Optimization Methods
Lecture 3

Solmaz S. Kia
Mechanical and Aerospace Engineering Dept.
University of California Irvine
solmaz@uci.edu

Parts to consult in Ref[2]: 8.1-8.3 and 8.5.
Parts to consult in Ref[1]: pages 22-33 and Appendix C

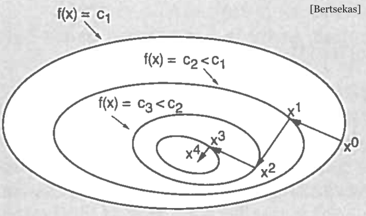
Numerical solvers for unconstrained optimization

$$x^\star = \underset{x \in \mathbb{R}^n}{\text{argmin}} \; f(x)$$

Iterative descent methods

- start from $x_0 \in \mathbb{R}^n$ (initial guess)
- successively generate vectors $x_1, x_2, \cdots$ such that

$$f(x_{k+1}) < f(x_k), \qquad k = 0, 1, 2, \cdots$$



$$x_{k+1} = x_k + \alpha_k \, d_k$$

<u>Design factors in iterative descent algorithms</u>:

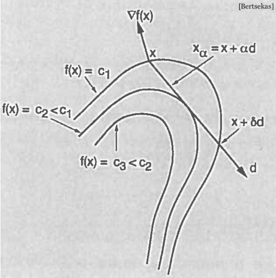
- what direction to move: descent direction
- how far move in that direction: step size

Successive descent method

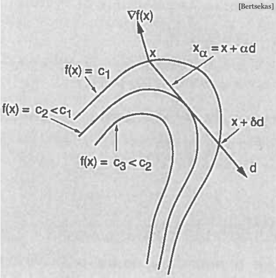
$$x_{k+1} = x_k + \alpha_k d_k \text{ such that } f(x_{k+1}) \leqslant f(x_k)$$

Descent direction design

$$\left.\begin{array}{l} f(x_{k+1}) = f(x_k + \alpha_k d_k) \approx f(x_k) + \alpha \nabla f(x_k)^\top d_k \\ \text{Requirement}: \ f(x_{k+1}) \leqslant f(x_k) \end{array}\right\} \Rightarrow \nabla f(x_k)^\top d_k < 0$$



$x_{k+1} = x_k + \alpha_k\, d_k$ such that $f(x_{k+1}) \leqslant f(x_k)$



Step-size design

$$\left.\begin{array}{l} \text{given } d_k \text{ that satisfies } \nabla f(x_k)^\top d_k < 0 \\ \text{Requirement: } f(x_{k+1}) \leqslant f(x_k) \\ \text{let } g(\alpha) = f(x_k + \alpha_k\, d_k) \end{array}\right\} \Rightarrow \alpha_k = \underset{\alpha>0}{\operatorname{argmin}}\ g(\alpha),$$

- There always exists $\alpha > 0$ such that $f(x_k + \alpha d_k) \leqslant f(x_k)$ because

$$g'(\alpha) = \frac{\partial g(\alpha)}{\partial \alpha} = \nabla f(x_k + \alpha\, d_k)^\top . d_k \Rightarrow g'(0) = \nabla f(x_k)^\top . d_k < 0$$

(Note that $g(\alpha) = f(x_k + \alpha d_k)$ and $g(0) = f(x_k)$)

Successive descent method

$$x_{k+1} = x_k + \alpha_k d_k, \text{ such that } f(x_{k+1}) \leqslant f(x_k)$$

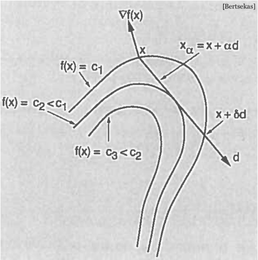
- **Decent direction**: $\nabla f(x_k)^\top d_k < 0$

General paradigm for the descent algorithms:

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

Lemma: Consider any positive definite matrix $B \in \mathbb{R}^{n \times n}$. For any point $x \in \mathbb{R}^n$ with $\nabla f(x) \neq 0$, the direction of $d = -B\nabla f(x)$ is a descent direction, i.e., $\nabla f(x)^\top d < 0$.

Proof: We have $(\nabla f(x))^\top(-B\nabla f(x)) = -\nabla f(x)^\top B \nabla f(x) < 0$ by assumption that $B > 0$.



Common choices of descent direction

$$x_{k+1} = x_k - \alpha_k B_k \nabla f(x_k), \quad B_k > 0$$

- **Steepest descent algorithm**: $B_k = I$
  Simplest descent direction but not always the fastest
- **Newtown's method**: $B_k = (\nabla^2 f(x_k))^{-1}$, $\alpha_k = 1$ (under the assumption that $\nabla^2 f(x) > 0$)
  Computationally expensive, but can have much faster convergence
- **Diagonally Scaled Steepest Descent**: $B_k = \begin{bmatrix} d_{1,k} & 0 & \cdots & 0 \\ 0 & d_{2,k} & \cdots & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & d_{n,k} \end{bmatrix}$,
  $d_{i,k} > 0$.
  For example $d^{i,k} = (\frac{\partial^2 f(x_k)}{(\partial x_i)^2})^{-1}$ (diagonally approximates Newton direction)
- **Modified Newton directions**
  - $B_k = (\nabla^2 f(x_0))^{-1}$ (computationally cheaper)
  - $B_k = (\nabla^2 f(x_k) + \gamma_k I)^{-1}$, $\gamma_k > |\lambda_{\min}(\nabla^2 f(x_k))|$ (to guarantee positive definiteness)
- **Quasi Newton directions**: will be discussed later

Question:Wether each limit point of a sequence $\{x_k\}$ generated by gradient successive decent algorithms is a stationary point.[1]

Observation: If d_k asymptotically becomes orthogonal to the gradient direction, $\frac{\nabla f(x_k)^\top d_k}{\|\nabla f(x_k)\|\|d_k\|} \to 0$ as x_k approaches a non-stationary point, there is a chance that the method will get "stuck" near that point.

A measure to avoid getting stuck: Let $d_k = -B_k \nabla f(x_k)$, $B_k > 0$.

If eigenvalues of B_k are bounded above and bounded away from zero:

$$\exists\ c_1, c_2 > 0 \text{ such that } c_1\|z\|^2 \leqslant z^\top B_k z \leqslant c_2\|z\|^2, \quad \forall z \in \mathbb{R}^n, k \in \mathbb{Z}_{\geqslant 0}$$

Then
$$\|\nabla f(x_k)^\top d_k\| = \|\nabla f(x_k)^\top B_k \nabla f(x_k)\| \geqslant c_1\|\nabla f(x_k)\|^2,$$
$$\begin{cases} \|d_k\|^2 = \|\nabla f(x_k)(B_k)^2 \nabla f(x_k)\| c_2^2 \|\nabla f(x_k)\| \leqslant c_2^2\|\nabla f(X_k)\|^2 \Rightarrow \\ c_1^2\|\nabla f(X_k)\|^2 \leqslant \|d_k\|^2 \leqslant c_2^2\|\nabla f(X_k)\|^2 \end{cases}$$

As long as $\nabla f(x_k)$ does not tend to zero, d_k cannot become asymptotically orthogonal to $\nabla f(x_k)$.

[1] We say $x \in \mathbb{R}^n$ is a **limit point of a sequence** $\{x_k\}$, if there exists a subsequence of $\{x_k\}$ that converges to x.

Theorem

Consider the sequence $\{x_k\}$ *generated by any decent algorithm with*

$$d_k = -B_k \nabla f(x_k), \quad B_k > 0$$

such that $\exists\, c_1, c_2 > 0$ *such that* $c_1\|z\|^2 \leqslant z^\top B_k z \leqslant c_2\|z\|^2, \quad \forall z \in \mathbb{R}^n, k \in \mathbb{Z}_{\geqslant 0}$, *and step size is chosen according to the minimization rule, or the limited minimization rule, (or the Armijo rule). Then, every limit point of* $\{x_k\}$ *is a stationary point.*

Note: A stationary point may not be a local minimum point. It may be a saddle point or even a local maximum. There is a result called the "capture theorem" (see Ref[1]), which informally states that isolated local minima tends to attract gradient methods.

Question to think about: How would you check if the point that you endup with (assuming it is stationary) is actually a local minimum?

$$x_{k+1} = x_k - \alpha_k B_k \nabla f(x_k), \quad B_k > 0$$

- **Exact line search**: $\alpha_k = \underset{\alpha \geqslant 0}{\operatorname{argmin}} f(x_k + \alpha d_k)$
  - A minimization problem itself, but an easier one (one dimensional).
  - If f convex, the one dimensional minimization problem also convex (why?).
- **Limited minimization**: $\alpha_k = \underset{\alpha \in [0,s]}{\operatorname{argmin}} f(x_k + \alpha d_k)$

  (tries not to stop too far)
- **Constant stepsize**: $\alpha_k = s > 0$ for all k
  (simple rule but may not converge if it is too large or may converge too slow because it is too small)
- **Diminishing step size**: $\alpha_k \to 0$, and $\sum_{k=1}^{\infty} \alpha_k = \infty$. For example $\alpha_k = \frac{1}{k}$
  - Descent not guaranteed at each step; only later when becomes small.
  - $\sum_{k=1}^{\infty} \alpha_k = \infty$ imposed to guarantee progress does not become too slow.
  - Good theoretical guarantees, but unless the right sequence is chosen, can also be a slow method.
- **Successive step size reduction**: well-known examples are Armijo rule (also called Backtracking) and Goldstein rule
  (search but not minimization)

Limited minimization: $\alpha_k = \underset{\alpha \in [0,s]}{\mathrm{argmin}} f(x_k + \alpha d_k)$

- Assumption: $g(\alpha)$ is unimodal over $[0, s]$

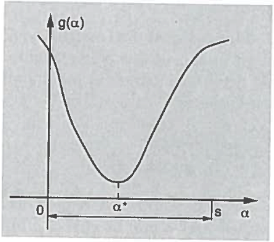


Image credit [Bertsekas]

Figure C.2. A strictly unimodal function g over an interval $[0, s]$ is defined as a function that has a unique global minimum α^* in $[0, s]$ and if α_1, α_2 are two points in $[0, s]$ such that $\alpha_1 < \alpha_2 < \alpha^*$ or $\alpha^* < \alpha_1 < \alpha_2$, then

$$g(\alpha_1) > g(\alpha_2) > g(\alpha^*)$$

or

$$g(\alpha^*) < g(\alpha_1) < g(\alpha_2),$$

respectively. An example of a strictly unimodal function, is a function which is strictly convex over $[0, s]$.

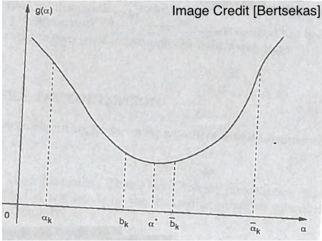
minimize g over $[0, s]$ by determining at the kth iteration an interval $[\alpha_k, \bar{\alpha}_k]$ containing $\alpha^\star$.

Solutions we explore

- Golden Section method
- Quadratic fit method

Stepsize selection via limited minimization: Golden Section method

Given $[\alpha_k, \bar{\alpha}_k]$, determine $[\alpha_{k+1}, \bar{\alpha}_{k+1}]$ such that $\alpha^\star \in [\alpha_{k+1}, \bar{\alpha}_{k+1}]$.



- **Initialization**: $[\alpha_0, \bar{\alpha}_0] = [0, s]$
- **Step k**:

$$\begin{cases} b_k = \alpha_k + \tau(\bar{\alpha}_k - \alpha_k), \\ \bar{b}_k = \alpha_k - \tau(\bar{\alpha}_k - \alpha_k), \end{cases}$$

- compute $g(b_k)$ and $g(\bar{b}_k)$

(1) If $g(b_k) < g(\bar{b}_k)$: $\begin{cases} \alpha_{k+1} = \alpha_k, \quad \bar{\alpha}_{k+1} = b_k & \text{if } g(\alpha_k) \leqslant g(b_k) \\ \alpha_{k+1} = \alpha_k, \quad \bar{\alpha}_{k+1} = \bar{b}_k & \text{if } g(\alpha_k) > g(b_k) \end{cases}$

(2) If $g(b_k) > g(\bar{b}_k)$: $\begin{cases} \alpha_{k+1} = \bar{b}_k, \quad \bar{\alpha}_{k+1} = \bar{\alpha}_k & \text{if } g(\bar{b}_k) \geqslant g(\bar{\alpha}_k) \\ \alpha_{k+1} = b_k, \quad \bar{\alpha}_{k+1} = \bar{\alpha}_k & \text{if } g(\bar{b}_k) < g(\bar{\alpha}_k) \end{cases}$

(3) If $g(b_k) = g(\bar{b}_k)$: $\alpha_{k+1} = b_k, \quad \bar{\alpha}_{k+1} = \bar{b}_k.$

- **Stop**: If $(\bar{\alpha}_k - \alpha_k) < \epsilon$

- The intervals are obtained using $\tau = \frac{3-\sqrt{5}}{2} \approx 0.381966011250105$
- τ satisfies $\tau = (1-\tau)^2$ (see page 746 of Ref[1])
- related to Fibonacci number sequence

Strictly unimodal g: the interval $[\alpha_k, \bar{\alpha}_k]$ contains $\alpha^\star$ and $(\bar{\alpha}_k - \alpha_k) \to 0$

Stepsize selection via limited minimization: quadratic fit

Quadratic fit: start with $[\alpha_1, \alpha_2, \alpha_3]$ that brackets the minimum
Step 1:fit a quadratic curve to $\{\alpha_1, \alpha_2, \alpha_3\}$:

$$q(\alpha) = g(\alpha_1)\frac{(\alpha-\alpha_2)(\alpha-\alpha_3)}{(\alpha_1-\alpha_2)(\alpha_1-\alpha_3)} + g(\alpha_2)\frac{(\alpha-\alpha_1)(\alpha-\alpha_3)}{(\alpha_2-\alpha_1)(\alpha_2-\alpha_3)} + g(\alpha_3)\frac{(\alpha-\alpha_1)(\alpha-\alpha_2)}{(\alpha_3-\alpha_1)(\alpha_3-\alpha_2)}$$

Step 2: Find the minimum point of this quadratic curve, which is

$$\alpha_4 = \frac{1}{2}\frac{b_{23}g(\alpha_1)+b_{31}g(\alpha_2)+b_{12}g(\alpha_3)}{a_{23}g(\alpha_1)+a_{31}g(\alpha_2)+a_{12}g(\alpha_3)}, \quad a_{ij} = \alpha_i - \alpha_j, \; b_{ij} = \alpha_i^2 - \alpha_j^2.$$

Step 3: $$[\alpha_1, \alpha_2, \alpha_3]_{\textbf{new}} = \begin{cases} [\alpha_1, \alpha_2, \alpha_4] & \text{if } \alpha_4 > \alpha_2 \text{ and } g(\alpha_4) \geqslant g(\alpha_2), \\ [\alpha_2, \alpha_4, \alpha_3] & \text{if } \alpha_4 > \alpha_2 \text{ and } g(\alpha_4) < g(\alpha_2), \\ [\alpha_4, \alpha_2, \alpha_3] & \text{if } \alpha_4 < \alpha_2 \text{ and } g(\alpha_4) \geqslant g(\alpha_2), \\ [\alpha_1, \alpha_4, \alpha_2] & \text{if } \alpha_4 < \alpha_2 \text{ and } g(\alpha_4) < g(\alpha_2), \end{cases}$$

Step 4: Check if $|\alpha_3 - \alpha_1| < \epsilon$ if satisfied stop and take α_2 as minimum point otherwise go to step 1 and repeat the process.

Quadratic fit has a super-linear convergence (order of $p = 1.2$) and converges faster than Golden-section method.

Safeguarding the process: the point α_4 must not be too close to any of the three existing points else the subsequence fit will be ill-conditioned. This is especially needed when the polynomial fit algorithm is embedded in a n-variable routine (searching for stepsize in optimization algorithms). This is taken care of by defining a measure δ and moving or bumping the point away from the existing point by this amount. For example you can use the following routine:

$$\begin{cases} \text{if } |\alpha_4 - \alpha_1| < \delta, & \text{then set} \quad \alpha_4 = \alpha_1 + \delta \\ \text{if } |\alpha_4 - \alpha_3| < \delta, & \text{then set} \quad \alpha_4 = \alpha_3 - \delta \\ \text{if } |\alpha_4 - \alpha_2| < \delta, \text{ and } \alpha_2 > 0.5(\alpha_1+\alpha_3) & \text{then set} \quad \alpha_4 = \alpha_2 - \delta \\ \text{if } |\alpha_4 - \alpha_2| < \delta, \text{ and } \alpha_2 \leqslant 0.5(\alpha_1+\alpha_3) & \text{then set} \quad \alpha_4 = \alpha_2 + \delta \end{cases}$$