Week 7-8

Nicolo Michelusi

I. Optimization algorithms

- Sometimes we can explicitly solve for the optimal solution in closed form, by solving KKT conditions directly, or solving the Lagrangian and dual problems (see previous examples)
- However, often a closed-form solution is not possible, and we need to resort to numerical algorithms
  - A numerical algorithm starts from some initial estimate x_0, and iteratively generate new estimates by
  $$x_{k+1} = T(x_k)$$
  Hopefully, as $k \to \infty$, $x_k \to x^*$, the optimal solution
  - When does such a sequence converges to the optimal solution?
  - If so, how long does it take to converge to a certain accuracy? (sample complexity)
- Example: compute $\sqrt{2}$ using only $+, -, \times, /$.
  $$x = \sqrt{2} \Leftrightarrow (x-1)(x+1) = 1 \Leftrightarrow x = \frac{1}{x+1} + 1$$
  This suggests the update
  $$T(x_k) = \frac{1}{x_k+1} + 1$$
  which is such that $T(\sqrt{2}) = \sqrt{2}$ (i.e., $\sqrt{2}$ is a fixed point of $x = T(x)$)
  To prove convergence, let $x, y \geq 1$, and consider $|T(x) - T(y)|$:
  $$|T(x) - T(y)| = \frac{|y-x|}{(x+1)(y+1)} \leq \frac{1}{4}|y-x|$$
  Therefore, choosing $y = \sqrt{2}$ we get
  $$|x_{k+1} - \sqrt{2}| = |T(x_k) - \sqrt{2}| \leq \frac{1}{4}|x_k - \sqrt{2}| \leq \cdots \leq \frac{1}{4^{k+1}}|x_0 - \sqrt{2}|$$
  and therefore x_k converges linearly to $\sqrt{2}$, by initializing it with $x_0 \geq 1$.
  However, not all algorithms converge:
  $$x = \sqrt{2} \Leftrightarrow (x-1)(x+1) = 1 \Leftrightarrow x = \frac{1}{x-1} - 1$$
  but the algorithm $x_{k+1} = \frac{1}{x_k-1} - 1$ does not converge

II. Algorithms for Unconstrained optimization

- Solve $\min f(x)$, f convex

  Optimality condition is

$$f'(x^*; x - x^*) \geq 0, \ \forall x$$

  When f is differentiable, the optimality condition becomes

$$\nabla f(x^*) = 0$$

- Assume f differentiable; consider the iteration of the type

$$x_{k+1} = T(x_k) = x_k - \alpha \nabla f(x_k)$$

  Note that x^* is a fixed point of the mapping $T(x)$: if $x_k = x^*$, then $T(x_k) = x^*$.
- Example: $f(x) = \frac{1}{2}x^2$

- Note: the algorithm does not converge when α is too large; it converges slowly if α is too small..
- Proof of convergence (for $\alpha > 0$ sufficiently small). Need to show that

1) $f(x_k)$ decreases across iterations
2) $\|x_k - x^*\|_2$ decreases sufficiently fast across iterations

  Typically, we need stronger structural properties of the function, in addition to convexity

- First approach

Lemma 1. *Assume f is continuously differentiable and $\exists L > 0$ such that*

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2,\ \forall x, y, \in \mathbb{R}^n$$

(gradient is Lipschitz continuous with parameter L) Then,

$$f(y) \leq f(x) + \nabla f(x)^T (y - x) + \frac{L}{2}\|x - y\|_2^2,\ \forall x, y, \in \mathbb{R}^n$$

-

Theorem 2. *Assume the same conditions as before hold; f is bounded below by f^*; and $0 < \alpha < 2/L$. Then $\nabla f(x_k) \to 0$ for $k \to \infty$.*

- Norm approach:

  Lemma 3. *If f is convex, then*

  $$(\nabla f(x) - \nabla f(y))^T (x - y) \geq 0,\ \forall x, y, \in \mathbb{R}^n$$

  (this holds also if ∇ is a sub-gradient)

  A mapping that satisfies this condition is called "monotone mapping"

•

Lemma 4. *If f is convex, differentiable, and its gradient is Lipschitz continuous with parameter L, i.e.*

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2, \ \forall x, y, \in \mathbb{R}^n$$

then

$$(\nabla f(x) - \nabla f(y))^T (x - y) \geq \frac{1}{L}\|\nabla f(x) - \nabla f(y)\|_2^2, \ \forall x, y, \in \mathbb{R}^n$$

•

Theorem 5. *Assume that*

$$(\nabla f(x) - \nabla f(y))^T (x - y) \geq \frac{1}{L} \|\nabla f(x) - \nabla f(y)\|_2^2, \ \forall x, y, \in \mathbb{R}^n;$$

$0 < \alpha < 2/L$ *and* $\exists x^*$ *with* $\nabla f(x^*) = 0$. *Then, the sequence of points generated by*

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$

converges, and the limit x_∞ *satisfies* $\nabla f(x_\infty) = 0$.

- These results prove convergence to (one) optimal point x^*. However, they do not provide guarantees on how much time it takes to converge. To this end, we need stronger conditions (e.g., strong convexity)

  Theorem 6. *If f is strongly convex with Lipschitz continuous gradient with parameter L,*

  $$L\|x-y\|_2^2 \geq [\nabla f(x) - \nabla f(y)]^T (x-y) \geq \rho\|x-y\|_2^2, \ \forall x,y \in \mathbb{R}^n$$

  for some $\rho > 0$ (note that we must have $\rho \leq L$), and $0 < \alpha < \frac{2\rho}{L^2}$, then x_k converges to x^* with linear rate. In particular,

  $$\|x_k - x^*\| \leq \xi^k \|x_0 - x^*\|$$

  where $\xi = \sqrt{1 + \alpha^2 L^2 - 2\alpha\rho} \in (0,1)$.

- Scaled gradient descent algorithm:

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$

converges if the gradient of f is Lipschitz continuous with parameter L and $\alpha < 2/L$.

The algorithm can be made faster by properly scaling the gradient by a positive definite matrix $P \succ 0$:

$$x_{k+1} = x_k - \alpha P \nabla f(x_k)$$

This algorithm converges if the gradient of f is Lipschitz continuous and $\alpha < \frac{2}{L\lambda_{\max}}$, where $\lambda_{\max}$ is the maximum eigenvalue of f.

To see this, note that the this is equivalent to a change of variables:

- Example

$$\min_{x_1,x_2} \frac{1}{2}(x_1^2 + \rho x_2^2)$$

  where $\rho \gg 1$

- These algorithms can be generalized as follows:

$$x_{k+1} = x_k + d_k$$

where d_k is a descent direction:

$$\nabla f(x_k)^T d_k \leq -\epsilon \|\nabla f(x_k)\|_2^2, \ \epsilon > 0$$

Further, assume that

$$\|d_k\|_2 \leq M \|\nabla f(x_k)\|_2$$

Then, we can prove the following:

Theorem 7. *Assume d_k is a descent direction and $\epsilon > \frac{LM^2}{2}$. Assume f is bounded below. Then, if $\epsilon > \frac{LM^2}{2}$, $\nabla f(x_k) \to 0$ for $k \to \infty$.*

To see this,

$$f(x_{k+1}) = f(x_k + d_k) \leq f(x_k) + \nabla f(x_k)^T d_k + \frac{L}{2}\|d_k\|_2^2 \leq f(x_k) - \left(\epsilon - \frac{LM^2}{2}\right) \|\nabla f(x_k)\|_2^2$$

hence

$$f^* \leq f(x_{n+1}) \leq f(x_0) - \left(\epsilon - \frac{LM^2}{2}\right) \sum_{k=0}^{n} \|\nabla f(x_k)\|_2^2$$

hence we must have $\|\nabla f(x_k)\|_2 \to 0$ for $k \to \infty$.

Examples of descent directions:

  - $d_k = -\alpha \nabla f(x_k)$ (standard gradient descent algorithm)

  - $d_k = -\alpha P \nabla f(x_k), \ P \succ 0$ (scaled gradient descent algorithm)

- Assume a strongly convex function $H(x) \succ \rho I,\ \forall x$, such that $H(x) \prec \lambda_{\max} I$. $d_k = -\alpha H(x)^{-1} \nabla f(x_k)$ (Newton algorithm)

Note: Newton direction is the one that minimizes a second order Taylor approximation of the objective function

$$f(y) \simeq f(x_k) + \nabla f(x_k)^T (y - x_k) + \frac{1}{2}(y - x_k)^T H(x_k)(y - x_k)$$

minimized at

$$y^* - x_k = -H(x_k)^{-1} \nabla f(x_k)$$

- These proofs require the function to be smooth (Lipschitz continuous gradient)

What if this condition is not satisfied? We need to use sub-gradients. In this case, the standard gradient descent does not converge to the optimal point, but may keep oscillating:

Theorem 8. *Assume f is convex and its subgradients are bounded, $\|\nabla f(x)\|_2 \leq M$. Consider the subgradient descent algorithm*

$$x_{k+1} = x_k - \alpha \nabla f(x_k),$$

where $\nabla f(x)$ is a subgradient of f at x. Then, for any $\epsilon > 0$ and $\alpha < \epsilon / M^2$, $\forall k \geq 0$ there exists $n \geq k$ such that

$$f(x_n) < f(x^*) + \epsilon,$$

(i.e. x_n is an $\epsilon-$suboptimal point)

To guarantee convergence to the optimal point, we need to use a diminishing step-size.

Theorem 9. *Assume f is convex and its subgradients are bounded, $\|\nabla f(x)\|_2 \leq M$. Consider the subgradient descent algorithm*

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k),$$

where $\nabla f(x)$ is a subgradient of f at x. Then, if

$$\sum_k \alpha_k = \infty, \ \sum_k \alpha_k^2 < \infty,$$

then $x_k \to x^*$, where x^* has a sub-gradient $\nabla f(x^*) = 0$

III. CONSTRAINED OPTIMIZATION ALGORITHMS

- Solve $\min f(x),$ s.t. $x \in \mathcal{F}$, f convex, $\mathcal{F}$ is convex

  Optimality condition is

$$f'(x^*; x - x^*) \geq 0, \ \forall x \in \mathcal{F}$$

  where $x^* \in \mathcal{F}$

  When f is differentiable, the optimality condition becomes

$$\nabla f(x^*)^T (x - x^*) = 0, \ \forall x \in \mathcal{F}$$

- However, the normal gradient descent algorithm does not work any more because the new x_{k+1} might fall outside of $\mathcal{F}$
- Three solutions to this problem:

1) Associate a penalty to constraint violation: choose convex $g(x)$ such that

$$g(x) = 0, \ x \in \mathcal{F}$$

$$g(x) > 0, \ x \notin \mathcal{F}$$

   and solve the unconstrained problem

$$\min f(x) + \beta g(x)$$

   The solution will approach the original constrained problem as $\beta \to \infty$

2) Interior point method: choose $g(x)$ such that $g(x) \to \infty$ as x approaches the boundary of $\mathcal{F}$ from inside; then, minimize

$$\min f(x) + \beta g(x)$$

   as before; due to the barrier, the optimal solution is in the interior of $\mathcal{F}$; as $\beta \to 0$, the optimal solution tends to the solution of the unconstrained problem

3) Projection method: after each update, project x_{k+1} back to its feasible set:

$$[x_{k+1}]^+ = \arg\min_{x \in \mathcal{F}} \|x - x_{k+1}\|_2$$

In the first two cases, the problem is converted to an unconstrained problem; we can then use gradient based algorithms; however, it may be difficult to ensure the Lipschitz continuity of the gradient.

IV. Projection and gradient projection algorithm

- Define the projection

$$[x]^+ = \arg\min_{y \in \mathcal{F}} \|y - x\|_2$$

Example: $\mathcal{F} \equiv \otimes_i [a_i, b_i]$ (projection onto a box)

- Projection theorem (Bertsekas&Tsitsiklis,P.211)

  Theorem 10.

  1) *$\forall x$, $\exists$ a unique $z \in \mathcal{F}$ that minimizes $\|y - x\|_2$ over all $y \in \mathcal{F}$; hence, $[x]^+$ is uniquely defined.*
  2) *$z = [x]^+$ if and only if $(y - z)^T(x - z) \leq 0, \ \forall y \in \mathcal{F}$*
  3) *The mapping $p(x) = [x]^+$ is continuous and non-expansive, i.e.*

$$\|p(x) - p(y)\|_2 \leq \|x - y\|_2, \ \forall x, y \in \mathbb{R}^n$$

- Gradient projection algorithm

$$x_{k+1} = [x_k - \alpha \nabla f(x_k)]^+$$

Lemma 11. *Assume f is convex and differentiable. Then $x^* = \arg\min_{x \in \mathcal{F}} f(x)$ if and only if*

$$x^* = [x^* - \alpha \nabla f(x^*)]^+,$$

i.e. x^* is a fixed point of the gradient projection algorithm.

Theorem 12. *If f is convex, with Lipschitz continuous gradient with parameter L, there exists some x^* such that $x^* = [x^* - \alpha \nabla f(x^*)]^+$, and $0 < \alpha < 2/L$, then x_k converges and its limit minimizes $f(x)$ over $\mathcal{F}$.*

If further f is strongly convex, we have the following linear convergence result

Theorem 13. *If f is strongly convex with Lipschitz continuous gradient with parameter L,*

$$L\|x-y\|_2^2 \geq [\nabla f(x) - \nabla f(y)]^T(x-y) \geq \rho\|x-y\|_2^2,\ \forall x, y \in \mathbb{R}^n$$

for some $\rho > 0$ (note that we must have $\rho \leq L$), $x^* = \arg\min_{x\in\mathcal{F}} f(x)$ (unique since f is strongly convex), and $0 < \alpha < \frac{2\rho}{L^2}$, then x_k converges to x^* with linear rate. In particular,

$$\|x_k - x^*\| \leq \xi^k \|x_0 - x^*\|$$

where $\xi = \sqrt{1 + \alpha^2 L^2 - 2\alpha\rho} \in (0, 1)$.

- Scaled gradient projection algorithm: similar to the unconstrained case, we can define the scaled version of the algorithm

$$x_{k+1} = [x_k - \alpha P \nabla f(x_k)]^+$$

  However, in this case, we need to take special case at the projection operation. To see this, treat the scaled algorithm as a change of variables:

- Projection in the dual
  - In general, the projection operation can be difficult to carry out if the constraints set is in a complex form.
  - However, projection is easy in the dual domain, since the constraint set is always a quadrant. In addition, the subgradient has a simple form.
- Primal problem:

$$\min f_0(x)$$
$$\text{s.t. } f_i(x) \le 0,\ \forall i$$
$$Ax = b$$

Lagrangian:

$$L(x, \lambda, \nu) = f_0(x) + \sum_i \lambda_i f_i(x) + \nu^T(Ax - b),\ \lambda \ge 0$$

Dual function

$$g(\lambda, \nu) = \min_x L(x, \lambda, \nu)$$

the minimization of the Lagrangian is unconstrained, hence it can be accomplished using a standard unconstrained gradient descent algorithm.

Dual problem

$$\max g(\lambda, \nu)$$
$$\text{s.t. } \lambda \ge 0$$

This can be solved using the gradient projection algorithm.

The subgradient of g at $(\lambda^{(k)}, \nu^{(k)})$ is given by

$$\nabla g(\lambda^{(k)}, \nu^{(k)}) = [f_1(x^{(k)}), \ldots, f_m(x^{(k)}), Ax - b]$$

where

$$x^{(k)} = \arg\min_x L(x, \lambda^{(k)}, \nu^(k))$$

To show that this is indeed a subgradient, need to show that

$$g(\lambda, \nu) \le g(\lambda^{(k)}, \nu^{(k)}) + \nabla g(\lambda^{(k)}, \nu^{(k)})^T([\lambda; \nu] - [\lambda^{(k)}; \nu^{(k)}]),\ \forall \lambda \ge 0, \forall \nu$$

(note that g is concave)

In fact we have

$$g(\lambda^{(k)},\nu^{(k)})+\nabla g(\lambda^{(k)},\nu^{(k)})^T([\lambda;\nu]-[\lambda^{(k)};\nu^{(k)}])$$

$$=L(x^{(k)},\lambda^{(k)},\nu^{(k)})+\sum_i f_i(x^{(k)})(\lambda_i-\lambda_i^{(k)})+(\nu-\nu^{(k)})^T(Ax^{(k)}-b)$$

$$=L(x^{(k)},\lambda,\nu)\geq \min_x L(x,\lambda,\nu)=g(\lambda,\nu).$$

As a result, the gradient projection algorithm for the dual is of the following simple form:

$$\lambda_i^{(k+1)}=[\lambda_i^{(k)}+\alpha_k f_i(x^{(k)})]^+$$

$$\nu^{(k+1)}=\nu^{(k)}+\alpha_k(Ax^{(k)}-b)$$

(possibly, diminishing step-size if not differentiable)

- Example: waterfilling in fading channels

$$\max_p \sum_g \mathbb{P}(g) \ln(1 + gp(g))$$

$$\text{s.t. } 0 \leq p \leq P_{\max}$$

$$\sum_g \mathbb{P}(g)p(g) \leq \bar{P}$$

- Example: utility maximization of a single resource

$$\max_x \sum_i U_i(x_i)$$

$$\text{s.t. } x \geq 0$$

$$\sum_i x_i \leq R$$

- Example: distributed optimization over a network

$$\min_x \sum_i f_i(x)$$