe method

- Nonlinear problem with linear constraints: (P) $\min\{ f(x) : Ax \leq b \}$
- Linear equalities are easy, linear inequalities less so ...

  ... but still plenty of efficient available software for LP ("black box")
- Solve a NLP by a sequence of LPs

```
procedure x = FWM ( f , A , b , x , ε ) {
  while( ‖ ∇f(x) ‖ > ε ) do {
    x̄ ← argmin { ⟨ ∇f(x) , y ⟩ : Ay ≤ b }; d ← x̄ − x;
    α ← Line_Search( f , x , d , 1 ); x ← x + αd;
    };
  }
```

- $\langle \nabla f(x) , d \rangle \leq 0$: d (almost) a descent direction (**why?**)
- Other stopping criterion: $\langle \nabla f(x) , d \rangle \geq -\varepsilon$:

  $\langle \nabla f(x) , d \rangle = 0 \Longrightarrow x$ local optimum

Exercise: prove the latter by exhibiting μ (hint: write the dual of the LP)

- f convex $\Longrightarrow f(x) + \langle \nabla f(x) , d \rangle \leq v(P)$ (**why?**) $\Longrightarrow$

  $\langle \nabla f(x) , d \rangle = 0 \Longrightarrow x$ global optimum

Exercise: prove the latter in at least two different ways

Frank-Wolfe method (cont.d)

- As usual, needs a feasible x to start with
- Convergence easy enough, more or less run-of-mill descent algorithm
- Have to solve a LP at each iteration, can of course be rather costly
- Exploit structure to solve the LP more efficiently

Exercise: describe fast LP solutions for box constraints and disjoint simplices

- Even if LP reasonably cheap, convergence rather slow:
  trusting linear model very far from where ∇f is computed
- Solution seen already: stabilization
  - add (box) constraint $\| y - x \|_\infty \leq \tau$
  - add (separable) penalty $\mu \| y - x \|_2^2$ to objective function
- Subproblems does not get much worse (LP $\rightarrow$ QP, but separable)
  could even become easier

Exercise: describe fast LP/QP solutions for box constraints and disjoint simplices for either type of stabiliaztion

- Have to manage τ/μ somehow
- Generally worth it

Frank-Wolfe $\rightsquigarrow$ cutting-plane $\rightsquigarrow$ Bundle

- Want a better direction? Use a better model!
- You see, my son, in the convex case first-order information is not so crap: it is globally valid, not only locally
- What if I just collect a bunch of it and use it all?
- $\{ x^i \} \longrightarrow \mathscr{B} = \{ (x^i , f^i = f(x^i) , g^i \in \partial f(x^i)) \} \equiv$ bundle of first-order information
- $f_{\mathscr{B}}(x) = \max\{ f^i + g^i(x - x^i) : (x^i , f^i , g^i) \in \mathscr{B} \} \equiv$ cutting-plane model of f (first-plus-ε-order model)
- Solve $\bar{x} \leftarrow \operatorname{argmin} \{ f_{\mathscr{B}}(x) : Ax \leq b \}$: constrained version of the cutting-plane algorithm
- Converges with fixed stepsize $\alpha = 1$, or line search
- More difficult to exploit structure (but not impossible)
- Still non-stabilized, but easy to stabilize:

  $$\bar{x} \leftarrow \operatorname{argmin} \{ f_{\mathscr{B}}(y) + \mu \| y - x \|_2^2 : Ay \leq b \}:$$

  constrained version of Bundle method
- Clearly works for non-differentiable (convex) f

sounds strangely familiar ...

Frank-Wolfe++: Sequential Quadratic Programming

- Want a better direction? Use a better model!
- Of course, second-order model if you have one:

$$\bar{x} \leftarrow x + \text{argmin} \left\{ \tfrac{1}{2} d^T \nabla^2 f(x) d + \nabla f(x) d \ : \ A(x + d) \leq b \right\}$$

- Can approximate $\nabla^2 f(x)$ with quasi-Newton formulæ
- Fast convergence if done properly
- Solve a constrained QP at each iteration
- Many complicated details
- Not for the faint of heart, not for today

Outline

Dual methods

- So far, kept $Ax \leq b$ and gotten $\mu \geq 0$ in the end (primal)
- Can we do the reverse (dual)? Of course we can
- $\forall$ fixed $\lambda \geq 0$, $\psi(\lambda) = \min_x\{ \frac{1}{2}x^T Qx + qx + \lambda(b - Ax) \} \leq v(\mathrm{P})$
  ψ concave + $Q \succ 0 \Longrightarrow$ optimal solution $x(\lambda) = Q^{-1}(\lambda A - q)$ (**why?**)
  ψ differentiable (**why?**), $\nabla\psi(\lambda) = b - Ax(\lambda)$ (**why?**)
- Lagrangian dual (D) $\max\{ \psi(\lambda) : \lambda \geq 0 \} \equiv$ (P) (**why?**)
- Solve (D) by any method for C^1, but $\psi \notin C^2$ in general
- λ^* optimal for (D) $\Longrightarrow x(\lambda^*)$ optimal for (P) (**check**)
- Feasible solution only asymptotically, but
  valid lower bound on $v(\mathrm{P})$ at every iteration
- Can behave very differently, e.g., degeneracy not an issue
- Q singular $\Longrightarrow \psi(\lambda) = -\infty$ happens $\Longrightarrow$ constraints in (D)
- Extends to $f(x)$ strictly convex (must solve general nonlinear problem)
- $f(x)$ not convex serious issue, ψ has to be computed exactly

Dual methods $\rightsquigarrow$ decomposition

- Partial dual: (P) $\min\{ f(x) : Ax \leq b , Ex \leq d \}$

  $\psi(\lambda) = \min_x\{ f(x) + \lambda(b - Ax) : Ex \leq d \}$

- Complicating constraints $Ax \leq b$ relaxed, easy constraints $Ex \leq d$ kept
- Subproblem constrained, but can exploit special structure

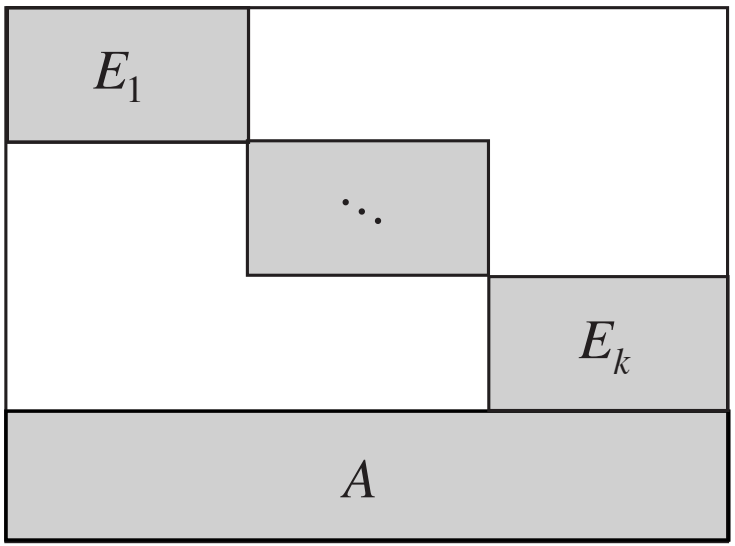
  (and less issues with $\psi(\lambda) = -\infty$)

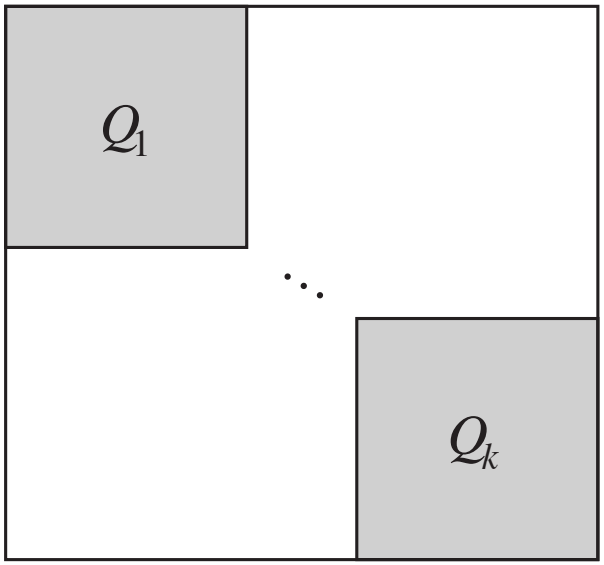
Dual methods $\rightsquigarrow$ decomposition

- Partial dual: (P) $\min\{ f(x) : Ax \leq b\,,\ Ex \leq d \}$

  $\psi(\lambda) = \min_x\{ f(x) + \lambda(b - Ax) : Ex \leq d \}$
- Complicating constraints $Ax \leq b$ relaxed, easy constraints $Ex \leq d$ kept
- Subproblem constrained, but can exploit special structure

  (and less issues with $\psi(\lambda) = -\infty$)
- Typical special structure: $Ax \leq b$ linking constraints





- $\psi(\lambda) = \sum_{k\in K} \psi^k(\lambda)$, each a smaller $\equiv$ simpler problem
- Algorithms can exploit sum-structure of ψ
- Not for the faint of heart, not for today

Outline

Barrier methods

- Pros of dual methods: (D) ($\approx$) unconstrained (would be with $Ax = b$)
- Cons of dual methods: $\psi \notin C^2$ (not even $\in C^1$ if f not strictly convex), $x(\lambda)$ never feasible
- Would like: unconstrained, $\in C^2$, x feasible
- First and last obvious: $f + \imath_X$, but $i_X \notin C^1$
- Can get C^2 if you accept to solve almost (P), but not quite: $\mu > 0$

  (P_μ) $\min\{ f_\mu(x) = f(x) - \mu \sum_{i=1}^m \log(\, b_i - A_i x \,) \}$
- f_μ strictly convex (if f convex), $f_\mu \in C^2$ (if $f \in C^2$)
- $f_\mu(x) < \infty \equiv x \in \mathit{int}\, X$, $f_\mu(x) \to \infty$ as $x \to \partial X$ (barrier function)
- $\exists!\, x_\mu$ optimal of (P_μ) (**why?**): $\mathcal{C} = \{\, x_\mu \,:\, \mu \in (0, \infty) \,\}$ central path
- $x_\infty = \lim_{\mu \to \infty} x_\mu$ analytic center of X (maximize product of slacks)
- $x_0 = \lim_{\mu \to 0} x_\mu$ optimal solution to (P) (analytic center of optimal face)
- Idea: start ($\approx$) at center x_∞, and ($\approx$) follow $\mathcal{C}$ to reach x_0
- Always (strictly) fasible, never touch ∂X

Barrier function & Newton's method

- Barrier function is self-concordant . . . very good for Newton's method: converge very quickly to x_μ if started within neighbourhood $\mathcal{N}$ of $\mathcal{C}$
- x^i "close" to $x(\mu^i)$: very few Newton's steps (typically one) give x^{i+1} "much closer" to $x(\mu^i) \Longrightarrow$
  x^{i+1} "close" to $x(\mu^{i+1})$ with $\mu^{i+1} \ll \mu^i$ $(\mu^{i+1} = \tau\mu^i,\ \tau < 1)$
- One Newton's step reduces μ by a constant factor $\Longrightarrow$
  exponentially fast convergence (and τ is "good" $\ll 1$)
- Path following: $\{ x^i \}$ "close" to $\{ x(\mu^i) \}$ for $\mu^i \searrow 0$ exponentially
- Appropriate choices $\Longrightarrow$ polynomial-time algorithm for specific problems (LP, convex QP, SOCP, SDP)
- Best choices in theory not best in practice (worst-case $\neq$ average case)
  "short step" better in theory, "long step" way better in practice
- Worst-case bound on iterations $\approx O(\log n \log(1/\varepsilon))$, in practice
  ≈ 100 iterations solve any problem as $n \nearrow \infty$
- Completely insensitive to degeneration (doh!)
- Long story short: very good convergence, but Newton's steps costly

Computing Newton's step

- Focus on quadratic case (P) $\quad \min\{\, \frac{1}{2}x^T Qx + qx \;:\; Ax \le b \,\}$
- Could compute Newton's step as usual

Exercise: compute $\nabla f_\mu(x)$ and $\nabla^2 f_\mu(x)$

- Cleaner derivation out of KKT conditions of (original) problem:
  $Qx + \lambda A = -q,\ Ax + s = b,\ \lambda \ge 0,\ s \ge 0,\ \lambda_i s_i = 0 \quad i = 1, \ldots, m$
- "Slackened KKT": $\lambda_i s_i = \mu \quad i = 1, \ldots, m$ characterize $x(\mu)$ (**check**)
- Useful notation: Λ, S diagonal matrices with λ_i, s_i on the diagonal:
  $\Lambda S = \mu u$ only nonlinear term in S-KKT, linearize it by Newton's method
- $x \to x + \Delta x$, $s \to s + \Delta s$, $\lambda \to \lambda + \Delta\lambda$ (current iterate + displacement):
  $(\lambda_i + \Delta\lambda_i)(s_i + \Delta s_i) = s_i\Delta\lambda_i + \lambda_i\Delta s_i + \lambda_i s_i + \Delta\lambda_i\Delta s_i$

$$\begin{bmatrix} Q & A^T & 0 \\ A & 0 & I \\ 0 & S & \Lambda \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -(Qx+q) - \lambda A \\ b - Ax - s \\ \mu u - \Lambda S u - \Delta\Lambda\Delta S u \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ \mu u - \Lambda S u \end{bmatrix} \quad (*)$$

assuming primal and dual feasibility $(Qx + \lambda A = -q,\ Ax + s = b)$

Primal-Dual Interior-Point (Barrier) Method

- Pick (x, λ, s) feasible, $\lambda > 0$ and $s > 0$, s.t. $\Lambda S = \mu u$ for some "large" μ
- Solve $(*)$ for $(\,\Delta x\,,\,\Delta s\,,\,\Delta\lambda\,)$
- Compute step α, $x \leftarrow x + \alpha\Delta x$, $s \leftarrow s + \alpha\Delta s$, $\lambda \leftarrow \lambda + \alpha\Delta\lambda$
- reduce μ (say, $\mu \leftarrow \tau\mu$ for $\tau < 1$), iterate
- New iterate primal and dual feasible if old one was (**check**):
  (D) $\max\{\, -\lambda b - \frac{1}{2}x^T Qx \;:\; Qx + \lambda A = -q\;,\; \lambda \geq 0\,\}$ (**check**) $\Longrightarrow$
  $-\lambda b - \frac{1}{2}x^T Qx \leq v(\mathrm{D}) \leq v(\mathrm{P}) \leq \frac{1}{2}x^T Qx + qx$ (note: $x \neq$ in (P), (D))
  (note: this requires same step in primal and dual space)
- Complementarity gap $= (\,\frac{1}{2}x^T Qx + qx\,) - (\,-\lambda b - \frac{1}{2}x^T Qx\,) = \lambda s$
- Both upper bound and lower bound on $v(\mathrm{P})$, get near as $\lambda s \searrow 0$
- $\lambda s = \Lambda S u = \mu m$ on $x(\mu) \equiv$ exact solutions of S-KKT (expected)
- Reduce complementarity gap for fixed μ, then reduce μ

Interior-Point Methods: Solving $(*)$

- Solving $(*)$ by far the most costly step, exploit structure
- Last group easy: $\Delta s = \Lambda^{-1}(\,\mu u - S\Delta\lambda\,) - s \Longrightarrow$
  $A\Delta x + \Delta s = 0 \Longrightarrow A\Delta x - [\,\Lambda^{-1}S\,]\Delta\lambda = s - \mu\Lambda^{-1}u \Longrightarrow$
  modified normal equations (note: $\Lambda^{-1}S \succ 0$ diagonal)

$$\begin{bmatrix} Q & A^T \\ A & -\Lambda^{-1}S \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} 0 \\ s - \mu\Lambda^{-1}u \end{bmatrix} \quad (**)$$

- Use second group: $A\Delta x - \Lambda^{-1}S\Delta\lambda = s - \mu\Lambda^{-1}u \Longrightarrow$
  $\Delta\lambda = \mu S^{-1}u + \Lambda S^{-1}A\Delta x - \lambda$ (**check**) $\Longrightarrow$
  $[\,Q + A^T\Lambda S^{-1}A\,]\Delta x = A^T(\,\lambda - \mu S^{-1}u\,)$ (**check**) reduced KKT
  $M = Q + A^T\Lambda S^{-1}A \succ 0$ if A has full column rank (**check**) (it should)
- Cholesky factorization of M (can be dense, permute rows of A . . .)
- Structure-exploiting Krylov-like methods for $(**)$
- Federico's playground
- Predictor-corrector variant: solve, add fixed term $\Delta\Lambda\Delta Su$ in r.h.s. of $(*)$, solve again (possibly any number of times)
- Sensible if can re-use information (factorization, . . .) from first solve

Interior-Point Methods: initial point, stepsize, reducing μ

- Initial (x, λ, s) is no problem: do not assume feasibility, use residuals $r^D = -(Qx + q) - \lambda A$, $r^P = b - Ax - s$ in $(*) \rightsquigarrow$ feasibility (quickly)
- Stepsize: maximum α stepsize s.t. both $\lambda + \Delta\lambda \geq 0$, $s + \Delta s \geq 0$ multiplied by $\bar{\alpha} < 1$ to keep interior ($\bar{\alpha} = .995, .9995$)
- In the unfeasible case may select $\alpha^P \neq \alpha^D$ that minimize λs

Exercise: develop formula for optimal α^P, α^D

- Choosing μ: $\mu = \rho(\lambda s)/m$, $\rho < 1$ fixed
- Reasonable automatic value: $\rho = 1/m$
- More sophisticated formulæ using α^P/α^D and for predictor-corrector
- Very good convergence, but large time/memory cost per iteration
- Special formulæ for $Dx = d$, box constraints $0 \leq x \leq u$, blocks ...

Exercise: develop (P) $\min\{ \frac{1}{2} x^T Q x + qx \ : \ Ax = b \ , \ 0 \leq x \leq u \}$

- May have numerical problems (dividing by very small numbers) especially on empty/unbounded problems

Outline

Wrap up

- Constraints add a world of complication to optimization
- Many different cases, "structure constraints" $\times$ "structure objective"
  $\Longrightarrow$ very many different ways to exploit them
- We barely scratched the surface, there is lots more:
  - other barrier methods
  - penalty methods
  - algorithms for highly nonlinear constraints
  - and more, and more, . . .
- Not to mention getting global optima in the nonconvex case
- ML usually does not need all this, but other applications do
- ML requires large size and speed: something's gotta give
- Still plenty of ways to do nice things

Wrap up of Wrap ups: technical

- Models are important for algorithms, too (besides vice-versa)
- Models must be simple, but first- and second-order ones are!
- Want a better direction? Use a better model!
  If the world does not give you one, invent one yourself!
- Thank goodness you can go (much) faster than gradient,
  but there is only so much you can do with first-order methods
- Always keep it convex if possible, better if C^1, better still if C^2
- Duality an extremely useful tool, especially (but not only) in convex case
- Mind trade-offs: "fat" models $\rightsquigarrow$ fast convergence but high iteration cost
- If you don't know it estimate it, but be ready to revise your estimate
- Best choices in theory not best in practice (worst-case $\neq$ average case)
- A lot of details need be considered, numerical aspects nontrivial

Wrap up of Wrap ups of Wrap ups: philosophical

- Dabble with math-based algorithms? Have to know (some) maths (doh!)
- Learn simple things first: must know a line search to optimize in $\mathbb{R}^n$
- Algorithms can only get so far with nasty problems
  hence choose your problems (foes) wisely; ML most often does
- Always exploit all the structure of your problem
- There is no one-size-fits-all solution
- Linear algebra is crucial for doing optimization, vice-versa also quite true
- Your mileage may vary, so try, try, try!

Wrap up of Wrap ups of Wrap ups: philosophical

- Dabble with math-based algorithms? Have to know (some) maths (doh!)
- Learn simple things first: must know a line search to optimize in $\mathbb{R}^n$
- Algorithms can only get so far with nasty problems
  hence choose your problems (foes) wisely; ML most often does
- Always exploit all the structure of your problem
- There is no one-size-fits-all solution
- Linear algebra is crucial for doing optimization, vice-versa also quite true
- Your mileage may vary, so try, try, try!

Lots of Fun!