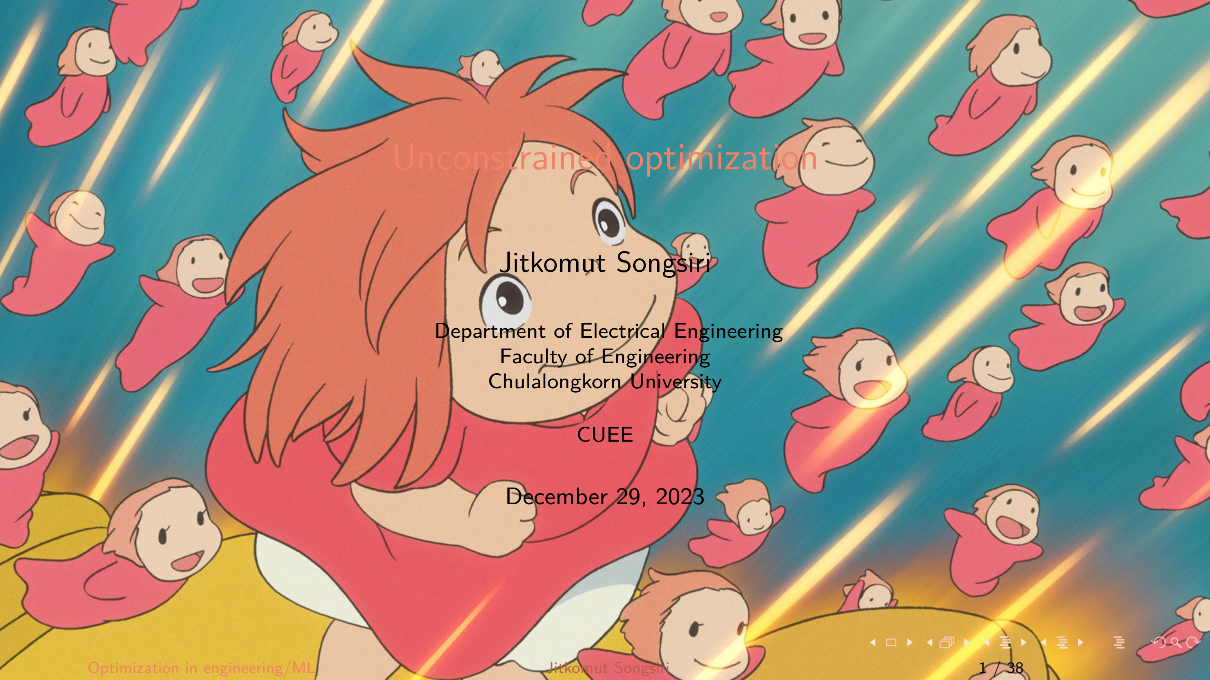


Unconstrained optimization

Jitkomut Songsiri

Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University

CUEE

December 29, 2023

Outline

Problem setting

problem: minimize $f(x)$ over all $x \in \mathbf{R}^n$

applicable numerical methods depend on the property of f

- smooth objective function (continuously differentiable)
- non-smooth objective function
- gradient-based methods used in ML: concern issues about flat regions or differential curvatures of f
- mini-batch optimization: f is a sum of functions of the same form

Function properties and general setting

Taylor's theorem:

assume that f has $n+1$ continuous derivatives

$f(z)$ can be expressed by Taylor series about x

$$f(z) = f(x) + f'(x)(z-x) + \frac{f''(x)(z-x)^2}{2!} + \cdots + \frac{f^{(n)}(x)(z-x)^n}{n!} + \underbrace{\frac{f^{(n+1)}(\xi)(z-x)^{n+1}}{(n+1)!}}_{E_n(z)}$$

where $E_n(z)$ called the remainder (hold for some ξ between z and x)

multivariate case: $f :\in \mathbf{R}^n \to \mathbf{R}$

$$\begin{array}{lll} \text{1st-order:} & f(x+\Delta x) & = f(x) + \nabla f(x)^T \Delta x + (1/2)\Delta x^T \nabla^2 f(\xi) \Delta x \\ \text{2nd-order:} & f(x+\Delta x) & = f(x) + \nabla f(x)^T \Delta x + (1/2)\Delta x^T \nabla^2 f(x) \Delta x + \text{remainder} \end{array}$$

Taylor approximation is the expression without the remainder term

Smooth objective

assumption: f is twice continuously differentiable

- **first-order necessary condition:**

$$\text{if } x^\star \text{ is a local minimizer of } f \text{ then } \nabla f(x^\star) = 0$$

- **second-order sufficient condition:** if $\nabla f(x^\star) = 0$ and $\nabla^2 f(x^\star) \succ 0$

$$\text{then } x^\star \text{ is a strict local minimizer of } f$$

local minimizers can be distinguished from other stationary points by examining positive definiteness of $\nabla^2 f$

Example

$f(x) = \frac{1}{3}x_1^3 + \frac{1}{2}x_1^2 + 2x_1x_2 + \frac{1}{2}x_2^2 - x_2 + 9$

the necessary condition for stationary points is

$$\nabla f(x) = \begin{bmatrix} x_1^2 + x_1 + 2x_2 \\ 2x_1 + x_2 - 1 \end{bmatrix} = 0 \quad \Rightarrow \quad u = (1, -1) \text{ or } v = (2, -3)$$

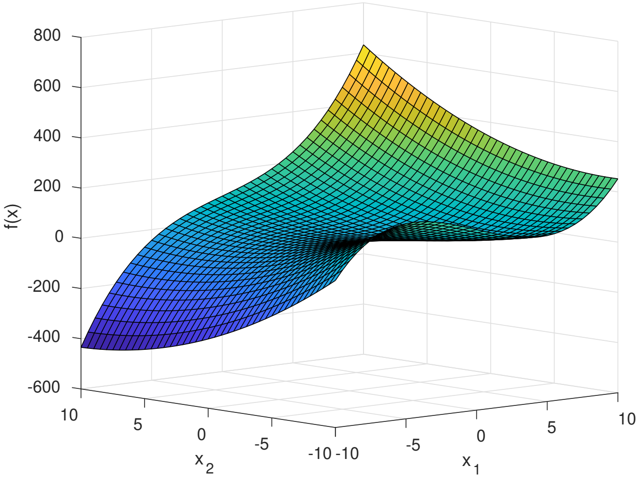
the Hessian matrix of f is

$$\nabla^2 f(x) = \begin{bmatrix} 2x_1 + 1 & 2 \\ 2 & 1 \end{bmatrix}, \quad \nabla^2 f(u) = \begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix}, \quad \nabla^2 f(v) = \begin{bmatrix} 5 & 2 \\ 2 & 1 \end{bmatrix}$$

- $\nabla^2 f(v) \succ 0$, so v is a local minimizer
- $\nabla^2 f(u)$ is indefinite, so u is neither a minimizer nore a maximizer of f
- f has neither a global minimizer nor a global maximizer since f is unbouded as $x_1 \to \infty$

Example of local minimum

a surface plot of $f(x) = \frac{1}{3}x_1^3 + \frac{1}{2}x_1^2 + 2x_1x_2 + \frac{1}{2}x_2^2 - x_2 + 9$



f is unbouded and has a local minimum

Properties of convex functions

definition: $f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$ for all x, y and $0 \leq \theta \leq 1$

first-order condition: f is convex if and only if $\mathbf{dom}\, f$ is convex and

$$f(y) \geq f(x) + \nabla f(x)^T(y - x), \quad \forall x, y \in \mathbf{dom}\, f$$

- RHS is the first-order Taylor approximation is a global underestimator of f
- if $\nabla f(x) = 0$ then for all $y \in \mathbf{dom}\, f$, we have $f(y) \geq f(x)$, that is x is a global minimizer of f

second-order condition: f is convex if and only if $\mathbf{dom}\, f$ is convex and

$$\nabla^2 f(x) \succeq 0, \quad \forall x \in \mathbf{dom}\, f$$

General optimization algorithm

algorithms for unconstrained optimizations have the same iterative form

$$x^{(k+1)} = x^{(k)} + t_k \Delta x^{(k)}$$

- each method differs by the search direction $\Delta x^{(k)}$
- the choices of step size t_k
  1. exact line search: optimal step size, *i.e.*,

     $$t_k = \underset{t \geq 0}{\operatorname{argmin}} \, f(x^{(k)} + t_k \Delta x^{(k)})$$

  2. a fixed nonnegative value
  3. a decaying sequence
  4. inexact line search: the objective value is improved in some sense

all choices of step size must yield the iteration convergence; more details in Chapter 3 of J. Nocedal textbook

Descent direction

initial sublevel set: $S_0 = \{\ x \in \mathbf{dom}\, f \mid f(x) \leq f(x^{(0)})\ \}$ and assume S_0 is closed

descent method: an iterative method has a descent property if

$$f(x^{(k+1)}) < f(x^{(k)}) \quad \text{(except when } x^{(k)} \text{ is optimal)}$$

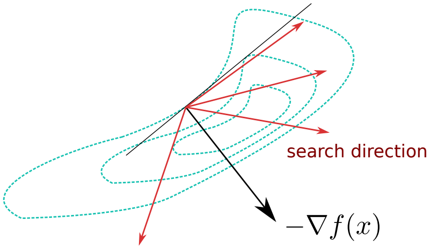
- it implies that for all k we have $x^{(k)} \in S_0$
- **descent direction:** **acute** angle between the search direction and $-\nabla f(x^{(k)})$

$$f(x^{(k)} + t\Delta x^{(k)}) = f(x^{(k)}) + t\nabla f(x^{(k)})^T \Delta x^{(k)} + \mathcal{O}(t^2)$$

if $\nabla f(x^{(k)})^T \Delta x^{(k)} < 0$ then for a sufficiently small t, we will have

$$f(x^{(k)} + t\Delta x^{(k)}) < f(x^{(k)})$$

Descent direction



when f is **convex** with the first-order condition: $f(y) \geq f(x) + \nabla f(x)^T(y-x)$

$$f(y) \geq f(x) + \nabla f(x)^T(y-x), \quad \forall x, y \in \mathbf{dom}\, f$$

the condition $\nabla f(x^{(k)})^T(y - x^{(k)}) \geq 0$ implies $f(y) \geq f(x^{(k)})$
for **convex** f, a search direction Δx is in a descent method must satisfy

$$\nabla f(x^{(k)})^T \Delta x^{(k)} < 0$$

Step length rules

denote $x^+ := x + t\Delta x$ (x is the current, x^+ is the next iteration)

traditionally, choices of step length (stepsize, learning rate) are

- exact line search: find t that minimizes $f(x + t\Delta x)$
- backtracking (or inexact) line search: choose $\beta, \alpha \in (0, 1)$, initialize t and check if

  $$f(x^+) \leq f(x) + \alpha t \nabla f(x)^T \Delta x$$

  otherwise, reduce the step length by $t := \beta t$ (this is called Armijo's condition)
- fixed step length: chosen to obtain a convergence
- diminishing step length: for example, $t_k = 1/k$ which satisfies the conditions

$$t_k \rightarrow 0, \quad \sum_{k=0}^{\infty} t_k = \infty, \quad \sum_{k=0}^{\infty} t_k^2 < \infty$$

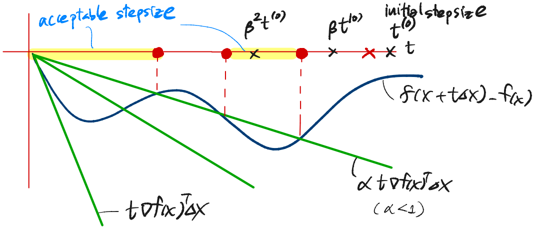
Backtracking line search

Armijo rule: stepsize selection rule that is based on successive reduction

1. choose parameters $0 < \alpha, \beta < 1$ and initialize a stepsize t
2. compute $x^+ = x + t\Delta x$ and evaluate $f(x + t\Delta x)$
3. if the condition

$$f(x + t\Delta x) \leq f(x) + \alpha t \nabla f(x)^T \Delta x$$

does not satisfy, then reduce t by computing $t := \beta t$ and repeat step 2)



Conventional methods

Methods for unconstrained problems

first-order methods: for continuously differentiable f

- steepest-descent method
- quasi-Newton methods
- trust-region method
- nonlinear conjugate gradient method

second-order method: Newton

first-order methods: for convex and Lipschitz continuously differentiable f

- FISTA
- Nesterov's second method

Outlines of conventional methods

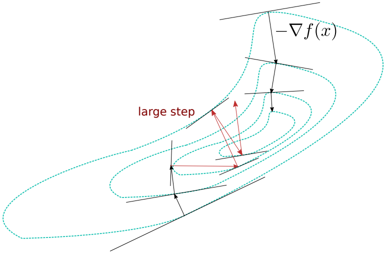
- steepest-descent
- Newton/quasi-Newton
- trust-region
- nonlinear conjugate gradient

Steepest-descent method

use the negative gradient direction

$$\Delta x^{(k)} = -\nabla f(x^{(k)})$$

and a line search to determine the step size $t^{(k)}$



- the search direction has a descent property if $\nabla f(x^{(k)}) \neq 0$
- minimizing the approximation $f(x^{(k)} + s) \approx f(x^{(k)}) + s^T \nabla f(x^{(k)})$ is done via

$$\underset{s \neq 0}{\text{minimize}} \quad \frac{s^T \nabla f(x^{(k)})}{\|s\| \cdot \|\nabla f(x^{(k)})\|}$$

which gives the solution: $s = -\nabla f(x^{(k)})$ as $\Delta x^{(k)}$

Newton method

the search direction satisfies

$$[\nabla^2 f(x^{(k)})]\Delta x^{(k)} = -\nabla f(x^{(k)})$$

- if $\nabla f^2(x^{(k)}) \succ 0$ then it is a descent direction
- the Newton direction, s, minimizes the quadratic approximation of f

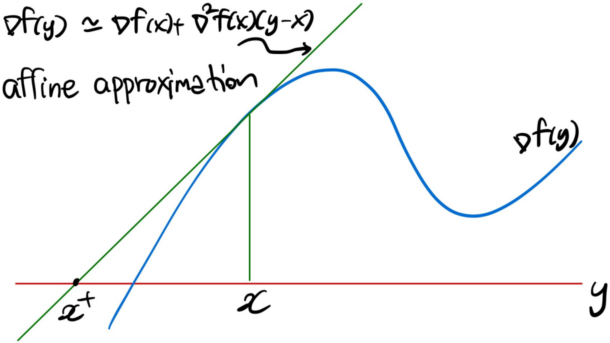
$$\nabla f(x^{(k)} + s) \approx f(x^{(k)}) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T \nabla^2 f(x^{(k)}) s$$

- classical Newton method use the step size of 1 and has a quadratic convergence
- the cost of solving the linear system for $\Delta x^{(k)}$ is $\mathcal{O}(n^3)$

Interpretation of Newton step

the linear approximation of $\nabla f(x^{(k)} + \Delta x^{(k)}) = 0$ gives the Newton direction s

$$0 = \nabla f(x^{(k)} + \Delta x^{(k)}) \approx \nabla f(x^{(k)}) + \nabla^2 f(x^{(k)})\Delta x^{(k)}$$



(in the figure, $x^+ \triangleq x^{(k+1)}$ and $x \triangleq x^{(k)}$)

Quasi-Newton method

approximate the Hessian at low cost, $H_k \approx \nabla f^2(x^{(k)})$ and $\Delta x^{(k)}$ satisfies

$$H_k \Delta x^{(k)} = -\nabla f(x^{(k)})$$

these methods can propogate H_k^{-1} to simplify the calculation of $\Delta x^{(k)}$

- **BFGS** (Broyden-Fletcher-Goldfarb-Shanno)

$$s = x^{(k)} - x^{(k-1)}, \quad y = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$$

$$H_k = H_{k-1} + \frac{yy^T}{y^T s} - \frac{H_{k-1} s s^T H_{k-1}}{s^T H_{k-1} s}$$

$$H_k^{-1} = \left(I - \frac{sy^T}{y^T s}\right) H_{k-1}^{-1} \left(I - \frac{ys^T}{y^T s}\right) + \frac{ss^T}{y^T s}$$

cost of the inverse update is $\mathcal{O}(n^2)$ as compared to $\mathcal{O}(n^3)$ for Newton

Quasi-Newton method

- **DFP** (Davidon-Fletcher-Powell): solution is dual of BFGS formula

$$H_k = \left(I - \frac{ys^T}{s^Ty}\right) H_{k-1}^{-1} \left(I - \frac{sy^T}{s^Ty}\right) + \frac{yy^T}{s^Ty}$$

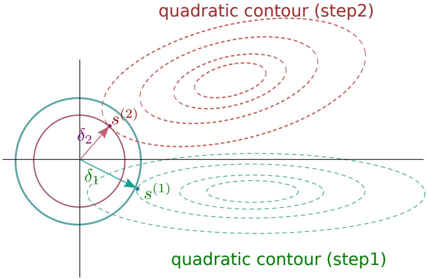
(interchange the roles of y and s in the expression of H_k^{-1} from BFGS)

Trust-region method

trust a quadratic approximation of $f(x^{(k)} + s)$ in region $\|s\| \leq \delta_k$

for each iteration, the method solves the subproblem for the search direction s

$$
\begin{array}{ll}
\text{minimize} & f(x^{(k)}) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T \nabla^2 f(x^{(k)}) s \\
\text{subject to} & \|s\| \leq \delta_k
\end{array}
$$



δ_k is updated by examining a reduction of f as compared to quadratic approximation

Trust-region method

the optimality conditions of the subproblem are

$$(\nabla^2 f(x^{(k)}) + \lambda I)s = -\nabla f(x^{(k)}), \quad \lambda(\delta_k - \|s\|) = 0$$

($\lambda \geq 0$ is the Lagrange multiplier) and the method guarantees that

$$\|s\| = \|(\nabla^2 f(x^{(k)}) + \lambda I)^{-1} \nabla f(x^{(k)})\| \leq \delta_k$$

- if δ_k is very large, $\lambda = 0$ then s approaches the Newton step
- if $\delta_k \to 0$ then λ must be large and dominate $\nabla^2 f$, which gives

$$s \approx -\frac{1}{\lambda} \nabla f(x^{(k)}) \quad \text{(closer to the gradient step)}$$

- the idea of solving the step: $(\nabla^2 f(x^{(k)}) + \lambda I)s = -\nabla f(x^{(k)})$ was first proposed by **Levenberg-Marquardt** (LM) for nonlinear least-squares problems where λ is called the *damping parameter*
- both LM and trust-region methods are also called *restricted Newton step methods*

Conjugate gradient method

- conjugate gradient (CG) method for linear equations
  - motivated from minimizing $(1/2)x^T Ax - b^T x$ or solving $Ax = b$
  - converges in at most n steps (can be less if A has less distinct eigenvalues)
- preconditioned CG: change of coordinates $x = By$ to make spectrum of $B^T AB$ more clustered
- nonlinear conjugate gradient method
  - extended to non-quadratic unconstrained problem
  - approximate a nonlinear f by a second-order Taylor series

$$f(x) \approx \tilde{f}(x) = (1/2)x^T \nabla^2 f(x) x + \nabla f(x)^T x + r$$

  - apply CG to $\tilde{f}$ while modifying the minimization of f along conjugate vectors
  - well-known modifications: Hestenes-Stiefel, Polak-Ribière, Fletcher-Reeves

CG method for linear equations

given a matrix A, a set of vectors $\{p_j\}$ are **conjugate** with A if

$$p_i^T A p_j = 0, \quad \text{if } i \neq j$$

- first assume that $\{p_i\}$ is known and $f(x) = (1/2)x^T Ax - b^T x$
- consider a trial point $z = \sum_{i=1}^m \alpha_i p_i$, it can be shown from conjugacy that

$$\underset{z}{\text{minimize}}\, f(z) \quad \Rightarrow \quad \alpha_i = \frac{b^T p_i}{p_i^T A p_i}$$

meaning if we can represent the solution as an LC of $\{p_i\}$, it can be found easily

Nonlinear least-squares

a specific type of unconstrained problem of the form

$$\text{minimize} \;\; f(x) := (1/2)[r_1(x)^2 + r_2(x)^2 + \cdots + r_q(x)^2]$$

- **Gauss-Newton method:** apply the Newton and neglect a term in $\nabla^2 f$

$$r(x) = (r_1(x), \ldots, r_q(x)), \quad \nabla f(x) = J(x)^T r(x), \quad J(x) \text{ is Jacobian of } r$$

$$\nabla^2 f(x) = J(x)^T J(x) + S(x) \approx J(x)^T J(x)$$

search direction: $$[J(x^{(k)})^T J(x^{(k)})] s^{(k)} = -J(x^{(k)})^T r(x^{(k)})$$

the method has a global convergence

- **Levenberg-Marquardt method:** replace the search direction equation with

$$[J(x^{(k)})^T J(x^{(k)}) + \lambda^{(k)} I] s^{(k)} = -J(x^{(k)})^T r(x^{(k)})$$

$\lambda^{(k)}$ is called *damping parameter* and updated at each iteration

Convergence rate of unconstrained methods

under the assumption that $x^{(k)} \to x^\star$ and f is generally nonlinear

methods	convergence rate	property
gradient descent	linear	first-order method
Newton	quadratic	second-order method
		expensive for large scale problems
Quasi Newton	superlinear	first-order method
CG for quadratic	n-step	first-order method
		only require matrix-vector products
CG for nonlinear f	global convergence	first-order method

Softwares

MATLAB: optimization toolbox

`fminunc` uses quasi-newton and trust-region

- quasi-newton: requires description of f, uses relative optimality tolerance, relative step tolerance
- trust-region: requires description of f and ∇f, uses absolute optimality tolerance, relative function tolerance, and absolute step tolerance
- `https://www.mathworks.com/help/optim/ug/fminunc.html`

`fminsearch` uses a derivative-free method

Python: scipy.optimize

- several methods including BFGS, Newton-conjugate-gradient, trust-region Newton-conjugate-gradient, trust-region truncated generalized Lanczos, trust-region nearly exact, Nelder-Mead simplex (derivative free method)
- `https://docs.scipy.org/doc/scipy/tutorial/optimize.html`

Nonlinear least-squares

MATLAB: optimization toolbox: lsqnonlin

- trust-region reflective (default) requires that the nonlinear system $r(x) \in \mathbf{R}^q$ cannot be underdetermined, *i.e.*, $q \geq n$
- `https://www.mathworks.com/help/optim/ug/lsqnonlin.html`
- `curvefit` solves a curve fitting problem, which is an application of NLS

Python: scipy.optimize.least_squares

- trust-region reflective is suitable for large sparse problems
- LM does not handle bound constraints and it does not work for under-determined nonlinear system
- another choice: **scipy.optimize.leastsq** solves the NLS without bounds
- **scipy.optimize.curve_fit** solves a curve-fitting problem using NLS

Accelerated gradient methods

Accelerated gradient methods

assumptions:

- f is convex and differentiable
- $\nabla f(x)$ is Lipschitz continuous with constant L

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$$

- optimal value $f^\star = \inf_x f(x)$ is finite and attained at $x^\star$

applying the following methods to the function class in the assumptions

- FISTA (Fast iterative shrinkage-thresholding algorithm)
- Nesterov's method

have $\mathcal{O}(1/k^2)$ convergence (improvement over the gradient method with rate $\mathcal{O}(1/k)$)

FISTA (Beck and Teboulle 2009)

initializes $x^{(0)}$ and set $y^{(1)} = x^{(0)}$, $\gamma_1 = 1$

$$\begin{aligned} x^{(k)} &= y^{(k)} - t_k \nabla f(y^{(k)}) \\ \gamma_{k+1} &= \frac{1 + \sqrt{1 + 4\gamma_k^2}}{2} \\ y^{(k+1)} &= x^{(k)} + \left(\frac{\gamma_k - 1}{\gamma_{k+1}}\right)\left(x^{(k)} - x^{(k-1)}\right) \end{aligned}$$

constant step size $t_k = 1/L$ (if L is known); otherwise, use backtracking

a convergence result showed that $f(x^{(k)}) - f^\star \leq \frac{2L\|x^{(0)} - x^\star\|_2^2}{(k+1)^2}$ (for constant step size)

Line search

before the update of x in iteration k, find a suitable t_k

$$
\begin{aligned}
& t := t_{k-1}, \quad \text{(define } t_0 = \hat{t} > 0) \\
& x := y - t\nabla f(y) \\
& \quad \text{while } \; f(x) > f(y) - \frac{t}{2}\|\nabla f(y)\|_2^2 \\
& \qquad t := \beta t, \quad \text{with } \beta < 1 \\
& \qquad x := y - t\nabla f(y) \\
& \quad \text{end}
\end{aligned}
$$

Lipschitz continuity of ∇f guarantees $t_k \geq t_{\min} = \min\{\hat{t}, \beta/L\}$

Nesterov's method

the Nesterov's second method (as algorithm 1 from Tseng 2008)

choose any sequence satisfying

$$\theta_0 \in (0,1] \quad \text{and} \quad \frac{1-\theta_k}{\theta_k^2} \leq \frac{1}{\theta_{k-1}^2}, \quad k \geq 2, \quad (\text{e.g., } \theta_k = \frac{2}{k+2} \;)$$

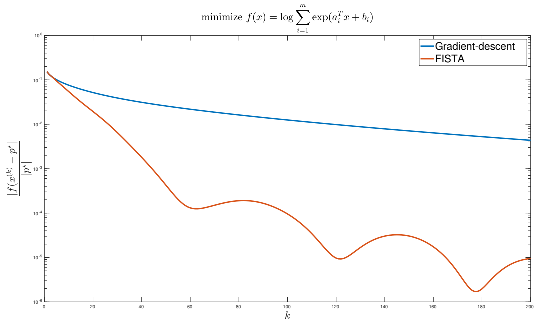
algorithm: choose $x^{(0)} = v^{(0)}$; for $k \geq 1$, repeat the steps

$$\begin{aligned} y &= (1-\theta_k)x^{(k-1)} + \theta_k v^{(k-1)} \\ v^{(k)} &= v^{(k-1)} - \frac{t_k}{\theta_k}\nabla f(y) \\ x^{(k)} &= (1-\theta_k)x^{(k-1)} + \theta_k v^{(k)} \end{aligned}$$

- $t_k = 1/L$ or use line search if L is unknown
- convergence: $f(x^{(k)}) - f^\star$ decreases with rate $\mathcal{O}(1/k^2)$

Result of FISTA

minimize $f(x) = \log\left(\sum_{i=1}^{m} e^{a_i^T x + b_i}\right)$ (convex problem)



- $n = 100, m = 200$ where a_i, b_i are randomly generated; using fixed $t = 0.1$
- faster convergence of FISTA but $f(x^{(k)})$ is not monotonically decreasing
- the descent version of FISTA can be found in Beck and Taboulle 2009

Further notes

- this lecture presents the accelerated gradient methods for

$$\underset{x}{\text{minimize}} \quad f(x)$$

  where f is *convex* and ∇f is Lipschitz continuous
- however, FISTA and Nesterov's method were originally proposed for a wider class

$$\underset{f}{\text{minimize}} \quad f(x) := g(x) + h(x)$$

  where g is continuously differentiable convex while h can be closed and convex (but not necessarily differentiable)
- we will revisit the two methods again in the topic of proximal algorithms

References

conventional algorithms for differentiable f

1. Lecture notes on *Optimization Methods for Large-Scale Systems*, EE263C, L. Vandenberhge, UCLA
2. S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, 2004
3. Chapter 12-13 in I. Griva, S.G. Nash, and A. Sofer, *Linear and Nonlinear Optimization*, SIAM, 2009
4. Chapter 5 in J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer 2006

accelerated gradient methods for convex f

1. A. Beck and M. Teboulle, *A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems*, SIAM J. Imaging Sciences, 2009
2. P. Tseng, *On Accelerated Proximal Gradient Methods for Convex-Concave Optimization*, Technical Report, 2008