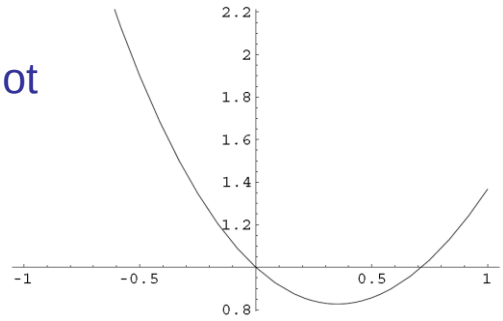
Table of Contents

Solving optimization problems

- What is an optimization problem?
- Examples of optimization problems
- Selected problem types and solution methods

A simple optimization problem

- The problem $\min_{x \in \Re} f(x) = e^{-x} + x^2$
- Has the following plot



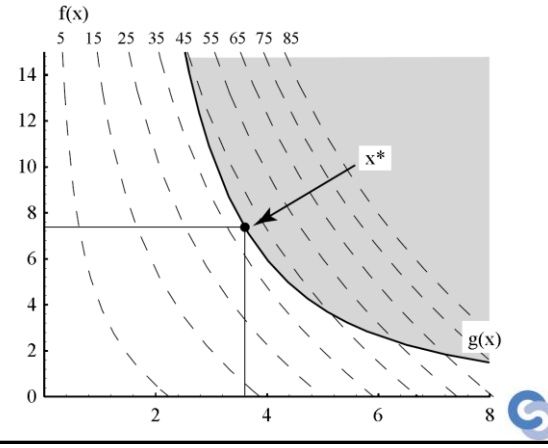
- And the solution: $f(x) = 0.827$ at $x = 0.352$

A 2D optimization problem

- How to manufacture a 0.3 l metal can with as little material as possible?
- The height of the can is *h*, its radius *r*, the volume $\pi \cdot r^2 h$ the surface area $2\pi \cdot r^2 + 2\pi \cdot rh$
- With some variable changes we get the optimization problem:
  - Objective function $\min_{x \in \Re^2} f(x) = x_1^2 + x_1 x_2,$
  - Constraint $g(x) = -x_1 x_2 + 300/\pi \le 0,$
  - Variables $x_i \ge 0,\ i = 1,2.$

A 2D optimization problem (2)

- The minimizing point x^* satisfies $f(x^*) \leq f(x)$ for all feasible points in $x \in \Re^n$
- The minimum $x^* \approx (3.6, 7.3)^T$
- Leads to the minimum function value $f(x^*) \approx 39$



A 2D optimization problem (3)

- The gradient of the objective function
$$f(x) = x_1^2 + x_1 x_2$$
- Is: $\nabla f(x) = \begin{pmatrix} 2x_1 + x_2 \\ x_1 \end{pmatrix}$
- The Hessian is: $\nabla^2 f(x) = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$
- The Jacobian of the constraint function $g(x) = -x_1 x_2 + 300/\pi$ is $J(x) = (-2x_1 x_2 \quad -x_1^2)$

Types of optimization problems

- Linear programming (LP):

$$\min_x c^T x, Ax = b \text{ or } Ax \le b, x \ge 0$$

- Integer programming (IP):

$$\min_{x,y} c^T x + d^T y, \text{ so that } Ax + Dy \le b$$

with $y_i, i = 1,...,r$ integer variables

- Quadratic programming (QP):

$$\min_x \frac{1}{2} x^T Q x + c^T x, Ax \le b$$

Types of optimization problems (2)

- (Unconstrained) Nonlinear optimization:

$$\min_x f(x)$$

- Nonlinear least squares problems:

$$\min_x \sum_i f_i(x)^2$$

- Nonlinear optimization, linear constraints:

$$\min_x f(x), Ax = b \text{ or } Ax \le b$$

- Nonlinear optimization, nonlinear constraints:

$$\min_x f(x), g_i(x) \le 0, h_j(x) = 0$$

Special optimization problems

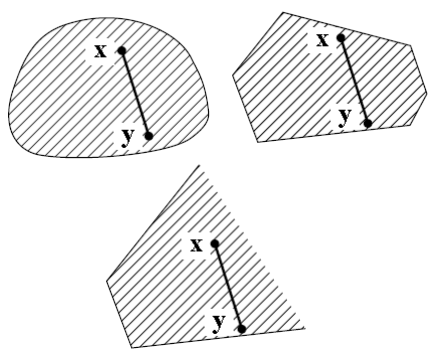
- Global optimization
- Non-smooth optimization
- Optimal control
- Dynamic programming
- Min-max problems:

$$\min_{x} \max_{t} \{ f_i(x), i = 1,..., m \} \text{ so that } x \in F$$

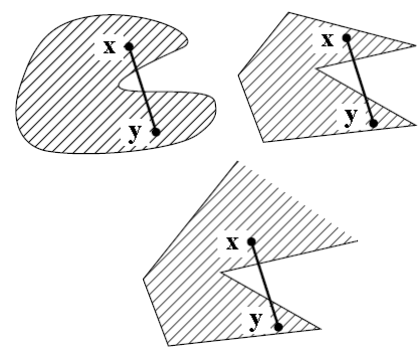
- Combinatorial optimization
- Graph problems (e.g. network flow)

Convex sets and optimization

- An optimization problem is convex when the objective function and the feasible set are convex



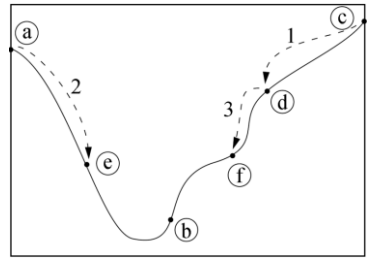
Convex sets



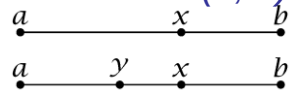
Non-Convex sets

Scalar functions

- Let $f : \Re \to \Re$ be continuous
- Principle of optimization



- The best choice is the *golden ratio*: lengths *(a, x)* and *(x, b) are* $\frac{3-\sqrt{5}}{2} \approx 0{,}38197$ *and* $\frac{\sqrt{5}-1}{2} \approx 0{,}61803$ compared to the total length of the interval *(a, b)*

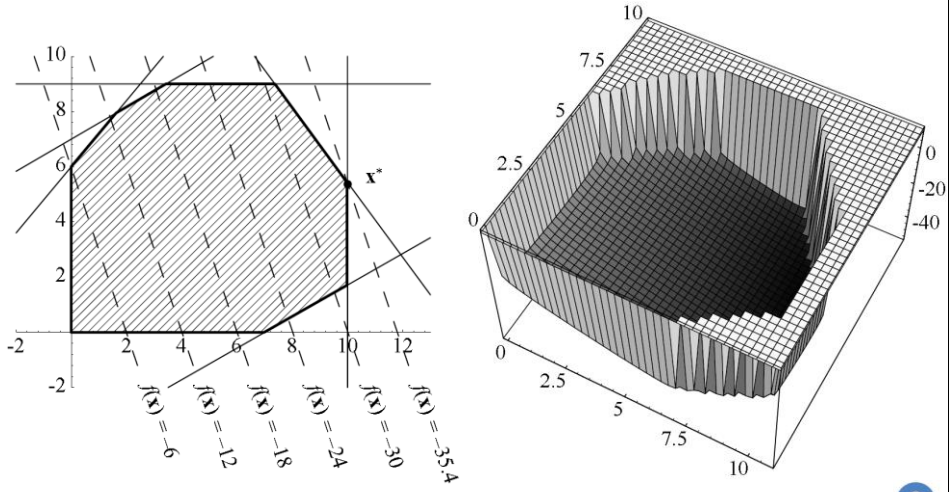


Linear programming

- Considering $\min_x c^T x, Ax \le b, x \ge 0$
- And the example $\min_x f(x) = -3x_1 - x_2$ so that

$$\begin{cases} -6x_1 + 5x_2 \le 30 \\ -7x_1 + 12x_2 \le 84 \\ x_2 \le 9 \\ 19x_1 + 14x_2 \le 266 \\ x_1 \le 10 \\ 4x_1 - 7x_2 \le 28 \\ x_1 \ge 0, x_2 \ge 0 \end{cases}$$

Linear programming (2)



Integer programming

- Mixed-integer programming (MIP):

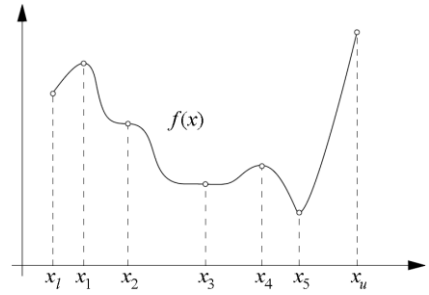
$$\min_{x,y} c^T x + d^T y, \text{ so that } Ax + Dy \le b$$

  where $y_i, i = 1,\ldots,r$ are integer-valued variables, and $x \ge 0, \quad 0 \le y \le w$

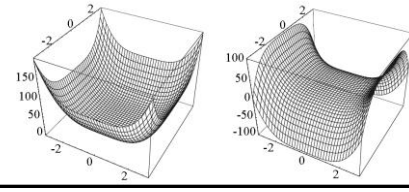
- Integer programming is much harder than LP
- Example:
  - 35 binary variables (0/1)
  - There are $2^{35} \approx 34 \times 10^9$ cases
  - If you can handle 1000 cases in one second, it would take 400 days to solve the problem

Integer programming (2)

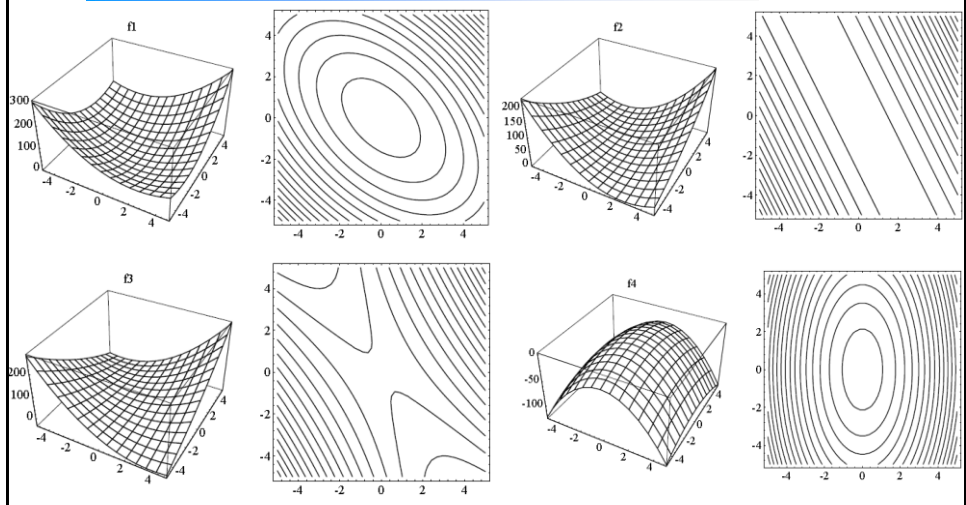
- There are approximate and heuristic solution methods for MIP problems
- Local minimum
  - There is a neighborhood with radius $r, r > 0$ where the function has the minimum value on the center x^*



- Global minimum is the smallest of the local minima
- Saddle point

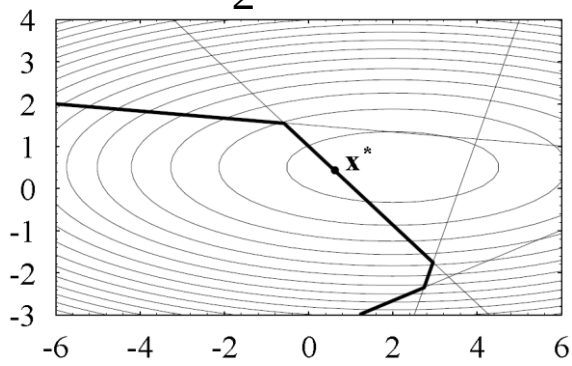


Quadratic functions



Quadratic programming

- The problem $\min_x \frac{1}{2} x^T Q x + c^T x,\ Ax \le b$



- If the matrix Q is positive definite → this is a convex optimization problem = easy

Nonlinear optimization

- Optimizing continuous function $f : \Re^n \to \Re$

$$\min_x f(x), g_i(x) \le 0, \mathrm{i} = 1,...,\mathrm{p},$$

$$h_j(x) = 0,\ j = 1,...,1.$$

- Function *f(x)* is the objective function
- Functions $g_i(x)$ and $h_j(x)$ are constraints
- **There is no general method for solving nonlinear optimization problems**
- Therefore we will first look at unconstrained optimization

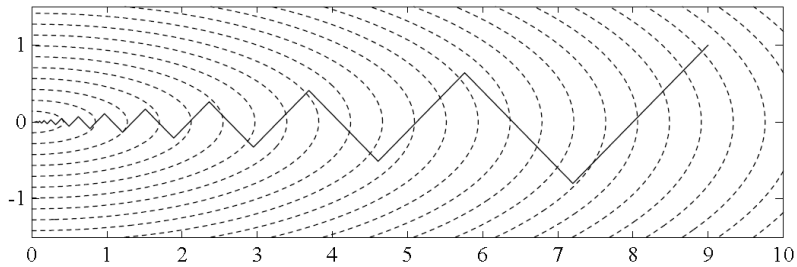
Unconstrained optimization: steepest descent (bad method!)

- Select the direction, where the gradient is steepest:

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

- Optimizing a quadratic function

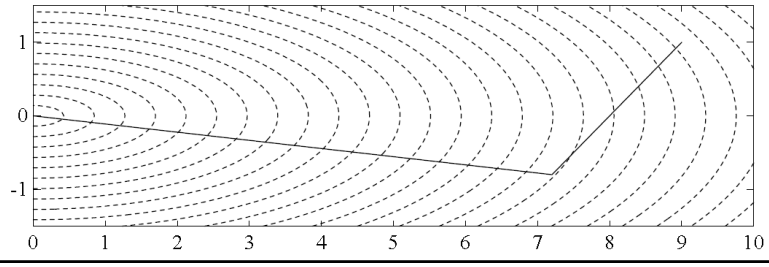
$$f(x) = \tfrac{1}{2} x_1^2 + \tfrac{9}{2} x_2^2$$



- Converges only linearly, and may be very slow!

Conjugate gradient method

- Modest memory requirements
- Convergence is (super)linear
- Finds the minimum of an *n* dimensional quadratic function in *n* steps
- Optimizing a quadratic function $f(x) = \tfrac{1}{2} x_1^2 + \tfrac{9}{2} x_2^2$

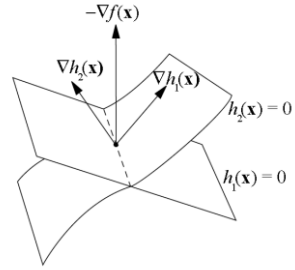


Constrained nonlinear optimization

- The problem: $\min_{x \in \Re^n} f(x), g_i(x) \le 0, \mathrm{i} = 1,...,\mathrm{p},$

$$h_j(x) = 0,\ j = 1,...,1.$$

- The constraints may be linear or nonlinear



- Sequential quadratic programming (SQP) may be the most **used** and most **robust** method

Nonlinear least squares

- The problem: $\min_x F(x) = \sum_{i=1}^{n} (f_i(x))^2 = f(x)^T f(x)$

where $x \in \Re^m$ and $f : \Re^m \to \Re^n$

- Newton's iteration for the problem

$$(J(x^k)^T J_k + S_k)s_k) = -J(x^k)^T f^k$$

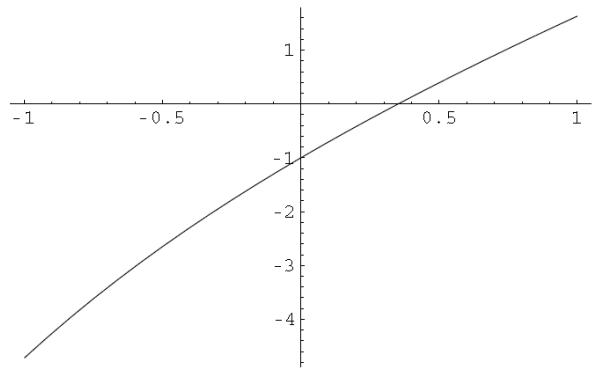
$$x^{k+1} = x^k + s^k$$

- This is a costly solution method, because you need $\frac{1}{2} mn(n+1)$ second derivatives to calculate S_k
- The most used methods are Gauss-Newton (simpler) and Levenberg-Marquardt (more robust)

Table of Contents

Nonlinear Equation Example

- The problem $f(x) = -e^{-x} + 2x = 0$



- The solution: $x = 0.351734$

Nonlinear equations

- Find the roots of the system of equations:

$$f : \Re^n \to \Re^n, \quad f(x) = 0$$

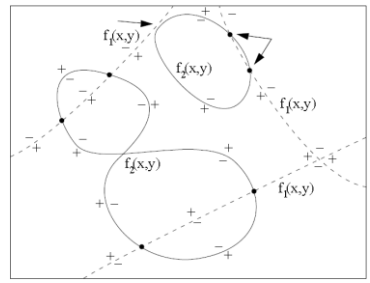
- Numerical methods find an approximate solution point $x^* \in \Re^n$
- If n > 1, we have the system of equations:

$$\begin{cases} f_1(x_1,...,x_n) = 0 \\ f_2(x_1,...,x_n) = 0 \\ ... \\ f_n(x_1,...,x_n) = 0 \end{cases}$$

- We have a linear system of equations, if $f(x) = Ax - b$ where A is a matrix and b is a vector in $\Re^n$

Finding the solution

- Solving a system of equations is much harder than solving a single equation!



- Numerical methods are iterative:
  - Given a starting point x^1 we compute $x^{k+1} = x^k + s^k, k = 1,2,...$
- A good starting point is essential for speedy convergence: x^1 should be near the solution x^*

189

Guidelines for solving the problem

- Find an area containing the root
- The better the initial guess is – the faster the convergence
- Select a suitable solution method depending on the problem type:
  - Real or complex problem
  - One or several equations
  - State of nonlinearity
- When candidate solution has been found:
  - Verify it

190

Solution methods for one equation

- If the **derivative is easy** to compute:
  - Newton's method is efficient
  - A good initial guess also helps
- If the derivative is **not available**
  - Brent's method is robust and fast
- Polynomial equations have special solution algorithms
- Solution methods are available in the NAG and IMSL commercial libraries

Solution methods for systems of equations

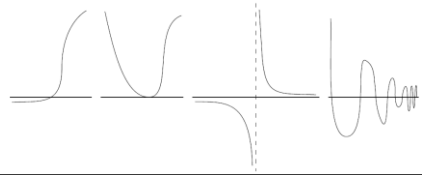
- If the derivatives are easy to compute and you have a good initial guess:
  - Newton's method is efficient
- When the derivatives are not available:
  - Quasi-Newton methods are good choices
- It is also possible to use methods for nonlinear least squares problems:

$$f(x) = 0 \rightarrow \min_x \sum_i f_i(x)^2$$

- However, the solutions of the minimization problem may not solve the system of equations

Problems with finding a root

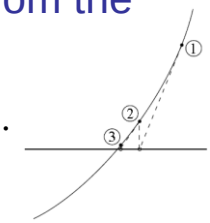
- If the function is continuous on the interval $[a,b]$ and $f(a)f(b) < 0$ there is at least one root
- Compare with the equation: $f(x) = (x-a)^2 = 0$
- Singularities: function $\sin(1/x)$ has infinitely many roots near the point $x = 0$
- Singular function $1/(x-a)$ changes sign near $x = a$ although there is no root
- Bad scaling: solution may lie in $x \in [1, 1+10^{-100}]$



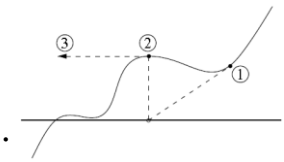
Newton's method for one equation

- Newton's method can be derived from the Taylor expansion of function $f(x)$:

$$f(x+d) \approx f(x) + f'(x)d + \frac{f''(x)}{2}d^2 + \ldots$$



- Setting $f(x+d)=0$ we get $d = -\frac{f(x)}{f'(x)}$
- Newton's method:
  1. Guess an initial solution x_1
  2. Compute $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, k = 1,2,3,\ldots$



- If $f'(x_k)=0$ for some *k*, the method **fails**
- If $f'(x^*)=0$ the method is **not efficient**

Newton's method for systems of equations

- The problem: $f(x)=0$
- Guess an initial solution $x^1 \in \Re^n$
- Solve the linear equations $J(x^k)s_N^k = -f(x^k)$

$$x^{k+1} = x^k + s_N^k$$

  with $J(x^k)$ is the Jacobian of the function *f*, and s_N^k is the Newton step
- The iteration converges quadratically near the solution
  - May fail if the initial guess is poor
- The Jacobian matrix is **expensive** to compute

Software for nonlinear equations

- Mathematica and Matlab offer routines for solving simple nonlinear equations
  - Not too many variables
  - Too much nonlinearity
- IMSL and NAG subroutine libraries offer reliable routines for Fortran & C programmers
- The Netlib archive (http://www.netlib.org) contains source code for solution routines

Table of Contents