ALGORITHMS AND DATA STRUCTURES II

Lecture 13

Optimization Algorithms

Lecturer: K. Markov

markov@u-aizu.ac.jp

FINAL EXAM

- **When:** August 7^{th} (Wednesday),
  1^{st} and 2^{nd} periods (9:00 – 10:40)
- **Where:** M5 (Markov group), M6 (Yen group)
- **Scope:** Lectures 7 to 12

- What you CAN use:
  - Lecture handouts from the course webpage,
  - Textbooks, dictionary, calculator.
- What you CANNOT use:
  - Exercise sheets, written notes, memos, etc.
  - Computers, smart-phones, cell-phones.

Decision Making Problems

- **Category 1:**
  - The set of possible alternatives for the decision is a finite discrete set typically consisting of a small number of elements.
  - Solution: scoring methods

- **Category 2:**
  - The number of possible alternatives is either infinite, or very large, and the decision may be required to satisfy some constraints.
  - Solution: unconstrained and constrained optimization methods

Category 2 Decision problems

1) Get a precise definition of the problem, all relevant data and information on it.
   - Controllable inputs (decision variables)
   - Uncontrollable factors (random variables)

2) Construct a mathematical (optimization) model of the problem.
   - Build objective functions and constraints.

1) Solve the model
   - Apply the most appropriate algorithms for the given problem.

Problem Specification

Suppose we have a cost function (or objective function)

$$f(\mathbf{x}) : \mathbb{R}^N \longrightarrow \mathbb{R}$$

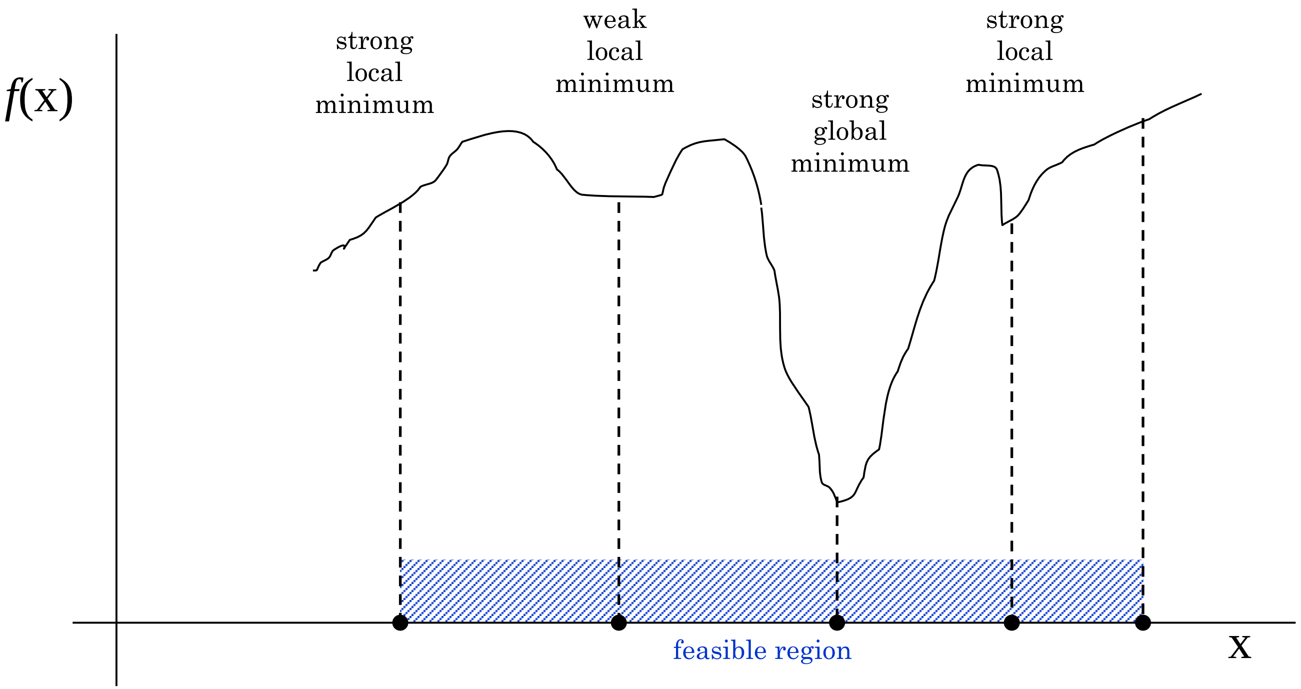
Our aim is to find values of the parameters (decision variables) $\mathbf{x}$ that minimize this function

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

Subject to the following constraints:

- Equality: $c_i(\mathbf{x}) = 0$
- Non-equality: $c_j(\mathbf{x}) \geq 0$

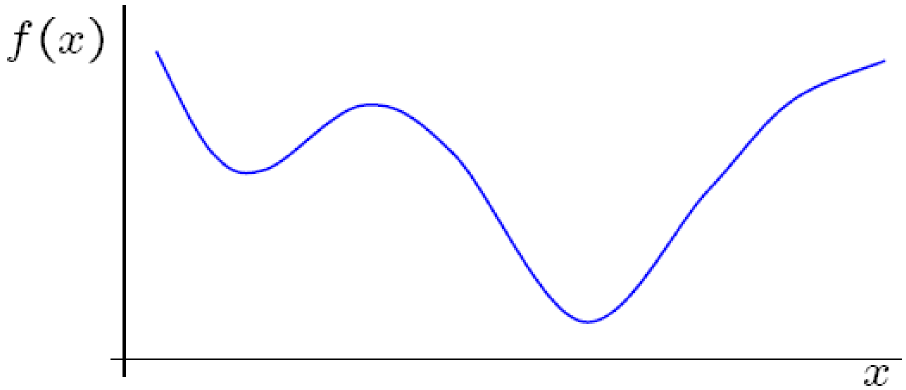
Types of minima



- Which of the minima is found depends on the starting point.
- Such minima often occur in real applications.

Unconstrained optimization

$$\min_{x} f(x)$$



How to determine the minimum?

- Search methods (Dichotomous, Fibonacci, Golden-Section)
- Approximation methods.
    1. Polynomial interpolation
    2. Newton method
- Combination of both.

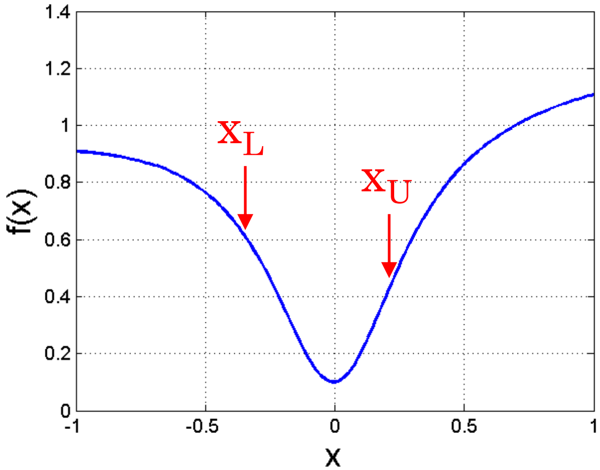
Search methods

General Algorithm

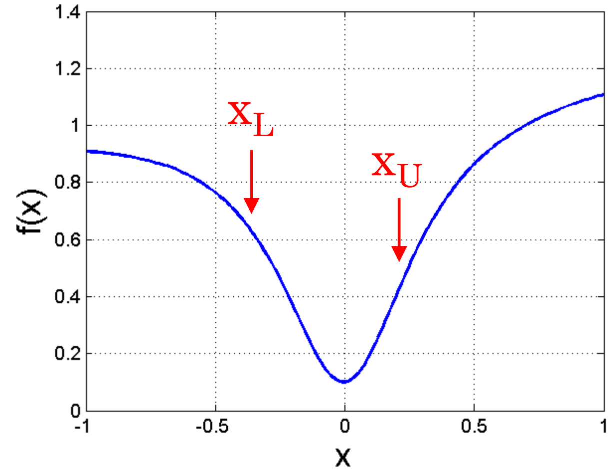
1. Start with the interval ("bracket") $[x_L, x_U]$ such that the minimum x^* lies inside.
2. Evaluate $f(x)$ at two point inside the bracket.
3. Reduce the bracket.
4. Repeat the process.

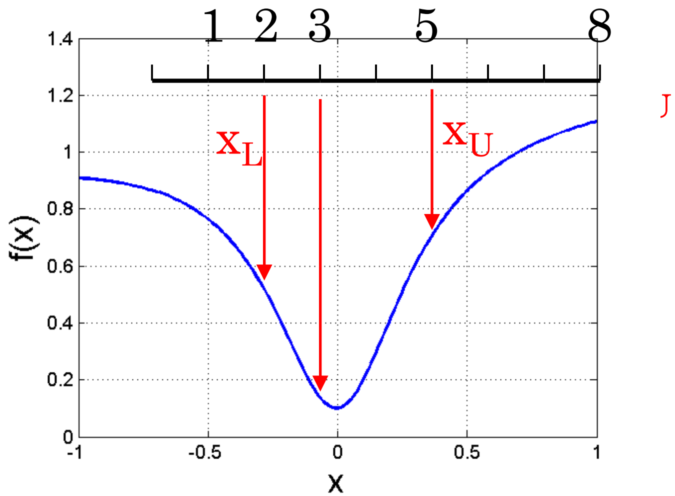
Can be applied to any function and differentiability is not essential.

Search methods



Dichotomous



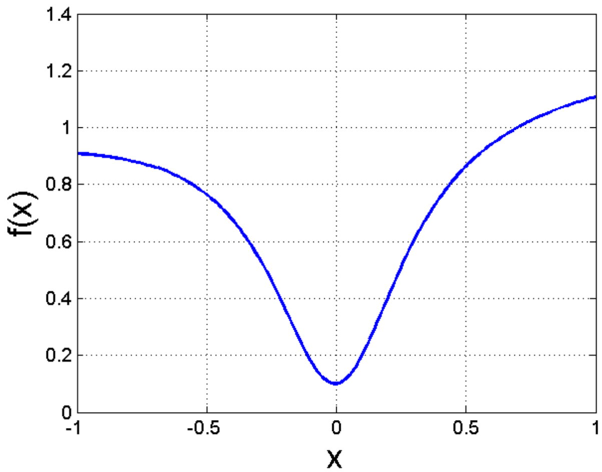


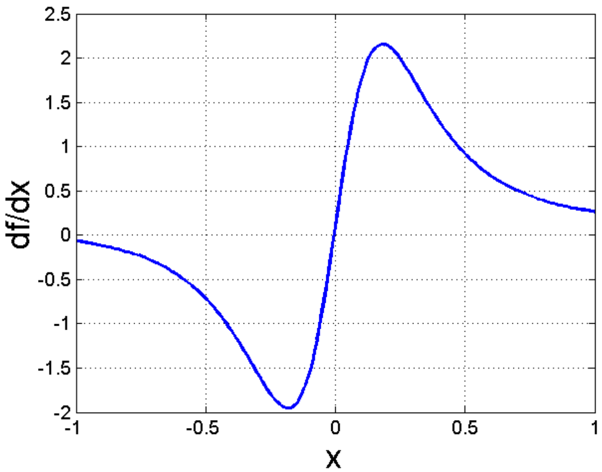
Fibonacci: 1 1 2 3 5 8 …

1D FUNCTIONS

As an example consider the function

$$f(x) = 0.1 + 0.1x + x^2/(0.1 + x^2)$$

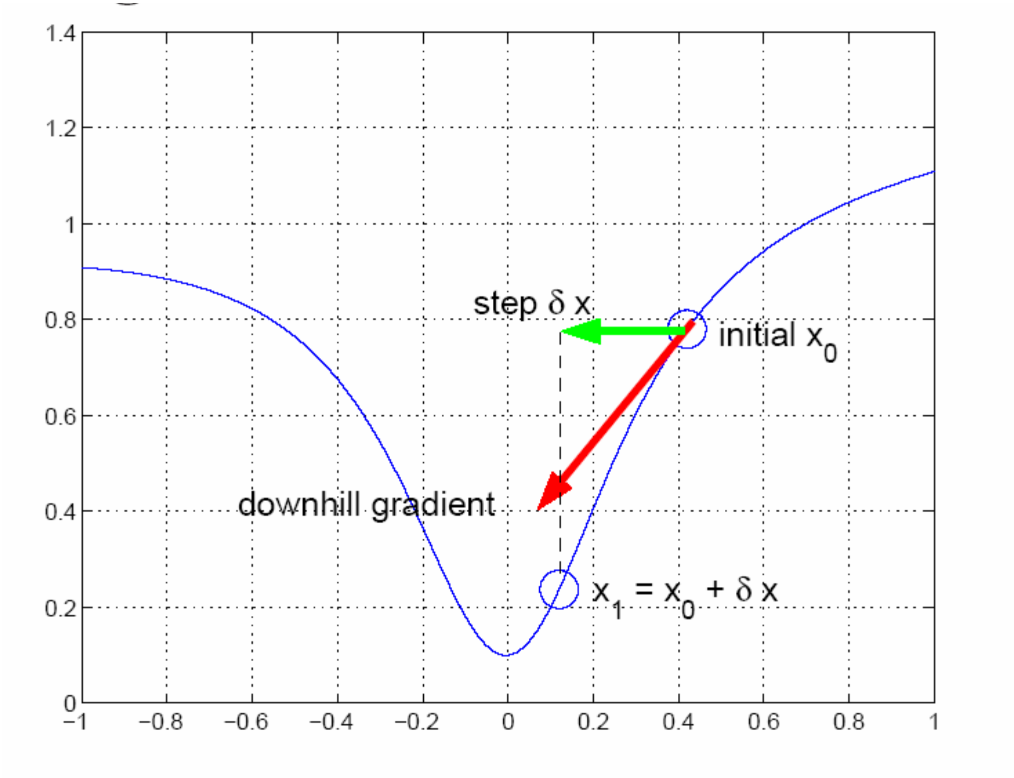




Assume we do not know the actual function expression from now on.

GRADIENT DESCENT

Given a starting location, x_0, examine $\partial f / \partial x$ and move in the downhill direction to generate a new estimate, $x_1 = x_0 + \delta x$



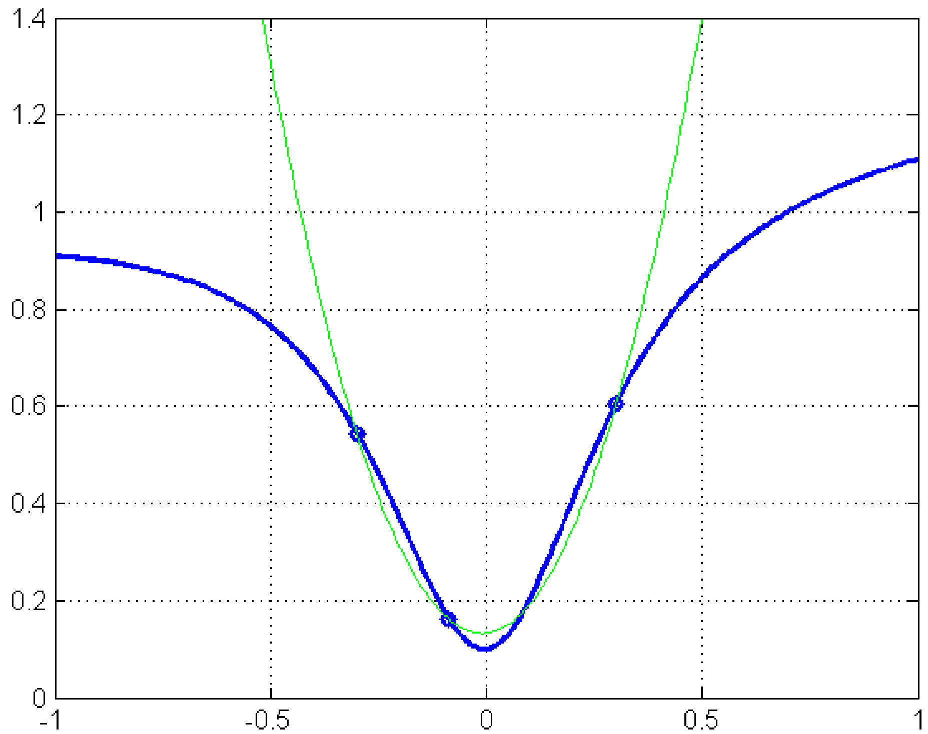
How to determine the step size δx ?

Polynomial interpolation

Algorithm:

1. Bracket the minimum.
2. Fit a quadratic or cubic polynomial which interpolates $f(x)$ at some points in the interval.
3. Jump to the (easily obtained) minimum of the polynomial.
4. Throw away the worst point and repeat the process.

POLYNOMIAL INTERPOLATION



- Quadratic interpolation using 3 points, 2 iterations
- Other methods to interpolate?
  - 2 points and one gradient
  - Cubic interpolation

Newton method

Fit a quadratic approximation to $f(x)$ using both gradient and curvature information at x.

- Expand $f(x)$ locally using a Taylor series:

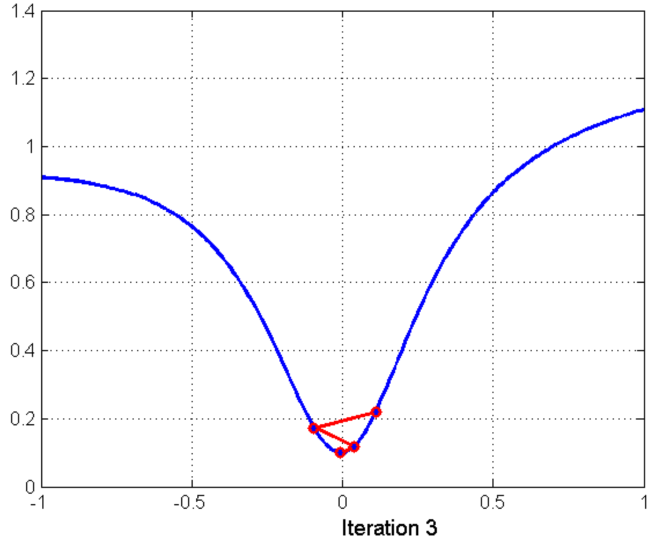
$$f(x+\delta x) = f(x) + f'(x)\delta x + \frac{1}{2}f''(x)\delta x^2 + o(\delta x^2)$$

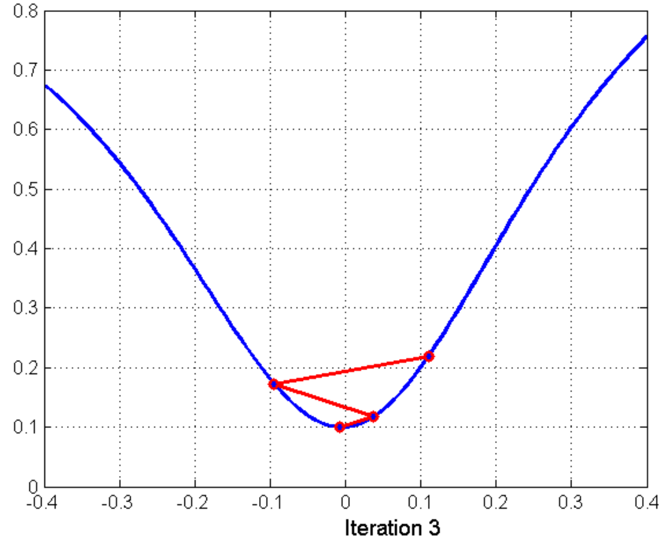
- Find the δx which minimizes this local quadratic approximation:

$$\delta x = -\frac{f'(x)}{f''(x)}$$

- Update x: $$x_{n+1} = x_n + \delta x = x_n - \frac{f'(x)}{f''(x)}$$

Newton method

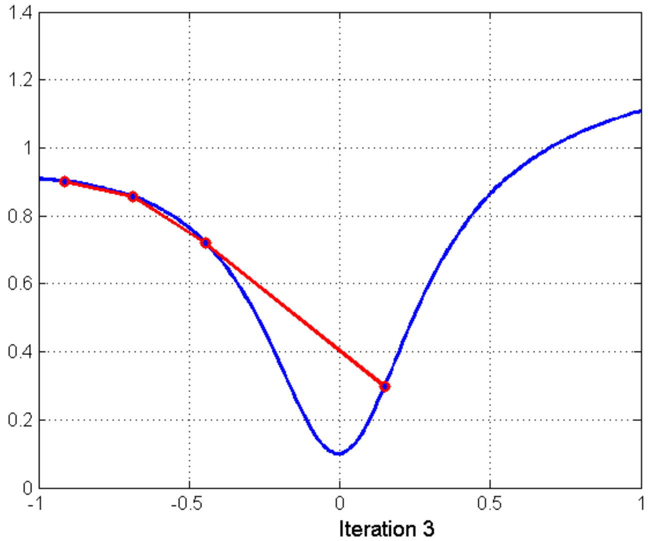


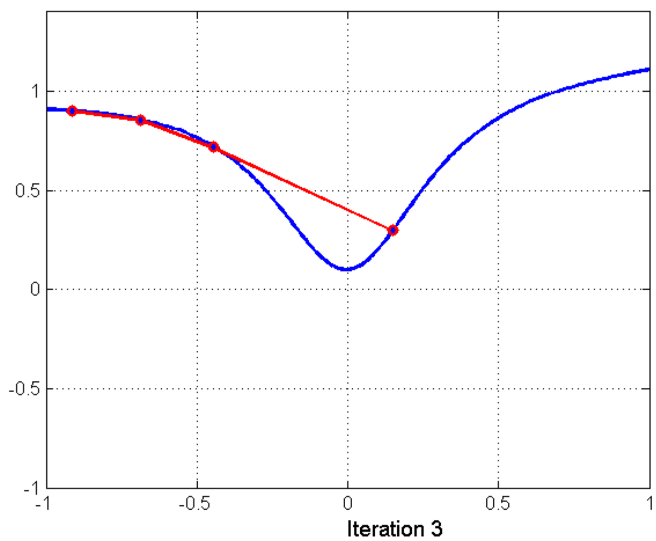


- Avoids the need to bracket the root.
- Quadratic convergence (decimal accuracy doubles at every iteration).

NEWTON METHOD

- Global convergence of Newton's method is poor.
- Often fails if the starting point is too far from the minimum.

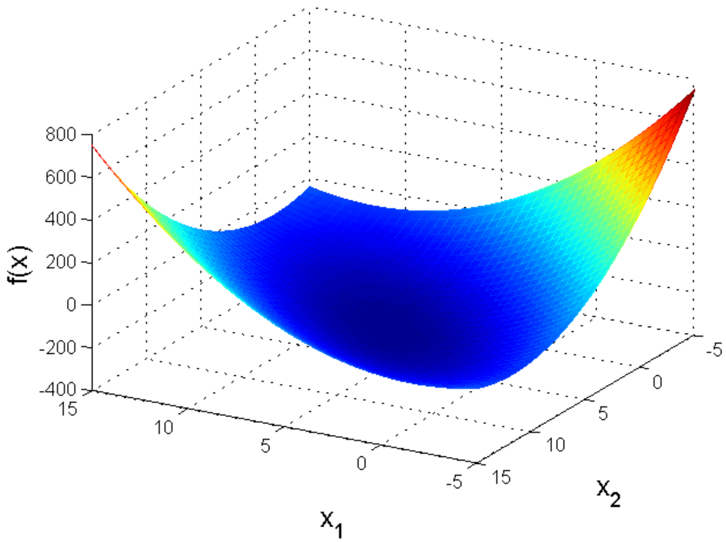


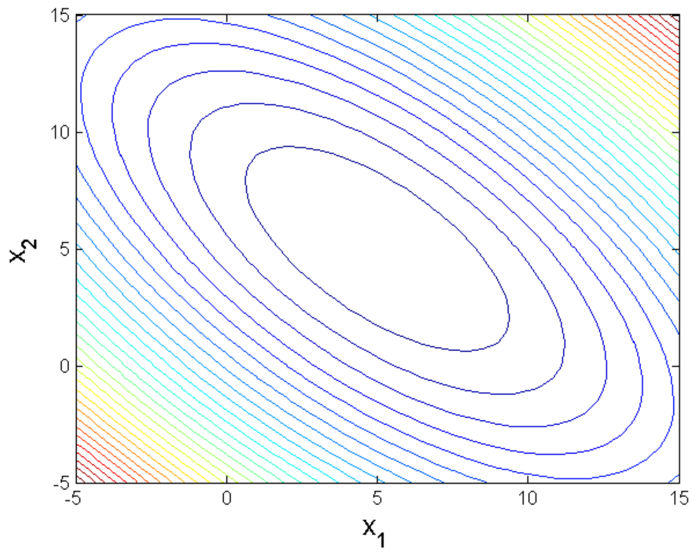


- In practice, must be used with a globalization strategy which reduces the step length until function decrease is assured.

Extension to N dimensions

- How big N can be?
  - Problem sizes can vary from a handful of parameters to many thousands.
- We will consider examples for $N = 2$, so that cost function surfaces can be visualized.





Generic Optimization Algorithm

1. Start at $\boldsymbol{x}_0, k = 0$.
2. Compute a search direction $\boldsymbol{p}_k$
3. Compute a step length α_k, such that $f(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k) < f(\boldsymbol{x}_k)$
4. Update: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$
5. Check for convergence (stopping criteria) e.g. $\frac{\partial f}{\partial \boldsymbol{x}} = 0$

Reduces optimization in N dimensions to a series of (1D) line minimizations

Taylor Expansion

A function may be approximated locally by its Taylor series expansion at point $\mathbf{x}^*$

$$\boxed{f(\mathbf{x}^* + \mathbf{x}) \approx f(\mathbf{x}^*) + \nabla f^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \mathbf{H} \mathbf{x}}$$

where the gradient $\nabla f(\mathbf{x}^*)$ is the vector

$$\nabla f(\mathbf{x}^*) = \left[\frac{\partial f}{x_1} \dots \frac{\partial f}{x_N}\right]^T$$

and the Hessian $\boldsymbol{H}(x^*)$ is the symmetric matrix

$$\mathbf{H}(\mathbf{x}^*) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N^2} \end{bmatrix}$$

Quadratic functions

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

- The vector $\boldsymbol{g}$ and the Hessian $\boldsymbol{H}$ are constant.
- Second order approximation of any function by the Taylor expansion is a quadratic function.

We will assume only quadratic functions for a while.

CONDITIONS FOR A MINIMUM

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

Expand $f(\boldsymbol{x})$ at a stationary point $\boldsymbol{x}^*$ in direction $\boldsymbol{p}$

$$\begin{aligned} f(\mathbf{x}^* + \alpha\mathbf{p}) &= f(\mathbf{x}^*) + \mathbf{g}(\mathbf{x}^*)^T\alpha\mathbf{p} + \frac{1}{2}\alpha^2\mathbf{p}^T\mathbf{H}\mathbf{p} \\ &= f(\mathbf{x}^*) + \frac{1}{2}\alpha^2\mathbf{p}^T\mathbf{H}\mathbf{p} \end{aligned}$$

since at the stationary point $\mathbf{g}(\mathbf{x}^*) = 0$

At a stationary point the behavior is determined by **H**.

Conditions for a minimum

- $\boldsymbol{H}$ is a symmetric matrix, and so has orthogonal eigenvectors:

$$\mathbf{H}\mathbf{u}_i = \lambda_i \mathbf{u}_i \qquad \|\mathbf{u}_i\| = 1$$

$$\begin{aligned} f(\mathbf{x}^* + \alpha \mathbf{u}_i) &= f(\mathbf{x}^*) + \frac{1}{2}\alpha^2 \mathbf{u}_i^T \mathbf{H}\mathbf{u}_i \\ &= f(\mathbf{x}^*) + \frac{1}{2}\alpha^2 \lambda_i \end{aligned}$$

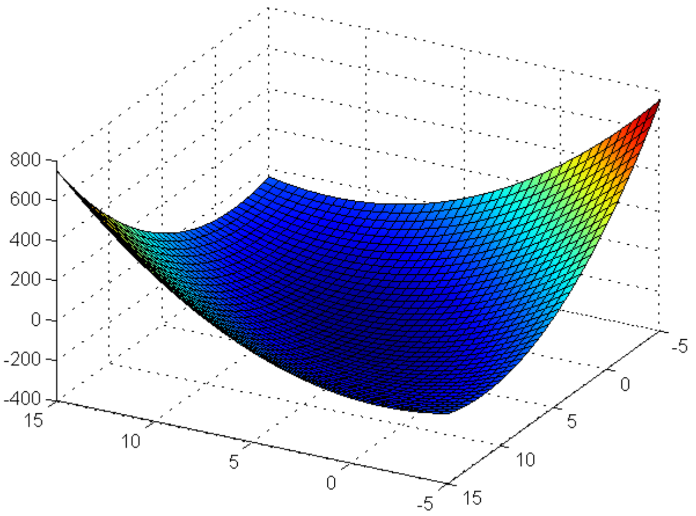
- As $|\alpha|$ increases, $f(\mathbf{x}^* + \alpha \boldsymbol{u}_i)$ increases, decreases or is unchanging according to whether λ_i is positive, negative or zero.

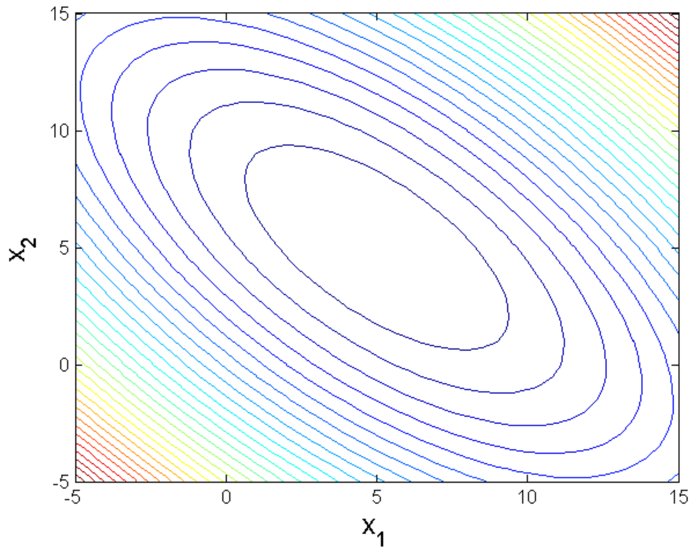
Examples of quadratic functions

Case 1: both eigenvalues positive

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

with $\quad a = 0\,, \qquad \mathbf{g} = \begin{bmatrix} -50 \\ -50 \end{bmatrix}\,, \qquad \mathbf{H} = \begin{bmatrix} 6 & 4 \\ 4 & 6 \end{bmatrix}$





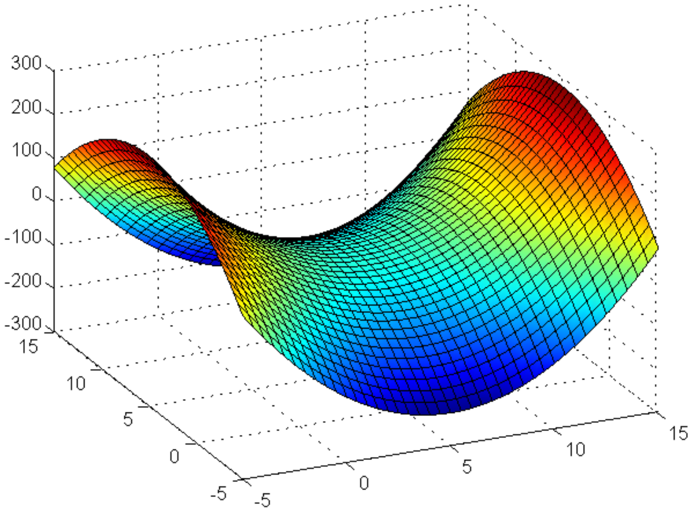
minimum

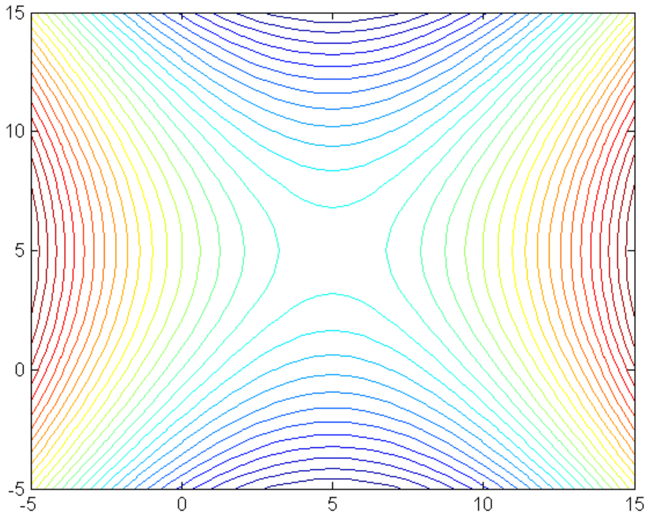
Examples of quadratic functions

Case 2: eigenvalues have different sign

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

with $a = 0\,,\qquad \mathbf{g} = \begin{bmatrix} -30 \\ 20 \end{bmatrix}\,,\qquad \mathbf{H} = \begin{bmatrix} 6 & 0 \\ 0 & -4 \end{bmatrix}$ indefinite





saddle point

Quadratic functions optimization

Assume that H is positive definite

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

$$\nabla f(\mathbf{x}) = \mathbf{g} + \mathbf{H}\mathbf{x}$$

There is a unique minimum at

$$\mathbf{x}^* = -\mathbf{H}^{-1}\mathbf{g}$$

If N is large, it is not feasible to perform this inversion directly.

Steepest descent

- Basic principle is to minimize the N-dimensional function by a series of 1D line-minimizations:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

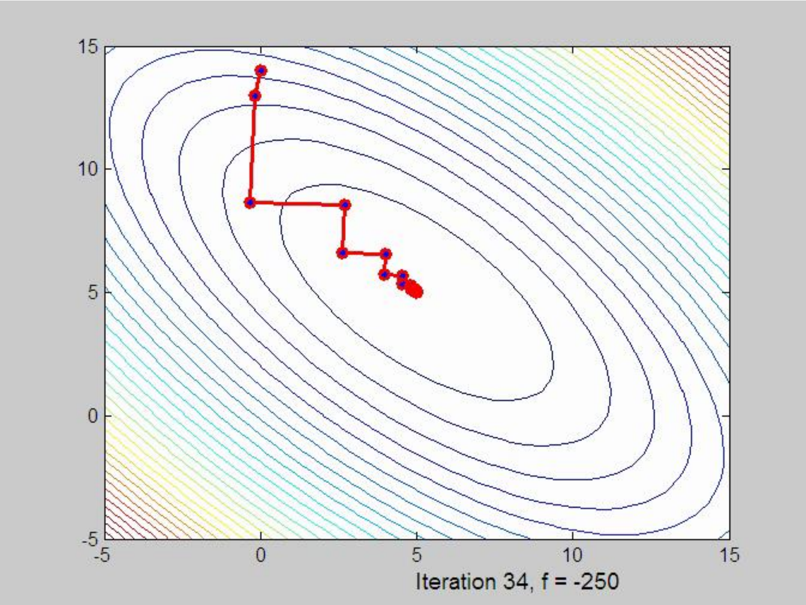
- The steepest descent method chooses $\boldsymbol{p}_k$ to be parallel to the gradient

$$\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$$

- Step-size α_k is chosen to minimize $f(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k)$. For quadratic forms there is a closed form solution:

$$\alpha_k = \frac{\mathbf{p}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{H} \mathbf{p}_k}$$

Steepest descent



- Everywhere, the gradient is perpendicular to the contour lines.
- After each line minimization the new gradient is always orthogonal to the previous step direction (true of any line minimization).
- Consequently, the iterates tend to zig-zag down the valley in a very inefficient manner.

Conjugate gradient

- Each $\boldsymbol{p}_k$ is chosen to be conjugate to all previous search directions with respect to the Hessian $\mathbf{H}$:

$$\mathbf{p}_i^T \mathbf{H} \mathbf{p}_j = 0 , \qquad i \neq j$$

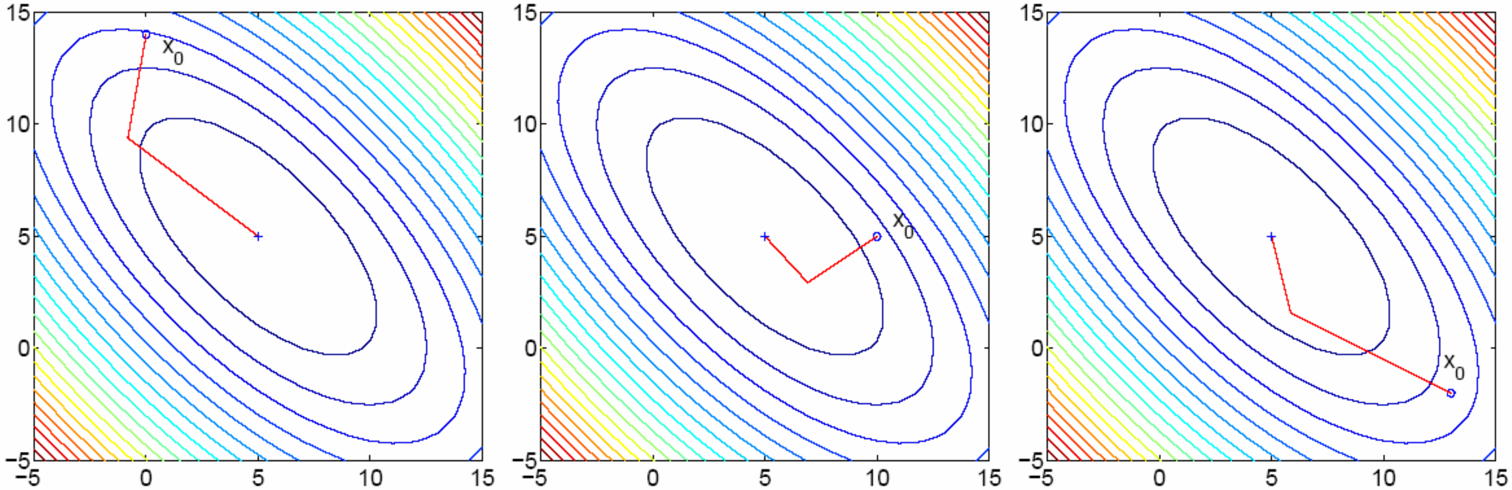
- The resulting search directions are mutually linearly independent.

- Remarkably, $\boldsymbol{p}_k$ can be chosen using only knowledge of $\boldsymbol{p}_{k-1}, \nabla f(\boldsymbol{x}_{k-1})$ and $\nabla f(\boldsymbol{x}_k)$:

$$\mathbf{p}_k = \nabla f_k + \left(\frac{\nabla f_k^\top \nabla f_k}{\nabla f_{k-1}^\top \nabla f_{k-1}} \right) \mathbf{p}_{k-1}$$

Conjugate gradient

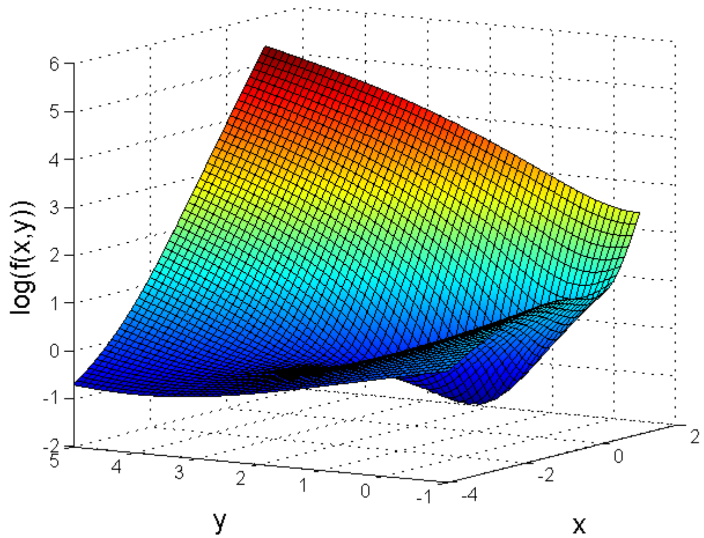
- An N-dimensional quadratic form can be minimized in at most N conjugate descent steps.

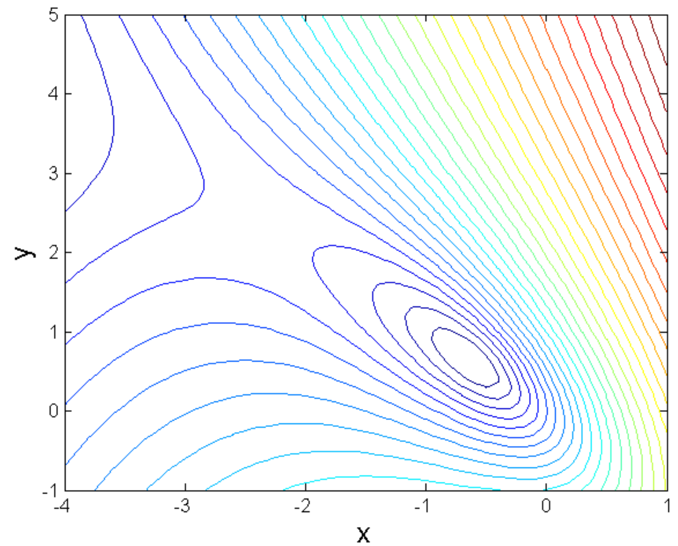


- 3 different starting points.
- Minimum is reached in exactly 2 steps.

General function Optimization

$$f(x, y) = \exp(x)(4x^2 + 2y^2 + 4xy + 2x + 1)$$





Apply methods developed using quadratic Taylor series expansion.

Summary

- Minimization of 1-D functions
  - Search methods
  - Approximation methods

- N-D functions -> finding the descent direction
- Taylor series -> Quadratic functions
- Newton method.
- Steepest descent.
- Conjugate Gradient.

That's All!