Basics of Unconstrained Optimization

Dr. José Ernesto Rayas-Sánchez

1

Outline

- Unconstrained optimization problem
- Recognizing a minimizer
- Stopping criteria
- Classification of unconstrained optimization methods
- Line search methods
- Trust region methods

Dr. J. E. Rayas-Sánchez 2

Unconstrained Optimization Problems

Formulation:

$$x^* = \arg\min_{x} u(x)$$

- $x \in \Re^n$
- $u: \Re^n \rightarrow \Re$

Recognizing a Local Minimizer

- x^s is a stationary point if $\nabla u(x^s) = \mathbf{0}$
- First order necessary condition:
  If x^* is a local minimizer then $\nabla u(x^*) = \mathbf{0}$
- Second order sufficient conditions:
  If $\nabla u(x^*) = \mathbf{0}$ and $\mathbf{H}(u(x^*))$ is positive definite then x^* is a strict local minimizer
- If a point is a stationary point and not a local minimizer or maximizer, the point is called a "saddle point"

Stopping Criteria

- A maximum number of iterations has been reached

$$i > i_{\max}$$

- The objective function is practically not decreasing

$$u(\boldsymbol{x}_i) - u(\boldsymbol{x}_{i+1}) < \varepsilon_1$$

- The absolute change in the optimization variables is small enough

$$\| \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \|_2 < \varepsilon_2$$

Stopping Criteria (cont)

- The relative change in the optimization variables is small enough

$$\| \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \|_2 < \varepsilon_3 (\| \boldsymbol{x}_i \|_2 + \varepsilon_4)$$

- The gradient is small enough

$$\| \nabla u(\boldsymbol{x}_i) \|_2 < \varepsilon_5$$

- The Hessian is positive definite near the solution ($\boldsymbol{x}_i \approx \boldsymbol{x}^*$, combined with previous criterion; usually not necessary)

$$(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i)^T \boldsymbol{H}(u(\boldsymbol{x}_i))(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i) > 0$$

Optimization Methods: A Broad Classification

- Non-descent methods
- Descent methods,
  $u(\boldsymbol{x}_{i+1}) < u(\boldsymbol{x}_i)$ for every i, or
  $u(\boldsymbol{x}_{i+1}) < u(\boldsymbol{x}_i)$ for $i > N$, where N is a number of initial steps
- Two fundamental strategies:
  - Line search methods
  - Trust region methods

Line Search Methods

- At the i-th iteration, the algorithm chooses a direction $\boldsymbol{d}_i$ and searches along this direction from the current iterate $\boldsymbol{x}_i$ for a new iterate $\boldsymbol{x}_{i+1}$ with a lower function value
- The search direction and the step size can be selected in several manners

A Generic Line Search Algorithm

begin
  $i = 0, \boldsymbol{x}_i = \boldsymbol{x}_0$
  repeat until *StoppingCriteria*
    $\boldsymbol{d}_i$ = SearchDirection($u, \boldsymbol{x}_i$)
    α_i = LineSearch($u, \boldsymbol{x}_i, \boldsymbol{d}_i$)
    $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{d}_i$
    $i = i + 1$
end

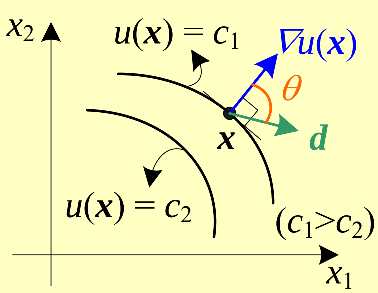
Search Directions

Any downhill direction $\pi/2 < \theta_i < 3\pi/2$

$$u(\boldsymbol{x}_i + \varepsilon \boldsymbol{d}_i) = u(\boldsymbol{x}_i) + \varepsilon \boldsymbol{d}_i^T \nabla u(\boldsymbol{x}_i) + O(\varepsilon^2)$$

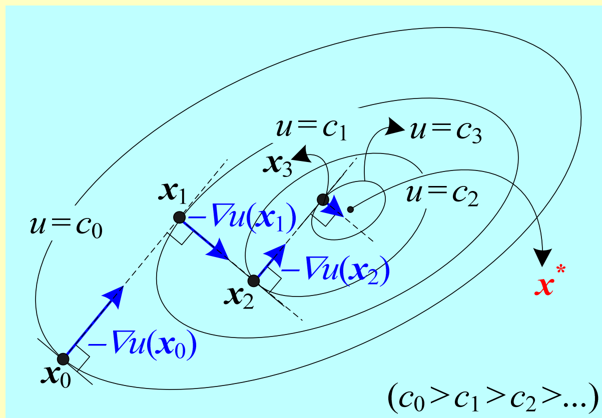
$\boldsymbol{d}_i^T \nabla u(\boldsymbol{x}_i) < 0$ → Condition for a downhill direction

$$\|\boldsymbol{d}_i\|_2 \, \|\nabla u(\boldsymbol{x}_i)\|_2 \cos\theta_i < 0$$



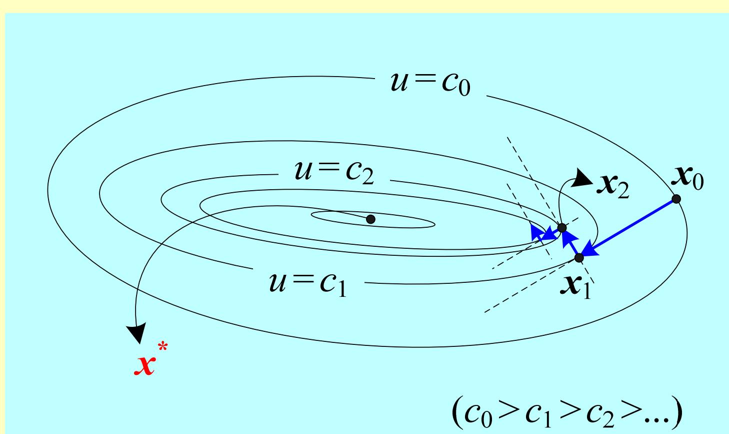
Steepest Descent Direction

- Steepest descent direction: $\boldsymbol{d}_i = -\nabla u(\boldsymbol{x}_i)$
- The steepest descent direction with exact line searches moves in orthogonal steps and it is globally convergent



Steepest Descent Direction (cont)

The steepest descent direction can be extremely slow (zigzagging)



Other Search Directions

- Newton direction

$$\boldsymbol{d}_i = -\boldsymbol{H}(u(\boldsymbol{x}_i))^{-1}\nabla u(\boldsymbol{x}_i)$$

in this case the most used step length is 1

$\boldsymbol{d}_i$ is a linear transformation of $-\nabla u(\boldsymbol{x}_i)$

- Quasi-Newton directions

$$\boldsymbol{d}_i = -\boldsymbol{B}_i^{-1}\nabla u(\boldsymbol{x}_i)$$

$\boldsymbol{B}_i$ is an approximation of $\boldsymbol{H}(\nabla u(\boldsymbol{x}_i))$ which is updated after each iteration to take into account the additional knowledge gained during the step

Other Search Directions (cont)

The two most popular updating formulas for $\boldsymbol{B}_i$ in Quasi-Newton directions:

- SR1 formula (Symmetric-Rank-One)

$$\boldsymbol{B}_{i+1} = \boldsymbol{B}_i + \frac{(\boldsymbol{y}_i - \boldsymbol{B}_i\boldsymbol{s}_i)(\boldsymbol{y}_i - \boldsymbol{B}_i\boldsymbol{s}_i)^T}{(\boldsymbol{y}_i - \boldsymbol{B}_i\boldsymbol{s}_i)^T\boldsymbol{s}_i}$$

- BFGS formula (Broyden, Fletcher, Goldfarb and Shanno)

$$\boldsymbol{B}_{i+1} = \boldsymbol{B}_i - \frac{\boldsymbol{B}_i\boldsymbol{s}_i\boldsymbol{s}_i^T\boldsymbol{B}_i}{\boldsymbol{s}_i^T\boldsymbol{B}_i\boldsymbol{s}_i} + \frac{\boldsymbol{y}_i\boldsymbol{y}_i^T}{\boldsymbol{y}_i^T\boldsymbol{s}_i}$$

where

$$\boldsymbol{s}_i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \qquad \boldsymbol{y}_i = \nabla u(\boldsymbol{x}_{i+1}) - \nabla u(\boldsymbol{x}_i)$$

Other Search Directions (cont)

Conjugate Gradient direction

$$\boldsymbol{d}_i = -\nabla u(\boldsymbol{x}_i) + \beta_i \boldsymbol{d}_{i-1}$$

where β_i is a scalar that ensures that $\boldsymbol{d}_i$ and $\boldsymbol{d}_{i-1}$ are conjugate

Any two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ are conjugate with respect to a symmetric positive definite matrix $\boldsymbol{A}$ if $\boldsymbol{a}^T\boldsymbol{A}\boldsymbol{b} = 0$

Finding the Step Size in Line Search Methods

- Exact line search

$$\alpha_i = \arg\min_{\alpha>0} u(\boldsymbol{x}_i + \alpha \boldsymbol{d}_i) = \arg\min_{\alpha>0} v(\alpha)$$

The exact line search stops at a point where the local gradient is orthogonal to the search direction

- Soft line search

(See: P.E. Frandsen, K. Jonasson, H.B. Nielsen and O. Tingleff, *Unconstrained Optimization*. Lyngby, Denmark: Department of Mathematical Modeling, Technical University of Denmark, 1999, pp. 26-30)

Trust Region Methods

- At the i-th iteration, a model m_i of the objective function u is created. The algorithm restricts the search for a minimizer of m_i to some region around $\boldsymbol{x}_i$. If the minimizer of m_i does not produce a sufficient decrease in u, the trust region is shrunk and the model is again minimized
- There are several ways to construct the model m_i, and to define the trust region (ball, elliptical, box-shaped, etc.)

A Generic Trust Region Algorithm

begin
  $i = 0$, $\boldsymbol{x}_i = \boldsymbol{x}_0$, $r = r_0 > 0$
  repeat until *StoppingCriteria*
    $m_i(\boldsymbol{s})$ = BuildModel(u, $\boldsymbol{x}_i$)
    $\boldsymbol{s}_i = \arg\min_{\boldsymbol{s}\in\Omega} m_i(\boldsymbol{s})$ where $\Omega = \{\boldsymbol{s} : \|\boldsymbol{s}\| \le r\}$
    $a = [u(\boldsymbol{x}_i) - u(\boldsymbol{x}_i + \boldsymbol{s}_i)]/[m_i(\boldsymbol{0}) - m_i(\boldsymbol{s}_i)]$
    if $a > 0.75$ **then** $r = 2r$
    if $a < 0.25$ **then** $r = r/3$
    if $a > 0$ **then** $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{s}_i$
    $i = i + 1$
end