Table of Contents

102

Finite Difference Models

- Discrete time steps
- State variables depend on their values at **previous** steps (mostly recursive systems)

$$N_{t+1}^{i} = f^{i}(N_t^1, N_t^2, \ldots, N_{t-1}^1, N_{t-1}^2, \ldots, t)$$

- Sometimes analytically solvable
- Simpler versions:
  - Only one previous time step affects the new values
  - Only one state variable
  - Linear dependence
- Analytical methods exist → **insight**
- Computers are very powerful → **numerical results**
- Example: Density Independent Population (Virus…) Growth

$$N_{t+1} = N_t + kN_t = N_0(1+k)^{t+1}$$

103

Useful Calculus Concepts

- A mapping $f(t):\Re \rightarrow \Re$ is a function if for each *t* there is only one value of *f(t)*
- **Difference quotient** of $f(t): \dfrac{f(t+h)-f(t)}{h}$
- **Derivative** $f'(t) = \lim_{h\to 0} \dfrac{f(t+h)-f(t)}{h}$
  - Measures the rate of change of *f(t)* at *t*
- To **differentiate** a function means to calculate its derivative
- The inverse operation is **integration**: $F' = f \Rightarrow F = \int f + C$
- However, not all functions are differentiable:
  - There a points where the derivative does not exist
  - For example: f(t) = |t| at t = 0
  - Sometimes these functions appear in models, so proper attention must be paid to their handling

Differential equations

- A differential equation relates the behavior of a function with its derivative(s).

$$G(t, f(t)), f'(t), f''(t), \dots, t^{(n)}) = 0$$

- In addition to the actual equation, other conditions are required in order to guarantee a unique solution.
- These can be either initial or boundary conditions.
  - Initial conditions are given at one point $t = t_0$
  - Boundary conditions are given at separate points

$$t = t_1, t = t_2, \dots, t = t_n$$

Numerical treatment of differential equations

- Ordinary differential equations (ODE) & partial differential equations (PDE) are used in physics to
  - Describe phenomena such as the flow of air around an aircraft
  - Bending of a bridge under various stresses
- Obtaining useful information however
  - “How much does this bridge sag if there are a hundred cars on it” is not that easy
- Techniques are required to turn ODEs & PDEs into computable problems

106

Ordinary Differential Equations

- Ordinary differential equations describe
  - How a quantity (scalar/vector) depends on a single variable
  - Typically, this variable denotes **time**
  - The value of the quantity at some starting time is given
  - This type of equation is called an Initial Value Problem (**IVP**)

107

Numerical methods for ODEs

- The most elementary / commonly used methods are **finite difference methods**
- Numerical solution results in two sets of data:
  - The argument **points**
  - The **function values** corresponding to these arguments
- Difference methods
  - Based on **approximating** the derivative(s) of a function in some interval with linear combinations of the function values

Numerical methods for ODEs (2)

- The first derivative of a function $f(t)$ is approximated by $$f'(t_k) \approx \sum_{j=k-j_-}^{k+j_+} a_j f(t_j)$$
- If the summation index j takes
  - Only values smaller than k, the method is **explicit**
  - Otherwise it is **implicit** and requires one or more algebraic equations
- The difference between the arguments is called step size
  - Usually denoted by h
  - h does not have to be constant

110

Partial Differential Equations

- Partial differential equations describe functions of several variables:
  - Denoting space and time
  - Similar initial values in ODEs, PDEs need values in space to give a uniquely determined solution
  - These are called boundary values → the problem is called a Boundary Value Problem (BVP)
  - Boundary value problems typically describe static mechanical structures

110

111

Sample Problem

- A heat equation has aspects of both IVPs and BVPs as:
  - It describes heat spreading through a physical object such as a rod
  - The initial value describes the initial temperature
  - The boundary values give prescribed temperatures at the ends of the rod
- Simplifications:
  - All functions involved have sufficiently many higher derivatives
  - Each derivative is sufficiently smooth

111

Initial Value Problems – IVP

- Many physical phenomena change over time, and typically the laws of physics give a description of the change, rather than of the quantity of interest itself.
  - Newton's second law $F = ma$
  - Is a statement about the change in position of a point mass expressed as

$$a = \frac{d^2}{dt^2} x = F/m$$

  - It states that acceleration depends linearly on the force exerted on the mass

Initial Value Problems – IVP (2)

- A closed form description $x(t) = \ldots$ can sometimes be derived analytically
- Usually some form of approximation or numerical computation is needed
- Newton's equation is a second order ODE, since it involves a second derivative
- This can be reduced to first order if we allow vector quantities: $u(t) = (x(t), x'(t))$

$$u' = Au + B, \qquad A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} 0 \\ F/a \end{pmatrix}$$

Initial Value Problems – IVP (3)

- Here we only consider scalar equations → the equation becomes:

$$u'(t) = f(t, u(t)), \qquad u(0) = u_0, \qquad t > 0$$

- Several numerical methods can solve it
- The initial value in some starting point t, we are interested in the behavior of u

$$t = 0, \quad u(0) = u_0, \quad \mathrm{t} \to \infty$$

- As an example: $f(x) = x \quad \to \quad u'(t) = u(t)$
- The equation states that the rate of growth is equal to the size of the population

Initial Value Problems – IVP (4)

- We consider the numerical solution and the accuracy of this process
- In a numerical method:
  - We consider **discrete size time steps** to approximate the solution of the **continuous time-dependent** process
  - This introduces a certain amount of **error**:
    - Analyze the error introduced in each time step
    - How this error adds up to a global error
- The need to limit the global error will impose restrictions on the used numerical scheme

116

Error and Stability

- Numerical computation involves inaccuracies
  - Machine arithmetic
  - Incremental errors: small perturbation in the initial value leads to large perturbations in the solution
- A differential equation is **stable** if solutions corresponding to **different initial values** u_0 converge to one another as $t \to \infty$
- Let us limit ourselves to the 'autonomous' ODE $u'(t) = f(u(t))$ in which the right hand side does not explicitly depend on $\boldsymbol{t}$

116

117

Criterium for Stability

- A sufficient criterium for stability is:

$$\frac{\partial}{\partial u} f(u) = \begin{cases} > 0 & \text{unstable} \\ = 0 & \text{neutrally stable} \\ < 0 & \text{stable} \end{cases}$$

- A simple example

  $f(u) = -\lambda u$ with solution $u(t) = u_0 e^{-\lambda t}$
- This problem is stable if $\lambda > 0$

117

Finite Difference Approximation

- Solving the problem numerically → transform the continuous problem into a discrete one by:
  - Looking at finite time/space steps
- Assuming all functions are sufficiently smooth, a straightforward Taylor expansion gives:

$$u(t+\Delta t) = u(t) + u'(t)\Delta t + u''(t)\frac{\Delta t^2}{2!} + u'''(t)\frac{\Delta t^3}{3!} + \ldots$$

- Thus $u'(t)$ is computed:

$$u'(t) = \frac{u(t+\Delta t) - u(t)}{\Delta t} + O(\Delta t^2)$$

Finite Difference Approximation (2)

The approximation is obtained by replacing a **differential operator** by a **finite difference**

$$u'(t) = \frac{u(t+\Delta t) - u(t)}{\Delta t}$$

- Substituting this in $u'(t) = f(t,u)$ gives

$$u(t+\Delta t) = u(t) + \Delta t \; f(t, u(t))$$

- If $t_0 = 0, t_{k+1} = t_k + \Delta t = \ldots = (k+1)\Delta t, u(t_k) = u_k$
- We thus get the **Explicit (forward) Euler** difference equation $u_{k+1} = u_k + \Delta t \; f(t_k, u_k)$

Explicit Euler: Alternative (easier) Formulation

120

- Considering the general first order differential equation $x'(t) = f(t, x(t))$ with some initial condition $x(0) = x_0$
- Euler's (explicit) method based on

$$x'(t) = \lim_{h \Rightarrow 0} \frac{x(t+h) - x(t)}{h}$$

- Instead of letting, $h \Rightarrow 0$ take a finite $h > 0$ and

$$x'(t) \approx \frac{x(t+h) - x(t)}{h}$$

120

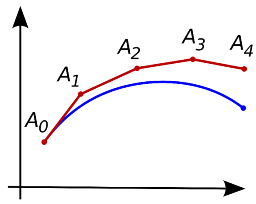
Explicit Euler: Alternative (easier) Formulation (2)

121

- Thus Euler's Explicit Method becomes

$$x(t+h) \approx x(t) + hx'(t) = x(t) + hf(t, x(t))$$

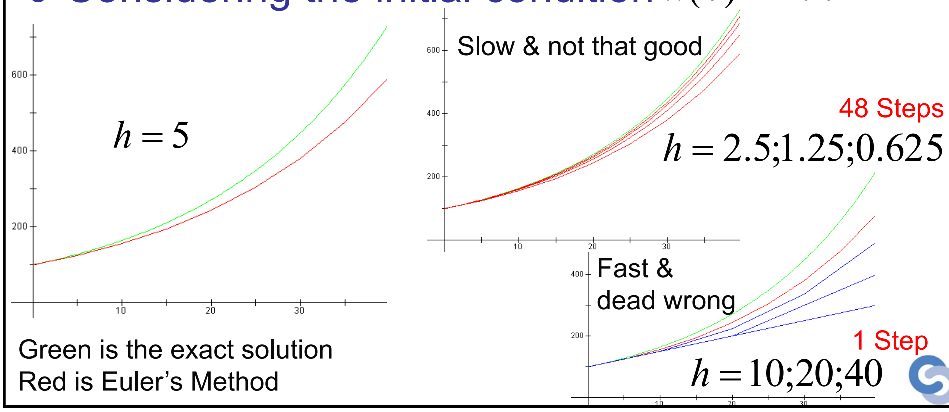
- Which is:
  - Simple to program
  - Very inefficient
  - It sometimes gives totally erroneous results
  - Highly dependent on the "right" choice for $\boldsymbol{h}$
  - Error is proportional in the first order with the step-size $\boldsymbol{h} \rightarrow$ Euler is a "**first order method**"



121

Euler's Method – Example

- Let us apply Euler's method to the equation $x'(t) = 0.05 \cdot x$ which has as solution $x = 100 \cdot e^{0.05t}$
- Considering the initial condition $x(0) = 100$



122

Implicit Euler's method

- Instead of using the value of the derivative at the present point in time t, use the value of the derivative at the future point in time $t+h$:

$$x'(t) = \lim_{h \Rightarrow 0} \frac{x(t) - x(t-h)}{h}$$

- Again, with a finite h we get an approximate equation

$$x'(t) \approx \frac{x(t) - x(t-h)}{h}$$

- And thus

$$x(t) \approx x(t-h) + hx'(t) = x(t-h) + hf(t, x(t))$$

123

124

Implicit Euler's method (2)

- As can be seen in
  $x(t) \approx x(t-h) + hx'(t) = x(t-h) + hf(t, x(t))$
- The unknown value $x(t)$ appears on **both sides** of the equation, and as an argument for the function $f(t,x)$ **on the right hand side**.
- This method is known as the **Implicit Euler method**.
- Finding $x(t)$ requires solving (possibly numerically) the equation for $x(t)$
- However, if an analytic formula for $x(t)$ exists, the method is very easy to use
- To solve Implicit Euler you need an **iterative solver**

124

125

Stability of Implicit Euler

- It is possible to take **larger time steps** without worrying about unphysical behavior
- Large time steps
  - Can make convergence to the steady state slower
  - But at least there will be no divergence
- Drawback – implicit methods are more complicated
  - Can involve nonlinear systems of equations to be solved in every time step

125

Systems of Differential Equations

- Systems of differential equations describe the dynamics of complicated models
- There are usually two or more state variables $x_j(t)$ whose time development should be studied
  - For a specified time interval
  - For an arbitrary long period of time
- Some systems contain both **algebraic** and **differential equations** (DAEs)
- Two-dimensional models are easy to analyze
  - It is often possible to draw pictures of what is going on

Systems of Differential Equations (2)

- Generic form of a system of differential equations

$$x_1'(t) = f_1(t, x_1(t), x_2(t), ..., x_n(t))$$
$$x_2'(t) = f_2(t, x_1(t), x_2(t), ..., x_n(t))$$
$$...$$
$$x_n'(t) = f_n(t, x_1(t), x_2(t), ..., x_n(t))$$

- If none of the functions f_j depend explicitly on t, the system is **autonomous**
- If every function f_j is linear, the system is said to be **linear**

Systems of Differential Equations (3)

- In order to have a unique solution
  - Initial and/or boundary conditions are needed
- Initial conditions are given at the moment $t = 0$
- Sometimes conditions are also given at the end point of the time interval
- It is also possible that part of the conditions are specified at the starting point and part at the end

General Model Behavior

- Systems with two state variables can exhibit: equilibrium and limit cycles
- Equilibrium behavior:
  - **Stable**: starting near an equilibrium, will keep you near that equilibrium
  - **Asymptotically stable**: starting near an equilibrium, will drift you closer and closer to the equilibrium
  - **Unstable**: starting exactly at the equilibrium, will keep you there. Any perturbation, however small, will drive you away
  - **Saddle point**: depending on the direction w.r.t. the equilibrium you can either drift towards it or away from it

130

General Model Behavior (2)

- Limit cycles:
  - **Neutral**: small perturbations will move you to another cycle
  - **Stable**: the effect of small perturbations will gradually disappear and the system drifts back to the original cycle
  - **Unstable**: small perturbations drive the system away from the cycle
- Possibly only numerical results
- Usually the long-term behavior is what matters

130

131

Table of Contents

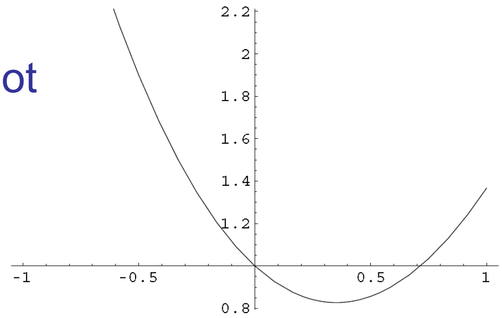
131

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: $f(x) = 0.827 \quad \text{at} \quad x = 0.352$

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 gradient of the objective function

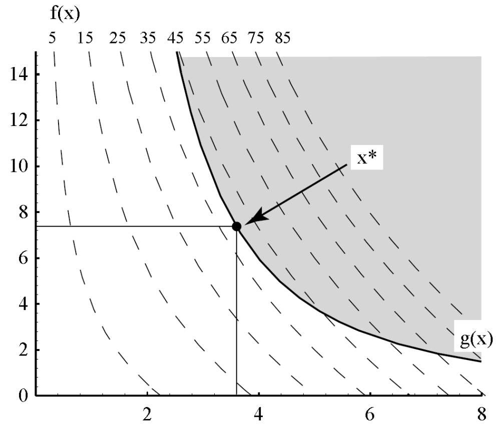
$$f(x) = x_1^2 + x_1 x_2$$

- 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 is

$g(x) = -x_1 x_2 + 300/\pi \qquad J(x) = (-2x_1 x_2 \quad -x_1^2)$

A 2D optimization problem (3)

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



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 $y_i, i = 1, \ldots, r$ 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:

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

138

Special optimization problems

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

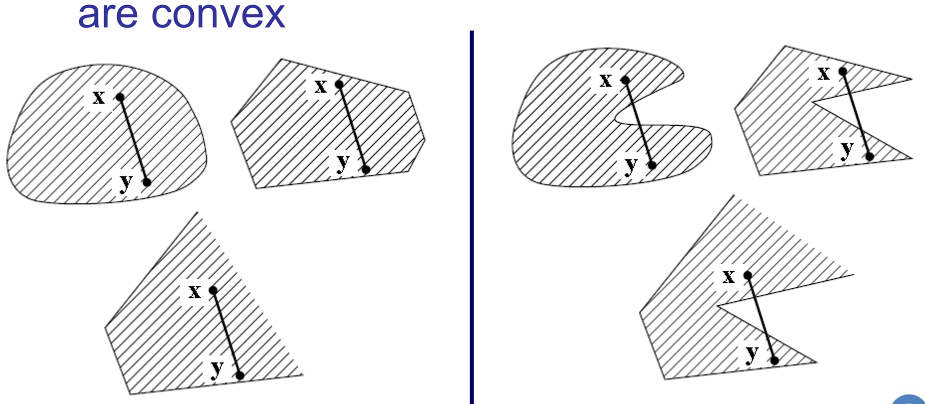
$$\min_x \max_t \{f_i(x), i = 1,\ldots,m\} \text{ so that } x \in F$$

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

139

Convex sets and optimization

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

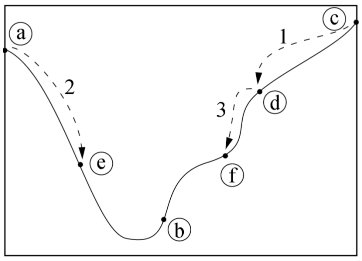


Convex sets Non-Convex sets

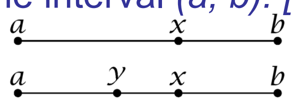
140

Scalar functions

- Let $f:\Re \rightarrow \Re$ 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)*: *[a, b] is to [a, x] as [a, x] is to [x, b]*



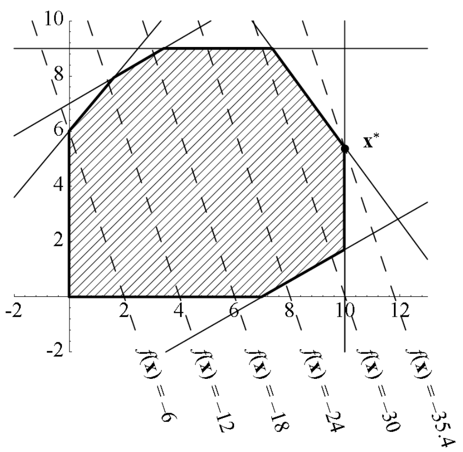
141

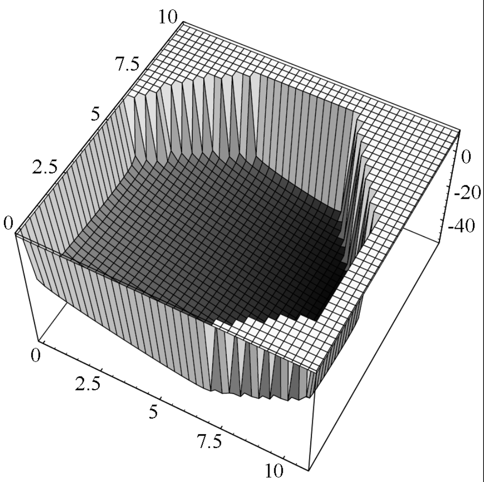
Linear programming

- 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

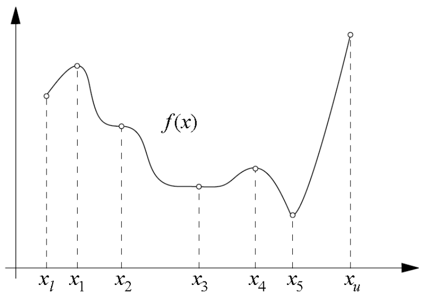
- Mixed-integer programming (MIP):

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

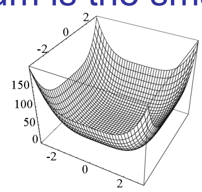
  where $y_i, i = 1,\ldots,r$ are integer-valued variables, and $x \ge 0, \quad 0 \le y \le w$
- Integer programming is much harder than LP
- Example:
  – 35 binary variables (0/1)
  – There are $2^{35} \approx 34 \times 10^9$ cases
  – If you can handle 1000 cases in one second, it would take 400 days to solve the problem

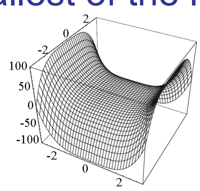
Integer programming (2)

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

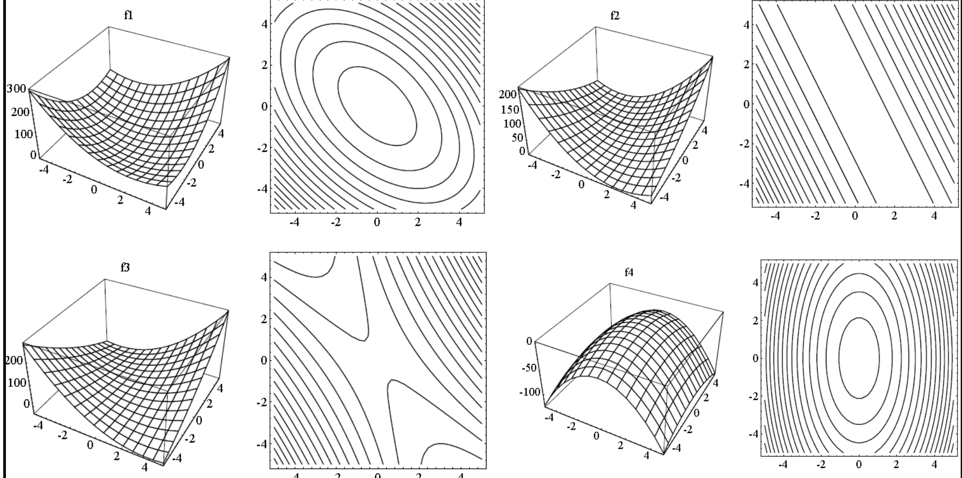


- Global minimum is the smallest of the local minima
- Saddle point



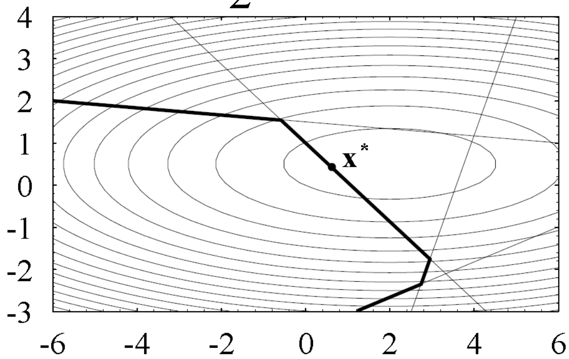


Quadratic functions



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

148

Nonlinear optimization

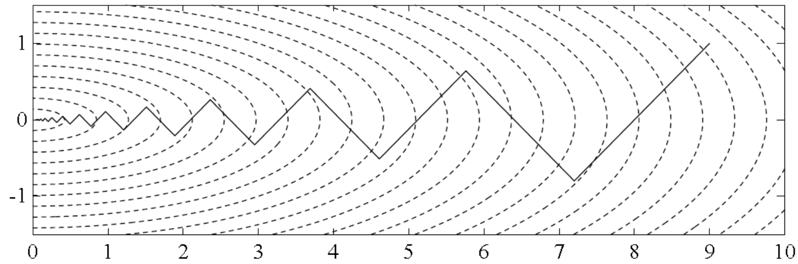
- Optimizing continuous function $f:\Re^n \rightarrow \Re$
$$\min_x f(x), g_i(x) \le 0,\ \mathrm{i}=1,\ldots,\mathrm{p},$$
$$h_j(x)=0, j=1,\ldots,r.$$
- Function *f(x)* is the objective function
- Functions $g_i(x)$ and $h_j(x)$ are constraints
- **There is no general method for solving nonlinear optimization problems**
- Therefore we will first look at unconstrained optimization

148

149

Unconstrained optimization: steepest descent (pretty 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
$$f(x)=\tfrac{1}{2}x_1^2+\tfrac{9}{2}x_2^2$$

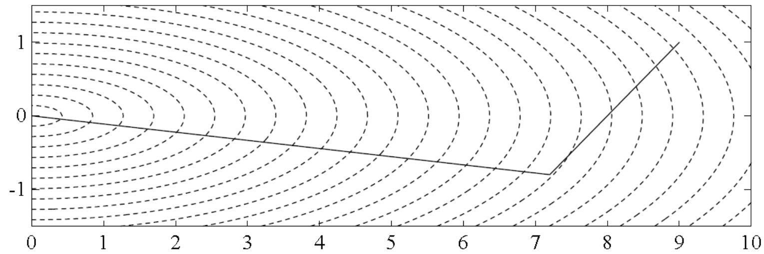


- Converges only linearly, and may be very slow!

149

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

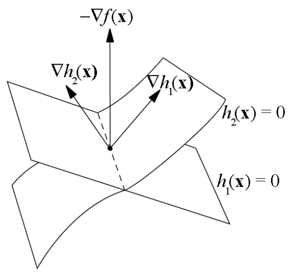


Constrained nonlinear optimization

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

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

- The constraints may be linear or nonlinear



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