Module 07
Optimization Algorithms

Ahmad F. Taha

CE 5999-02 Special Topics — Intro to Optimization

Email: ahmad.taha@vanderbilt.edu

Webpage: http://lab.vanderbilt.edu/taha

VANDERBILT UNIVERSITY

April 2, 2023

In this module

- General concepts
- Unconstrained minimization algorithms
- Newton's method to solve a system of nonlinear equations
- Constrained minimization algorithms with projection
- Minimization with linear equality constraints
- Barrier method for general convex problems

Optimization algorithm

Rule to provide a sequence $x^{(0)}, x^{(1)}, x^{(2)}, \ldots$

$$x^{(k+1)} = F^{(k)}(x^{(k)})$$

and a termination criterion.

Desirable properties of the algorithm

- If the algorithm is started at the problem solution, then it should not move
- Convergence
  Convergence can be expressed in several ways, e.g.,

$$\lim_{k\to\infty} f(x^{(k)}) = f(x^*)$$

$$\lim_{k\to\infty} x^{(k)} = x^*$$

- Convergence speed
  How fast does the algorithm converge?
  How many iterations does it take to guarantee that $f(x^{(k)}) - f^* \leq \varepsilon$?
- Complexity analysis
  How many operations (additions, multiplications, ...) does it take to converge, as a function of the problem size (number of variables, number of constraints)?

Unconstrained minimization algorithms

- Algorithms to solve

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

- Interpreted as methods to solve $\nabla f(x) = 0$
- If the problem is convex, convergence to the global minimum
- If the problem is not convex, convergence to a point satisfying $\nabla f(x) = 0$ (which may or may not be local minimum)
- Points satisfying $\nabla f(x) = 0$ are called stationary points

Descent algorithms

$$x^{(k+1)} = x^{(k)} + t^{(k)} \Delta x^{(k)} \text{ with } f(x^{(k+1)}) < f(x^{(k)})$$

- $\Delta x^{(k)}$: Search direction (vector in $\mathbb{R}^n$)
- $t^{(k)} > 0$: Stepsize

General descent algorithm:

Require: Starting point $x \in \mathrm{dom}(f)$
  repeat
    Compute search direction Δx
    Line search: Compute step size t
    Update: $x^+ := x + t\Delta x$
  until convergence criterion is satisfied

Search directions for descent

- To guarantee descent, it is necessary to choose $\Delta x^{(k)}$ so that $\nabla f(x^{(k)})^T \Delta x^{(k)} < 0$
- You can see this geometrically for a simple function $f(x) = x^2$
- For a nonconvex function, use Taylor's 1st-order approximation

$$f(x^{(k+1)}) = f(x^{(k)} + t^{(k)} \Delta x^{(k)}) \approx f(x^{(k)}) + t^{(k)} \nabla f(x^{(k)})^T \Delta x^{(k)}$$

- **Bottomline:** If $\nabla f(x^{(k)})^T \Delta x^{(k)} < 0$ and $t^{(k)}$ is sufficiently small, descent can be guaranteed (even if the function is nonconvex)
- $t^{(k)}$ **sufficiently small is not enough for convergence**
- Proper value of $t^{(k)}$ must be selected through *line search*

Typical search directions

1. Steepest descent: $\Delta x^{(k)} = -\nabla f(x^{(k)})$
2. Diagonally scaled steepest descent: $\Delta x^{(k)} = -D^{(k)} \nabla f(x^{(k)})$
   - $D^{(k)}$ is a diagonal matrix with positive entries on the diagonal
3. Newton's method: $\Delta x^{(k)} = -(\nabla^2 f(x^{(k)}))^{-1} \nabla f(x^{(k)})$
4. Levenberg-Marquardt modification of Newton's method: $\Delta x^{(k)} = -(\mu_k I + \nabla^2 f(x^{(k)}))^{-1} \nabla f(x^{(k)})$ with $\mu_k > 0$
5. Quasi-Newton methods: $\Delta x^{(k)} = -H^{(k)} \nabla f(x^{(k)})$ with $H^{(k)} \succ 0$
   - Generally, $H^{(k)}$ is selected so that $\Delta x^{(k)}$ approximates the Newton direction
   - A quasi-Newton method (together with line search) is one of the algorithms used by MATLAB's `fminunc`, which is a solver for unconstrained minimization
6. Conjugate gradient methods
7. Gauss-Newton method for nonlinear least squares

Typical stepsize selection methods

1. Exact line search: t solves $\min_{t>0} f(x + t\Delta x)$
   - Inexact line search: Minimize a quadratic or cubic interpolation of $g(t) = f(x + t\Delta x)$ with respect to t
2. Backtracking line search (Armijo rule)
   - In contrast to exact line search, backtracking only needs function evaluations
3. Constant stepsize: $t^{(k)} = t > 0$ for all $k = 0, 1, 2, \ldots$
   - Simple choice: no need for minimization as with the exact line search, no need for function evaluations as with backtracking
   - Convergence to stationary point under certain conditions
   - In practice, set t to a small value; in case of divergence, reduce t
4. Diminishing stepsize: $t^{(k)} \to 0$ as $k \to \infty$ and $\sum_{k=1}^{\infty} t^{(k)} = \infty$
   - This stepsize has good theoretical properties, but is slow in practice
   - Example: $t^{(k)} = \frac{a}{b+k}$

Backtracking line search

Parameters: $0 < \beta < 1$, $0 < \alpha < 1/2$

- Start with $t = 1$
- If $f(x + t\Delta x) < f(x) + \alpha t \nabla f(x)^T \Delta x$, accept t and exit
- Otherwise, set $t := \beta t$, and check the condition again
- As long as $\nabla f(x)^T \Delta x < 0$, the set of acceptable stepsizes t will always contain an interval of the form $[0, \delta]$, and the line search will finish in a finite number of steps

Inexact line search

- Main idea is to use a quadratic $h(t) = q_1 t^2 + q_2 t + q_3$ or a cubic $h(t) = c_1 t^3 + c_2 t^2 + c_3 t + c_4$ to approximate $g(t) = f(x + t\Delta x)$
- Start with an interval $[a, b]$ containing the minimum of $g(t) = f(x + t\Delta x)$ and possibly an additional point $c \in [a, b]$ such that $g(c) < g(a)$ and $g(c) < g(b)$
- Find the parameters of $h(t)$ using some combination of $h(a) = g(a)$, $h(b) = g(b)$, $h(c) = g(c)$, $h'(a) = g'(a)$, etc.
- Then, find the minimizer of the resulting $h(t)$, update the interval, and repeat
- Useful formula for implementation: $g'(t) = \nabla f(x + t\Delta x)^T \Delta x$

Convergence of gradient methods

- Generally, convergence to a *stationary point* is guaranteed when
  1. exact/inexact line minimization or backtracking line search is used, and
  2. the search direction is chosen so that $\nabla f(x^{(k)})^T \Delta x^{(k)} < 0$
- Additional conditions on f can guarantee convergence under the (simpler) constant or dimishing stepsize rules.
- For example, the steepest descent with constant stepsize $x^{(k+1)} = x^{(k)} - t\nabla f(x^{(k)})$ converges to a stationary point, that is, $\lim_{k\to\infty} \|\nabla f(x^{(k)})\|_2 = 0$, when the gradient is Lipschitz continuous

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

and the stepsize is chosen so that

$$0 < t < \frac{1}{L}$$

Convergence: Convexity vs. nonconvexity

- The statements on the previous slide apply to convex as well as nonconvex problems
- For convex problems, stationary points ($\nabla f(x) = 0$) are optimal
- For nonconvex problems, convergence to a stationary point is guaranteed (under conditions)
- For nonconvex problems, the convergence point may be local minimum and in general depends on the initialization point x^0
- Running the algorithm with multiple initializations and selecting the convergence point that yields the smallest objective value is an option in this case

Strong convexity and implications

- Consider the sublevel set defined by the initialization point $x^{(0)}$

$$S = \{x \in \mathrm{dom}(f) \,|\, f(x) \leq f(x^{(0)})\}$$

  and assume it is closed
- f is **strongly convex on** S if there exists $m > 0$ such that

$$\nabla^2 f(x) \succeq mI \text{ for all } x \in S$$

- Descent algorithms guarantee that the sequence $\{x^{(k)}\}_{k=0}^{\infty}$ remains in S
- If f is strongly convex with parameter m, then the suboptimality of any point $x \in S$ can be estimated as follows (see Sec. 9.1.2 of textbook for proof)

$$f(x) - f^* \leq \frac{1}{2m} \|\nabla f(x)\|_2^2$$

Convergence of steepest descent under strong convexity

- Consider the steepest descent: $\Delta x = -\nabla f(x)$
- If f is strongly convex with constant m and exact or backtracking line search is used, then the distance of $f(x^{(k)})$ from f^* can be bounded as follows:

$$f(x^{(k)}) - f^* \leq c^k (f(x^{(0)}) - f^*)$$

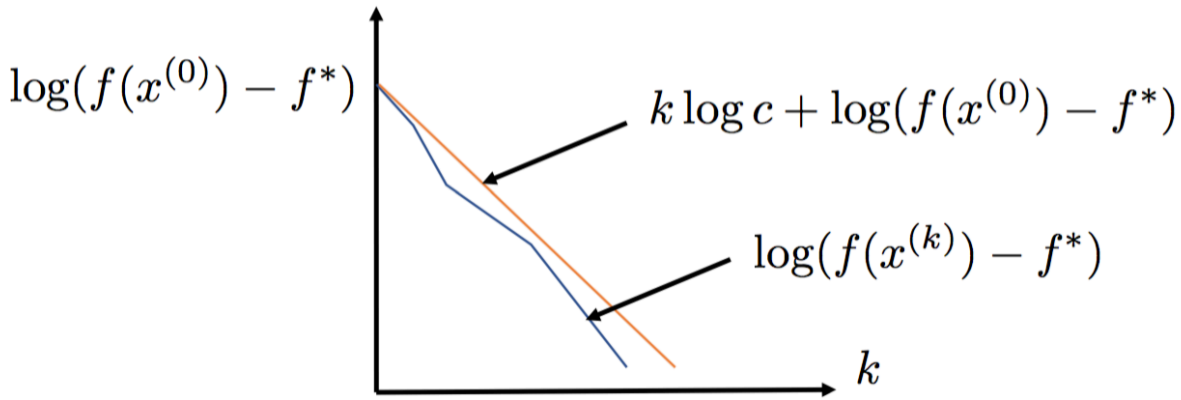
  where $0 < c < 1$ is a constant that depends on f and the line search method
  - See Sec. 9.3.1 of textbook for proof

Convergence rate

- Taking the logarithm on both sides, we obtain

$$\log_{10}(f(x^{(k)}) - f^*) \leq k \log_{10} c + \log_{10}(f(x^{(0)}) - f^*)$$

- If we plot the error $f(x^{(k)}) - f^*$ on a logarithmic scale versus k, it will be upper-bounded by a *line* with slope $\log_{10} c < 0$ and intercept $\log_{10}(f(x^{(0)}) - f^*)$
  - Slope depends on f and the line search method
  - Intercept depends on the initial suboptimality
- The algorithm is said to have *linear convergence rate*



Convergence (termination) criteria

1. Scale-free
   - $\|\nabla f(x^{(k)})\| \leq \epsilon \|\nabla f(x^{(0)})\|$
   - $\|x^{(k)} - x^{(k-1)}\| \leq \epsilon \|x^{(k)}\|$
2. Scale-dependent
   - $\|\nabla f(x^{(k)})\| \leq \epsilon$
   - $\|x^{(k)} - x^{(k-1)}\| \leq \epsilon$

- Scale-free are preferable to scale-dependent
- Setting of ϵ may require trial and error
- Algorithm specific criteria also exist
- If f is strongly convex with known strong convexity constant, a termination criterion based on gradient can guarantee a suboptimality level:

$$f(x^{(k)}) - f^* \leq \frac{1}{2m} \|\nabla f(x^{(k)})\|_2^2 \leq \epsilon$$

Newton's method

- Newton direction

$$\Delta x = -(\nabla^2 f(x))^{-1} \nabla f(x)$$

- If $\nabla^2 f(x) \succ 0$, then Δx is a descent direction

$$-\nabla f(x)^T (\nabla^2 f(x))^{-1} \nabla f(x) < 0$$

unless $\nabla f(x) = 0$, which implies that x is a stationary point

Newton's method:

Require: Starting point $x \in \mathrm{dom}(f)$

repeat

Compute search direction: Solve the linear system $\nabla^2 f(x) \Delta x = -\nabla f(x)$

Line search: Compute step size t by backtracking line search

Update: $x^+ := x + t\Delta x$

until convergence criterion is satisfied

Interpretation of the Newton direction

- Form the quadratic approximation of f at $x = x^{(k)}$:

$$f_{\text{quad}}(x + \Delta x) = f(x) + \nabla f(x)^T \Delta x + \frac{1}{2} \Delta x^T \nabla^2 f(x) \Delta x$$

- $f_{\text{quad}}(x + \Delta x)$ is convex quadratic with respect to Δx when $\nabla^2 f(x) \succ 0$
- Main idea: Instead of minimizing $f(x)$, minimize its approximation $f_{\text{quad}}(x + \Delta x)$ with respect to Δx
- Setting the gradient of $f_{\text{quad}}(x + \Delta x)$ with respect to Δx to zero, we get the optimality condition and the Newton update

$$\nabla^2 f(x) \Delta x = -\nabla f(x) \Longrightarrow \Delta x = -(\nabla^2 f(x))^{-1} \nabla f(x)$$

- If f is quadratic, Newton's algorithm will converge in one step
- If f is nearly quadratic, the algorithm should need very few steps

Convergence of Newton's method: Assumptions

- Consider the sublevel set defined by the initialization point $x^{(0)}$

$$S = \{x \in \mathrm{dom}(f) \mid f(x) \leq f(x^{(0)})\}$$

  and assume it is closed
- f is strongly convex on S with strong convexity constant m
- The Hessian of f is Lipschitz continuous on S with constant L:

$$\|\nabla^2 f(x) - \nabla^2 f(y)\|_2 \leq L\|x - y\|_2 \text{ for all } x, y \in S$$

  - $L = 0$ if f is a quadratic function
  - L measures how close f is to a quadratic

Unguarded Newton's method

Unguarded Newton's method:

Require: Starting point $x \in \mathrm{dom}(f)$

repeat

Compute search direction: Solve the linear system $\nabla^2 f(x)\Delta x = -\nabla f(x)$

Set step size $t = 1$

Update: $x^+ := x + t\Delta x$

until convergence criterion is satisfied

- Can be expected to work if $x^{(0)}$ is close to x^*

Levenberg-Marquardt Modification of Newton's method

- Newton's method can fail if $\nabla^2 f(x)$ is singular (Newton direction is not defined) or $\nabla^2 f(x) \prec 0$ (Newton direction is not a descent direction)
- Newton's method: $x^{(k+1)} = x^{(k)} - t^{(k)}[\nabla^2 f(x^{(k)})]^{-1}\nabla f(x^{(k)})$
- Levenberg-Marquardt modification:
  $x^{(k+1)} = x^{(k)} - t^{(k)}[\mu_k I + \nabla^2 f(x^{(k)})]^{-1}\nabla f(x^{(k)})$ with $\mu_k > 0$
- Search direction $\Delta x^{(k)} = -[\mu_k I + \nabla^2 f(x^{(k)})]^{-1}\nabla f(x^{(k)})$ can always be made to be a descent direction for sufficiently large μ_k
- Method approaches the behavior of Newton's method for $\mu_k \to 0$ and of a gradient method for $\mu_k \to \infty$
- Implementation aspects
  - Start with a small value of μ_k and increase it slowly until the iteration is descent: $f(x^{(k+1)}) < f(x^{(k)})$
  - Search direction computed as solution to $(\mu I + \nabla^2 f(x))\Delta x = -\nabla f(x)$
  - Combine with line search

Some observations

- Newton's method has several advantages over gradient and steepest descent
  - Convergence of Newton's method is rapid in general and quadratic near x^*
  - Once quadratic convergence phase is reached, at most six or so iterations are required to produce a solution
  - Newton's method is *affine invariant*: it is insensitive to the choice of coordinates, or the condition number of the sublevel sets of the objective
  - Newton's method scales well with problem size: its performance on problems in $\mathbb{R}^{10000}$ is similar to its performance on problems in $\mathbb{R}^{10}$, with only a small increase in the number of steps required
  - The good performance of Newton's method is not dependent on the choice of algorithm parameters
  - In contrast, the choice of norm for steepest descent plays a critical role in its performance
- Main disadvantage: the cost of forming and storing the Hessian, cost of computing the Newton step, which requires solving a set of linear equations.
- But you can exploit problem structure

Conjugate Gradient Algorithms

- How about an algorithm that is an intermediate between the method of steepest descent and Newton's method?
  - Solve quadratics of n variables in n steps
  - Requires no Hessian matrix evaluations
  - No matrix inversion and no storage of an $n \times n$ matrix required
- This method performs better than steepest descent, but not as well as Newton's
- The crucial factor in the efficiency of an iterative search method is the direction of search
- Need a definition: Let Q be a real symmetric $n \times n$ matrix. The directions $d^{(0)}, d^{(1)}, \ldots$ are Q-conjugate if, for all $i \neq j$, we have $d^{(i)T} Q d^{(j)} = 0$

Conjugate Gradient Algorithm for Quadratic Minimization

- Let's focus on this simply quadratic minimization problem $f(x) = 1/2x^T Qx - x^T b,\ Q \succ 0$
- In this algorithm, you can prove that $\alpha_k = \text{argmin}_{\alpha_k \geq 0} f(x^{(k)} + \alpha d^{(0)})$ has a closed form expression written below
- You can see that the directions $d^{(0)}, d^{(1)}, \ldots$ are indeed Q-conjugate

1. Set $k := 0$; select the initial point $\boldsymbol{x}^{(0)}$.
2. $\boldsymbol{g}^{(0)} = \nabla f(\boldsymbol{x}^{(0)})$. If $\boldsymbol{g}^{(0)} = \boldsymbol{0}$, stop, else set $\boldsymbol{d}^{(0)} = -\boldsymbol{g}^{(0)}$.
3. $\alpha_k = -\dfrac{\boldsymbol{g}^{(k)T}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)T}\boldsymbol{Q}\boldsymbol{d}^{(k)}}$.
4. $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$.
5. $\boldsymbol{g}^{(k+1)} = \nabla f(\boldsymbol{x}^{(k+1)})$. If $\boldsymbol{g}^{(k+1)} = \boldsymbol{0}$, stop.
6. $\beta_k = \dfrac{\boldsymbol{g}^{(k+1)T}\boldsymbol{Q}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)T}\boldsymbol{Q}\boldsymbol{d}^{(k)}}$.
7. $\boldsymbol{d}^{(k+1)} = -\boldsymbol{g}^{(k+1)} + \beta_k \boldsymbol{d}^{(k)}$.
8. Set $k := k + 1$; go to step 3.

Jacobian of a vector-valued function

Consider a function $G = \begin{bmatrix} G_1(x) \\ \vdots \\ G_n(x) \end{bmatrix} : \mathbb{R}^n \to \mathbb{R}^n$.

Jacobian matrix or **derivative matrix** of G in $n \times n$ matrix

$$\mathrm{D}G(x) = \begin{bmatrix} \frac{\partial G_1}{\partial x_1} & \cdots & \frac{\partial G_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial G_n}{\partial x_1} & \cdots & \frac{\partial G_n}{\partial x_n} \end{bmatrix} = \left[\frac{\partial G_i}{\partial x_j} \right]_{i,j}$$

Definitions in other books/papers may use the transpose of the above. Be careful.

Nonlinear least squares

- Nonlinear least squares are problems of the form

$$\text{minimize} \quad \sum_{i=1}^{m} r_i^2(x) = \|r(x)\|_2^2$$

- Functions $r_i : \mathbb{R}^n \to \mathbb{R}$ are given; define the vector of residuals

$$r(x) = \begin{bmatrix} r_1(x) \\ \vdots \\ r_m(x) \end{bmatrix}$$

- Linear least squares: the vector of residuals is an affine function of x

$$r(x) = Ax - b$$

- Examples of nonlinear least squares problems
  - Estimate the parameters of sinusoidal functions from measurements (see next)
  - Neural network training: Find the weights in activation functions of multi-layer neural networks

Example: Find the parameters of sinusoidal functions

- Sinusoidal function $A\sin(\omega t + \phi)$ with unknown $x = [A, \omega, \phi]^T$
- Obtain samples (possibly noisy) of the signal at times $t_1, t_2, \ldots, t_m$

$$y_i = A\sin(\omega t_i + \phi) + w_i$$

where w_i is the noise realization

- To find the unknown signal parameters in x, formulate the nonlinear least squares problem

$$\text{minimize} \quad \sum_{i=1}^{m} (y_i - A\sin(\omega t_i + \phi))^2$$

- Residual function is

$$r_i(x) = r_i(A, \omega, \phi) = y_i - A\sin(\omega t_i + \phi)$$

Gradient and Hessian of nonlinear least squares objective

- Define the objective function
  $f(x) = \frac{1}{2}\sum_{i=1}^{m} r_i^2(x) = \frac{1}{2}\|r(x)\|_2^2 = \frac{1}{2}r(x)^T r(x)$
- The Jacobian matrix of $r(x)$ is denoted by $J(x)$, that is, $[J(x)]_{ij} = \frac{\partial r_i(x)}{\partial x_j}$
- The gradient of f is

$$\nabla f(x) = J(x)^T r(x)$$

- The Hessian of f is

$$\nabla^2 f(x) = J(x)^T J(x) + \sum_{i=1}^{m} r_i(x)\nabla^2 r_i(x)$$

- Where the corresponding entries are computed as follows

$$\frac{\partial f(x)}{\partial x_j} = \sum_{i=1}^{m} r_i(x)\frac{\partial r_i(x)}{\partial x_j}$$

$$\frac{\partial^2 f}{\partial x_k \partial x_j} = \frac{\partial}{\partial x_k}\left[\sum_{i=1}^{m} r_i(x)\frac{\partial r_i(x)}{\partial x_j}\right] = \sum_{i=1}^{m}\left[\frac{\partial r_i(x)}{\partial x_k}\frac{\partial r_i(x)}{\partial x_j} + r_i(x)\frac{\partial^2 r_i(x)}{\partial x_k \partial x_j}\right]$$

Gauss-Newton method

- Newton's iteration would be

$$x^{(k+1)} = x^{(k)} - t^{(k)} \left[J(x^{(k)})^T J(x^{(k)}) + \sum_{i=1}^{m} r_i(x^{(k)}) \nabla^2 r_i(x^{(k)}) \right]^{-1} J(x^{(k)})^T r(x^{(k)})$$

- We would like to avoid the computation of second derivatives involved in the second part of the Hessian

$$\nabla^2 f(x) = J(x)^T J(x) + \sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x)$$

- In practice, we ignore the term $\sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x)$
  - In many applications, this term are neglibly small, especially near the solution
  - For example, we may have $r_i(x) \approx 0$ near the solution
- Gauss-Newton method

$$x^{(k+1)} = x^{(k)} - t^{(k)} [J(x^{(k)})^T J(x^{(k)})]^{-1} J(x^{(k)})^T r(x^{(k)})$$

Implementation aspects

- To ensure descent, we need $J(x^{(k)})^T J(x^{(k)})$ nonsingular
- This can be ensured with the Levenberg-Marquardt modification

$$x^{(k+1)} = x^{(k)} - t^{(k)}[\mu_k I + J(x^{(k)})^T J(x^{(k)})]^{-1} J(x^{(k)})^T r(x^{(k)})$$

- The above is referred to as the *Levenberg-Marquardt algorithm*, because the modification was originally developed specifically for the nonlinear least squares
- The method is combined with line search

Newton-Raphson method for nonlinear equations: Main idea

- We wish to solve the $n \times n$ nonlinear system of equations

$$G(x) = 0$$

- Our current iterate is $x^{(k)}$. We want to go from $x^{(k)}$ to $x^{(k+1)}$ and hit the solution

$$G(x^{(k+1)}) \approx 0$$

- Let's use the 1st-order Taylor approximation of G around $x^{(k)}$

$$G(x^{(k+1)}) = G(x^{(k)} + (x^{(k+1)} - x^{(k)})) = G(x^{(k)}) + \mathrm{D}G(x^{(k)})(x^{(k+1)} - x^{(k)})$$

- Using $G(x^{(k+1)}) = 0$ in the above, we can solve for $x^{(k+1)}$:

$$x^{(k+1)} = x^{(k)} - (\mathrm{D}G(x^{(k)}))^{-1} G(x^{(k)})$$

Newton-Raphson method for solving a nonlinear system of equations

Newton-Raphson method for nonlinear system of equations:

Require: Starting point $x \in \mathrm{dom}(G)$

  repeat

    Compute search direction: Solve the linear system $\mathrm{D}G(x)\Delta x = G(x)$

    Update: $x^+ := x + \Delta x$

  until convergence criterion is satisfied

Connection to Newton's method for optimization

- (Unguarded) Newton's method for unconstrained minimization of $f(x)$:

$$x^{(k+1)} = x^{(k)} - (\nabla^2 f(x^{(k)}))^{-1} \nabla f(x^{(k)})$$

- Can be thought of as Newton-Raphson method to solve the 1st-order optimality condition

$$G(x) = \nabla f(x) = 0$$

- What is the Jacobian of $\nabla f(x)$? It becomes the Hessian of f:

$$\mathrm{D}G(x) = \mathrm{D}\nabla f(x) = \left[\frac{\partial G_i}{\partial x_j}\right]_{i,j} = \left[\frac{\partial}{\partial x_j}\left(\frac{\partial f}{\partial x_i}\right)\right]_{i,j} = \left[\frac{\partial^2 f}{\partial x_i x_j}\right]_{i,j}$$

Convergence

- Newton-Raphson method for solving nonlinear equations is more general than Newton's method for unconstrained optimization
- Given $G(x) : \mathbb{R}^n \to \mathbb{R}^n$, when can you find an $f(x)$ so that $G(x) = \nabla f(x)$?
  - It can be done when $\mathrm{D}G(x)$ is a *symmetric* matrix
  - In this case, $\mathrm{D}G(x)$ is the Hessian, $\nabla^2 f(x)$, and the method minimizes $f(x)$
  - The availability of an objective function $f(x)$ means that we can also do line search
- Given $G(x) = 0$, it is generally not expected to find an $f(x)$ so that $G(x) = \nabla f(x)$
- Regarding the convergence of Newton's method for solving nonlinear equations, we can borrow some intuition from the version for unconstrained optimization
- Iteration $x^{(k+1)} = x^{(k)} - (\mathrm{D}G(x^{(k)}))^{-1} G(x^{(k)})$ corresponds to unguarded Newton's method (without line search)
- Local convergence can be expected (when $x^{(0)}$ is close to x^*, where $G(x^*) = 0$)

Methods for constrained minimization

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subj. to} \quad & x \in C \end{aligned}$$

- C is a convex set; f not necessarily convex
- If f is convex, convergence to a global optimum
- If f is not convex, convergence to a stationary point, i.e., a point that satisfies the first-order necessary optimality condition

$$\nabla f(x^*)^T (x - x^*) \geq 0 \text{ for all } x \in C$$

Projection onto a set

Consider a **convex** set $C \subset \mathbb{R}^n$ and a point $z \in \mathbb{R}^n$ (not necessarily in C). The *projection* of point z onto set C is denoted by $[z]_C$ and is defined as the solution of the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \|y - z\|_2^2 \\ \text{subj. to} \quad & y \in C \end{aligned}$$

The above optimization is convex. In fact, the objective is *strictly convex*, therefore the projection is *unique*.
If $z \in C$, then $[z]_C = z$.
Important example: When C is a box, $C = \{x \,|\, l \preceq x \preceq u\}$, then entry i of $[z]_C$ is

$$([z]_C)_i = \begin{cases} z_i, & \text{if } l_i \leq z_i \leq u_i \\ l_i, & \text{if } z_i < l_i \\ u_i, & \text{if } z_i > u_i. \end{cases}$$

A simple gradient projection method

$$x^{(k+1)} = [x^{(k)} - t^{(k)} \nabla f(x^{(k)})]_C$$

- After taking a steepest descent step, project onto C
- Convergence guaranteed with constant stepsize $t^{(k)} = t > 0$ sufficiently small under Lipschitz continuity of the gradient vector
- Several other search directions and step size selections possible (to a large extent, analogs of the ones in gradient methods for unconstrained minimization)
- Method is reasonable when projection is computationally cheap (e.g., onto a box)

Equality-constrained minimization

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & Ax = b \end{aligned}$$

- Variable $x \in \mathbb{R}^n$, data $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$; assume $\mathrm{rank}(A) = m$ without loss of generality
- Two methods to solve
- Eliminate equality constraints and reduce to an unconstrained problem
- Newton's method for the constrained problem

Newton's method for equality constrained minimization: Main idea

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & Ax = b \end{aligned}$$

- Suppose we are at *feasible* x (i.e., $Ax = b$), want to move to $x + v$
- Approximate the objective with a quadratic in v

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v \\ \text{subject to} \quad & A(x+v) = b \end{aligned}$$

Computing the Newton step

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v \\ \text{subject to} \quad & A(x+v) = b \end{aligned}$$

- The Newton step is $\Delta x_{\text{nt}} = v$, where v solves the optimality condition

$$\begin{bmatrix} \nabla^2 f(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}$$

Newton's method for equality constrained minimization

Require: Starting point $x \in \mathrm{dom}(f)$ satisfying $Ax = b$

repeat

Compute search direction $\Delta x_{\mathrm{nt}} = v$ as solution of the linear system

$$\begin{bmatrix} \nabla^2 f(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}$$

Line search: Compute step size t via backtracking

Update: $x^+ := x + t\Delta x_{\mathrm{nt}}$

until convergence criterion is satisfied

The algorithm ensures that all iterates remain feasible!

General convex optimization problem

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subj. to} \quad & f_i(x) \leq 0, \ i = 1, \ldots, m \\ & Ax = b \end{aligned}$$

- $f_i(x)$ $(i = 0, \ldots, m)$ convex, twice differentiable
- $A \in \mathbb{R}^{p \times n}$, $\text{rank}(A) = p$ (we have p linearly independent equality constraints)
- Optimal value f^* is finite (we will not solve an unbounded problem)
- A constraint qualification holds, so we have strong duality, $d^* = f^*$
- For example, these exist $\tilde{x}$ such that $f_i(\tilde{x}) < 0$ $(i = 1, \ldots, m)$ and $A\tilde{x} = b$

Reformulation via logarithmic barrier

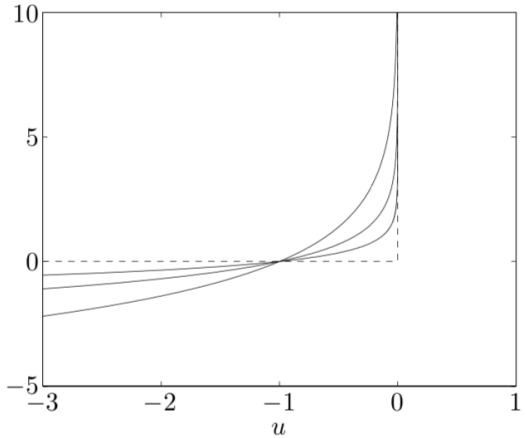
- The original problem is equivalent to the following using

$$\begin{aligned} \min \quad & f_0(x) + \sum_{i=1}^{m} I_-(f_i(x)) \\ \text{subj. to} \quad & Ax = b \end{aligned}$$

where $I_-(u) = 0$ if $u \leq 0$ and $I_-(u) = \infty$ otherwise is the indicator function for $\mathbb{R}_-$

- The indicator function is approximated as $I_-(u) \approx \frac{1}{t}\log(-u)$ for $t > 0$

$$\begin{aligned} \min \quad & f_0(x) - \frac{1}{t}\sum_{i=1}^{m} \log(-f_i(x)) \\ \text{subj. to} \quad & Ax = b \end{aligned}$$



Boyd & Vand., *Convex Opt.*

- Convex problem with equality constraints only and parameter $t > 0$
- The approximation improves as $t \to \infty$

Logarithmc barrier function

- Logarithmic barrier function is convex if $f_i(x)$, $i = 1, \ldots, m$, are convex

$$\phi(x) = -\sum_{i=1}^{m} \log(-f_i(x))$$

- Gradient

$$\nabla\phi(x) = \sum_{i=1}^{m} \frac{1}{-f_i(x)} \nabla f_i(x)$$

Centering problem and central path

- Multiply the objective function by t and the problem is equivalent to

$$\begin{aligned} \min \quad & t f_0(x) + \phi(x) \\ \text{subj. to} \quad & Ax = b \end{aligned}$$

- The above is called the *centering problem*
- Let $x^*(t)$ be its solution for a given $t > 0$
- As $t \to \infty$, the solution $x^*(t)$ approaches the solution of the original problem
- The sequence of resulting objective values $\{x^*(t) \,|\, t > 0\}$ is called the *central path*

Central path example

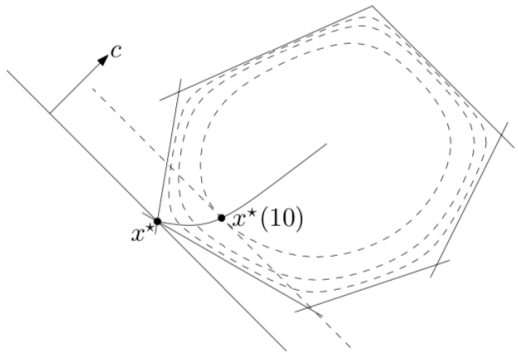
- LP with $x \in \mathbb{R}^2$ and 6 inequality constraints

$$\begin{aligned} \min \quad & c^T x \\ \text{subj. to} \quad & a_i^T x \leq b_i, \ i = 1, \ldots, 6 \end{aligned}$$

- Feasible set is defined by a polyhedron with 6 facets
  - For $t = 0$, the centering problem is

$$\min \quad \sum_{i=1}^{6} \log(b_i - a_i^T x)$$

  - This problem finds a center of the polyhedron
  - As t increases, centering path is shown as a solid line joining the center of the polyhedron to the corner x^*, which is the solution of the LP



Boyd & Vandenberghe, *Convex Opt.*

- Figure also shows the solution of the centering problem for $t = 10$

KKT conditions of the centering problem

- Let w be the Lagrange multiplier to $Ax - b = 0$ for the centering problem
- $x^*(t)$ satisfies the KKT conditions

$$\nabla \Big(t f_0(x) + \phi(x) + w^T (Ax - b) \Big) = 0, \quad Ax = b$$

or equivalently

$$t \nabla f_0(x) + \sum_{i=1}^{m} \frac{1}{-f_i(x)} \nabla f_i(x) + A^T w = 0, \quad Ax = b$$

- Divide by t

$$\nabla f_0(x) + \sum_{i=1}^{m} \frac{1}{-t f_i(x)} \nabla f_i(x) + A^T \frac{w}{t} = 0, \quad Ax = b$$

Vanishing optimality gap for the original problem

- It follows that $x^*(t)$ is the minimizer of the Lagrangian of the original problem

$$L(x, \lambda^*(t), \nu^*(t)) = f_0(x) + \sum_{i=1}^{m} \lambda_i^*(t) f_i(x) + \nu^*(t)^T (Ax - b)$$

  for $\lambda_i^*(t) = \frac{1}{-t f_i(x^*(t))}$ and $\nu^*(t) = \frac{w}{t}$
- $x^*(t)$ is primal feasible: $Ax^*(t) = b$ and $f_i(x^*(t)) < 0$
- $\lambda_i^*(t) = \frac{1}{-t f_i(x^*(t))} > 0$ is dual feasible and

$$L(x^*(t), \lambda^*(t), \nu^*(t)) = f_0(x^*(t)) + \sum_{i=1}^{m} \frac{1}{-t} + \nu^*(t) \cdot 0 = f_0(x^*(t)) - \frac{m}{t}$$

- We have that

$$f^* \geq g^*(\lambda^*(t), \nu^*(t)) = L(x^*(t), \lambda^*(t), \nu^*(t)) = f_0(x^*(t)) - \frac{m}{t}$$

- It follows from $f^* \geq f_0(x^*(t)) - \frac{m}{t} \geq f^* - \frac{m}{t}$ that $f_0(x^*(t)) \to f^*$ as $t \to \infty$

Approximate KKT conditions of the original problem

$x^*(t), \lambda^*(t), \nu^*(t)$ satisfy

1. Primal feasibility: $f_i(x_i) < 0$, $i = 1, \ldots, m$; $Ax = b$
2. Dual feasibility: $\lambda \succ 0$
3. Lagrangian optimality $\nabla f_0(x) + \sum_{i=1}^m \lambda_i \nabla f_i(x) + A^T \nu = 0$
4. Complementary slackness $\lambda_i f_i(x) = 1/(-t)$

where $x^*(t)$ is the solution of the centering problem; $\lambda_i^*(t) = \frac{1}{-t f_i(x^*(t))}$; $\nu^*(t) = \frac{w}{t}$; and w is the Lagrange multiplier of the centering problem

- $x^*(t), \lambda^*(t), \nu^*(t)$ *approximately* satisfy the KKT conditions of the original problem
- The precise complementary slackness is $\lambda_i f_i(x) = 0$ but now $\lambda_i f_i(x) = 1/(-t)$
- The condition is more accurately satisfied as t increases

Barrier method

Barrier method:

Require: Strictly feasible x; $t > 0$; $\mu > 1$; tolerance $\epsilon > 0$

Centering step: Compute $x^*(t)$ as solution to $\min f_0(x) + t\phi(x)$ subject to $Ax = b$

Update: $x = x^*(t)$

while $m/t \geq \epsilon$ **do**

Increase: $t := \mu t$

Centering step: Compute $x^*(t)$ as solution to $\min f_0(x) + t\phi(x)$ subject to $Ax = b$

Update: $x = x^*(t)$

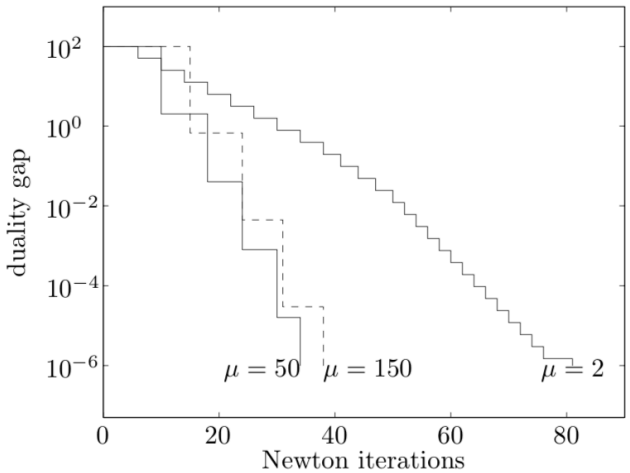
end while

- Convergence criterion guarantees that $f_0(x) - f^* \leq m/t \leq \epsilon$
- Centering step solved with Newton's method for convex problems with equality constraints (described earlier) initialized at x from previous centering step
- Larger μ implies fewer outer iterations but more inner (Newton) iterations

Example: LP

- Inequality LP with $A \in \mathbb{R}^{100 \times 50}$

$$\begin{aligned} \min \quad & c^T x \\ \text{subj. to} \quad & Ax \preceq b \end{aligned}$$



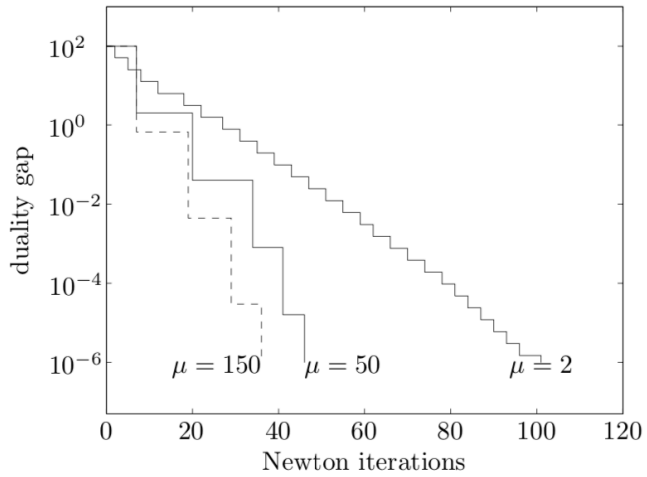
Boyd & Vandenberghe, *Convex Opt.*

- Initial x has duality gap 100; termination when duality gap is 10^{-6}
- Newton's method with backtracking used to solve the centering problem
- Horizontal axis shows the cumulative number of inner (Newton) iterations
- Staircase plot where each stair corresponds to one outer iteration
- Horizontal portion shows the Netwon iterations
- Vertical portion is the duality gap reduction at the end of each outer iteration by a factor of μ
- Total number of steps approximately the same for $\mu = 50$ and $\mu = 150$

Example: Geometric program

- Geometric program with $n = 50$ variables and $m = 100$ constraints

$$\begin{aligned} \text{min} \quad & \log\left(\sum_{k=1}^{5} e^{a_{0k}^T x + b_{0k}}\right) \\ \text{subj. to} \quad & \log\left(\sum_{k=1}^{5} e^{a_{ik}^T x + b_{ik}}\right), i = 1, \ldots, m \end{aligned}$$



- Randomly generated data
- Convergence behavior similar to LP

Phase I method

- The barrier method requires an initial x that is *strictly feasible*
- Such x is computed by a phase I method
- Feasibility problem

$$\text{Find } x \text{ such that } \quad f_i(x) \leq 0,\ i = 1, \ldots, m, \quad Ax = b \tag{F}$$

- Problem with variables $x \in \mathbb{R}^n$ and $s \in \mathbb{R}$ to minimize the maximum infeasibility

$$\begin{array}{ll} \min & s \\ \text{subj. to} & f_i(x) \leq s,\ i = 1, \ldots, m \\ & Ax = b \end{array} \tag{MI}$$

- This problem is always strictly feasible: Select any x in the domain of the problem such that $Ax = b$; and then set s to any value so that $s > f_i(x)$, $i = 1 \ldots, m$
- Problem (MI) can be solved with the barrier method

Feasibility using phase I method

- Problem (MI) wants to make s as small as possible
- Let s^* the optimal value of (MI)
- If $s^* < 0$, then the resulting x from the the solution of (MI) is strictly feasible for (F) and can be used as initialization for the main barrier method
- If $s^* > 0$, then there is no x to satisfy (F); we have a guarantee that the original problem is infeasible
- The case $s^* = 0$ can be analyzed theoretically but in practice, we cannot get exactly 0 as optimal value; the phase I method will exit with $|s^*| < \varepsilon$, where $\varepsilon > 0$ is small
- See textbook for a more refined phase I method that minimizes the sum of infeasibilities

Interior point methods

- Every iterate of the barrier method remains in the *interior* of the feasible set
- Primal-dual interior-point methods are similar to the barrier method and are widely used in optimization solvers for particular classes of convex problems (LP, QP, etc.)
- Primal-dual interior point methods update the primal and dual variables at each step (no distinction between outer and inner iteration)
- There are also variations for nonconvex problems
- For example, Matlab is using an interior point method with a logarithmic barrier as one of the algorithms for general purpose constrained optimization in `fmincon` `https://www.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html` (scroll down to the "interior point algorithm")