Numerical Methods and Stochastics
Part I: Numerical Methods

R. Verfürth

Fakultät für Mathematik
Ruhr-Universität Bochum

www.ruhr-uni-bochum.de/num1

Lecture Series / Bochum / Summerterm 2011

Contents

Boundary Value Problems for Ordinary Differential Equations

- Initial Value Problems
- Numerical Methods for Initial Value Problems
- Boundary Value Problems
- Simple Shooting
- Multiple Shooting
- Finite Difference Methods
- Variational Methods

Initial Value Problems

- Examples
- Equations of higher order
- Existence and uniqueness of solutions
- Dependence on the initial value

Initial Value Problems for 1st Order Ordinary Differential Equations

- Given:
  - I interval
  - $D \subset \mathbb{R}^d$ set
  - $f(t, y) : I \times D \to \mathbb{R}^d$ function
  - $t_0 \in I$ initial time
  - $y_0 \in D$ initial value
- Sought:

  Differentiable function $y(t) : I \to D$ with
  $y'(t) = f(t, y(t))$ for all $t \in I$ (differential equation)
  and $y(t_0) = y_0$ (initial condition)

Example: Constant Birth or Death Rate

- $y'(t) = \lambda y(t)$, $y(0) = c$
- Corresponds to:
  - $I = \mathbb{R}$,
  - $D = \mathbb{R}$,
  - $f(t, y) = \lambda y$,
  - $t_0 = 0$,
  - $y_0 = c$
- Solution:

  $y(t) = ce^{\lambda t}$

Example: Damped Oscillation

- $y'(t) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix} y(t)$, $y(0) = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$
- Corresponds to:
  - $I = \mathbb{R}$,
  - $D = \mathbb{R}^2$,
  - $f(t, y) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix} y$,
  - $t_0 = 0$,
  - $y_0 = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$
- Solution:

  $y(t) = e^{\lambda t} \begin{pmatrix} c_1 \cos(\omega t) - c_2 \sin(\omega t) \\ c_1 \sin(\omega t) + c_2 \cos(\omega t) \end{pmatrix}$

Example: Exploding Solution

- $y'(t) = y(t)^2$, $y(0) = 1$
- Corresponds to:
  - $I = \mathbb{R}$,
  - $D = \mathbb{R}$,
  - $f(t, y) = y^2$,
  - $t_0 = 0$,
  - $y_0 = 1$
- Solution:

  $y(t) = \dfrac{1}{1 - t}$ explodes for $t \to 1-$

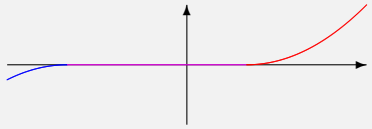
Example: Many Solutions

- $y'(t) = \sqrt{|y(t)|}$, $y(0) = 0$
- Corresponds to:
  - $I = \mathbb{R}$,
  - $D = \mathbb{R}$,
  - $f(t, y) = \sqrt{|y|}$,
  - $t_0 = 0$,
  - $y_0 = 0$
- Solutions:

  $y(t) = 0$

  $$y(t) = \begin{cases} 0 & \text{for } t < 0 \\ \frac{1}{4}t^2 & \text{for } t \geq 0 \end{cases}$$

  and infinitely many further solutions



Differential Equations of Higher Order

- Differential equations of higher order can be transformed into systems of 1st order by introducing new unknowns.
- Example: mechanical system
  - $Mx''(t) + Rx'(t) + Kx(t) = F(t)$, $x(0) = x_0$, $x'(0) = v_0$
  - Introducing $v(t) = x'(t)$ leads to
    $x'(t) = v(t)$,
    $v'(t) = M^{-1}F(t) - M^{-1}Rv(t) - M^{-1}Kx(t)$,
    $x(0) = x_0$, $v(0) = v_0$
  - This corresponds to
    $y(t) = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}$,
    $f(t, y) = \begin{pmatrix} 0 \\ M^{-1}F(t) \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ -M^{-1}K & -M^{-1}R \end{pmatrix} y$

Unique Solvability

- If f is continuously differentiable w.r.t. the variable y, there is an interval $J = (t_-, t_+) \subset I$ with $t_0 \in J$ and a unique function y, which is continuously differentiable on J and which solves the initial value problem $y'(t) = f(t, y(t))$, $y(t_0) = y_0$.
- Either $J = I$ or $y(t)$ tends to the boundary of D for $t \to t_\pm$.
- If the derivative of f w.r.t. the variable y remains bounded on $I \times D$, then $J = I$.

Dependence on the Initial Values

- If f is twice continuously differentiable w.r.t. the variable y, the solution y of the initial value problem $y'(t) = f(t, y(t))$, $y(t_0) = y_0$ is a differentiable function of the initial value y_0, i.e. $y(t) = y(t; y_0)$.
- The derivative $Z(t)$ of the function $y_0 \mapsto y(t; y_0)$ solves the initial value problem

  $Z'(t) = D_y f(t, y(t; y_0))Z(t)$, $Z(t_0) = I$.

  Here $D_y f(t, y)$ denotes the Jacobian of f w.r.t. the variable y and I is the identity matrix.

Example: Damped Oscillation

- $y'(t) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix} y(t)$, $y(0) = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$
- $y(t) = e^{\lambda t} \begin{pmatrix} c_1 \cos(\omega t) - c_2 \sin(\omega t) \\ c_1 \sin(\omega t) + c_2 \cos(\omega t) \end{pmatrix}$
- $D_y f(t,y) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix}$
- $Z(t) = \begin{pmatrix} z_{1,1}(t) & z_{1,2}(t) \\ z_{2,1}(t) & z_{2,2}(t) \end{pmatrix}$
- $Z'(t) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix} Z(t)$, $Z(0) = I$
- $z'_{1,1}(t) = \lambda z_{1,1}(t) - \omega z_{2,1}(t)$, $z_{1,1}(0) = 1$
  $z'_{1,2}(t) = \lambda z_{1,2}(t) - \omega z_{2,2}(t)$, $z_{1,2}(0) = 0$
  $z'_{2,1}(t) = \omega z_{1,1}(t) + \lambda z_{2,1}(t)$, $z_{2,1}(0) = 0$
  $z'_{2,2}(t) = \omega z_{1,2}(t) + \lambda z_{2,2}(t)$, $z_{2,2}(0) = 1$

Numerical Methods for Initial Value Problems

- Basic idea
- Runge-Kutta methods
- Order
- Stability

Basic Idea

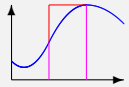
- Approximate the solution y of the initial value problem at discrete times $t_0 < t_1 < t_2 < \ldots$.
- Denote by $h_i = t_{i+1} - t_i$ the i-th time step size.
- The simplest scheme corresponds to $h_i = h$ for all i, i.e. $t_i = t_0 + ih$.
- Denote by η_i the approximation for $y(t_i)$.
- Compute η_{i+1} using f and η_i (single step methods) or using f and $\eta_i, \ldots, \eta_{i-m}$ (multi step methods).
- Many methods, in particular Runge-Kutta methods, are obtained by applying a suitable quadrature formula to the integral in the identity

$$\eta_{i+1} - \eta_i \approx y(t_{i+1}) - y(t_i) = \int_{t_i}^{t_{i+1}} f(s, y(s))ds.$$

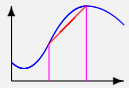
The Simplest Methods

- Explicit Euler Scheme:
  $\eta_0 = y_0$,
  $\eta_{i+1} = \eta_i + h_i f(t_i, \eta_i)$,
  $t_{i+1} = t_i + h_i$

- Implicit Euler Scheme:
  $\eta_0 = y_0$,
  $\eta_{i+1} = \eta_i + h_i f(t_{i+1}, \eta_{i+1})$,
  $t_{i+1} = t_i + h_i$



- Trapezoidal Rule alias Crank-Nicolson Scheme:
  $\eta_0 = y_0$,
  $\eta_{i+1} = \eta_i + \frac{h_i}{2}\left[f(t_i, \eta_i) + f(t_{i+1}, \eta_{i+1})\right]$,
  $t_{i+1} = t_i + h_i$



Runge-Kutta Methods

- $\eta_0 = y_0$

  $\eta_{i,j} = \eta_i + h_i \sum_{k=1}^{r} a_{jk} f(t_i + c_k h, \eta_{i,k})$ for $j = 1, \ldots, r$

  $\eta_{i+1} = \eta_i + h_i \sum_{k=1}^{r} b_k f(t_i + c_k h, \eta_{i,k})$

  $t_{i+1} = t_i + h_i$
- $0 \leq c_1 \leq \ldots \leq c_r \leq 1$
- r is called the stage number of the Runge-Kutta method.
- The method is called explicit, if $a_{jk} = 0$ for all $k \geq j$.
- The method is called implicit, if $a_{j,k} \neq 0$ for at least one $k \geq j$.

Order

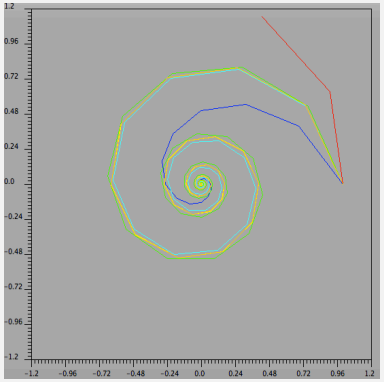
- A single step method is said to have the order $p > 0$, if $|y(t_1) - \eta_1| = O(h_1^{p+1})$.
- The order is a measure for the error committed by performing a single step of the method.
- If a single step method has order p and if f is continuously differentiable w.r.t. the variable y with a bounded derivative, then $|y(t_i) - \eta_i| = O\Big(\big(\max_{1 \leq j \leq i} h_j\big)^p\Big)$ for all i.
- Both Euler schemes are of order 1.
- The Crank-Nicolson scheme has order 2.

Stability

- The numerical method should yield a qualitatively correct solution for a large as possible range of step sizes.
- For the initial value problem $y'(t) = -100y(t)$, $y(0) = 1$ with exact solution $y(t) = e^{-100t}$ we obtain:
  - The explicit Euler scheme yields a decaying numerical solution only if $h_i \leq \frac{1}{50}$ for all i.
  - The implicit Euler and the Crank-Nicolson scheme both yield a decaying numerical solution for every step size.
- Explicit schemes cannot be stable.
- There are stable implicit Runge-Kutta schemes of arbitrary order.

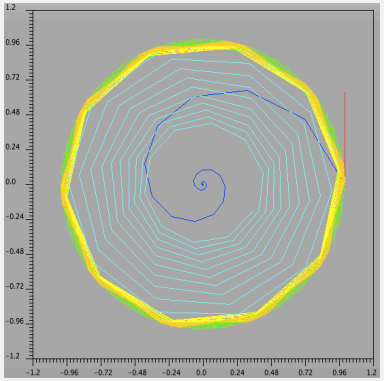
Example: Damped Oscillation

- $y'(t) = \begin{pmatrix} -0.9 & -6.3 \\ 6.3 & -0.9 \end{pmatrix} y(t)$
  $y(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
- Solution:
  $y(t) = e^{-0.9t} \begin{pmatrix} \cos(6.3t) \\ \sin(6.3t) \end{pmatrix}$
- 100 steps of
  explicit Euler,
  implicit Euler,
  Crank-Nicolson,
  classical Runge-Kutta,
  SDIRK order 3,
  SDIRK order 4
  with $h_i = 0.1$ for all i



Example: Undamped Oscillation

- $y'(t) = \begin{pmatrix} 0 & -6.3 \\ 6.3 & 0 \end{pmatrix} y(t)$
  $y(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
- Solution:
  $y(t) = \begin{pmatrix} \cos(6.3t) \\ \sin(6.3t) \end{pmatrix}$
- 100 steps of
  explicit Euler,
  implicit Euler,
  Crank-Nicolson,
  classical Runge-Kutta,
  SDIRK order 3,
  SDIRK order 4
  with $h_i = 0.1$ for all i



Boundary Value Problems

- Examples
- Existence and uniqueness of solutions

Boundary Value Problems for 1st Order Differential Equations

- Given:
  - I interval
  - $a, b \in I$ two different points
  - $D \subset \mathbb{R}^d$ set
  - $f(t, y) : I \times D \to \mathbb{R}^d$ function
  - $r(u, v) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ function
- Sought:

  Differentiable function $y(t) : I \to D$ with
  $y'(t) = f(t, y(t))$ for all $t \in I$ (differential equation)
  and $r(y(a), y(b)) = 0$ (boundary condition)

Example: Damped Oscillation

- $y'(t) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix} y(t)$, $y_1(0) = 1$, $y_1(\frac{\pi}{2\omega}) = 0$
- Corresponds to:
  - $I = \mathbb{R}$,
  - $D = \mathbb{R}^2$,
  - $f(t, y) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix} y$,
  - $a = 0$,
  - $b = \frac{\pi}{2\omega}$,
  - $r(u, v) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} u + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} v - \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
- Solution:
  $y(t) = e^{\lambda t} \begin{pmatrix} \cos(\omega t) \\ \sin(\omega t) \end{pmatrix}$

Example: Mechanical System

- $Mx''(t) + Rx'(t) + Kx(t) = F(t)$, $x(0) = x_0$, $x(L) = x_L$
- Introducing $v(t) = x'(t)$ leads to
  $x'(t) = v(t)$,
  $v'(t) = M^{-1}F(t) - M^{-1}Rv(t) - M^{-1}Kx(t)$,
  $x(0) = x_0$, $x(L) = x_L$
- This corresponds to
  $y(t) = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}$,
  $f(t,y) = \begin{pmatrix} 0 \\ M^{-1}F(t) \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ -M^{-1}K & -M^{-1}R \end{pmatrix} y$,
  $r(u,v) = \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} u + \begin{pmatrix} 0 & 0 \\ I & 0 \end{pmatrix} v - \begin{pmatrix} x_0 \\ x_L \end{pmatrix}$

Example: Eigenvalue Problem

- Sought are $u : [a,b] \to \mathbb{R}$ and $\lambda \in \mathbb{R}$ with
  $u'(t) = g(t, u(t))$, $\rho(u(a), u(b), \lambda) = 0$
- Corresponds to:
  - $y(t) = \begin{pmatrix} u(t) \\ \lambda \end{pmatrix}$,
  - $f(t,y) = \begin{pmatrix} g(t,y_1) \\ 0 \end{pmatrix}$,
  - $r(u,v) = \rho(u_1, v_1, v_2)$

Example: Free Boundary Problem

- Sought are $\beta > 0$ and $u : [0, \beta] \to \mathbb{R}$ with
  $u'(s) = g(s, u(s))$, $\rho(u(0), u(\beta)) = 0$
- Corresponds to:
  - $y(t) = \begin{pmatrix} u(t\beta) \\ \beta \end{pmatrix}$,
  - $t = \frac{s}{y_2}$,
  - $f(t,y) = \begin{pmatrix} y_2 g(ty_2, y_1) \\ 0 \end{pmatrix}$,
  - $r(u,v) = \rho(u_1, v_1)$

Unique Solvability

- For boundary value problems there is no general existence and uniqueness result similar to the one for initial value problems.
- The solvability and the the number of eventual solutions depends on the particular example and the interplay of differential equation and boundary condition.
- Example: Oscillation
  - $y'(t) = \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix} y(t)$, $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} y(0) + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} y(L) = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$
  - General solution of the differential equation:
    $y(t) = \begin{pmatrix} c_1 \cos(\omega t) - c_2 \sin(\omega t) \\ c_1 \sin(\omega t) + c_2 \cos(\omega t) \end{pmatrix}$
  - $L = \frac{2\pi}{\omega}$, $\alpha = 0$, $\beta = 1$ leads to the contradictory conditions $c_1 = 0$ and $c_1 = 1$.
  - $L = \frac{2\pi}{\omega}$, $\alpha = 0$, $\beta = 0$ leads to the single condition $c_1 = 0$ such that c_2 is arbitrary.

Simple Shooting

- Basic idea
- Properties

Idea

- Denote by $y(t;s)$ the solution of the initial value problem $y'(t) = f(t, y(t))$, $y(a;s) = s$.
- Then $y(t;s)$ solves the boundary value problem $y'(t) = f(t, y(t))$, $r(y(a), y(b)) = 0$ iff $r(s, y(b;s)) = 0$.
- Using Newton's method compute a zero of the function $F(s) = r(s, y(b;s))$.
- The derivative $DF(s)$ of F at the point s is $DF(s) = D_u r(s, y(b;s)) + D_v r(s, y(b;s))Z(b;s)$, where Z solves the initial value problem $Z'(t;s) = D_y f(t, y(t;s))Z(t;s)$, $Z(a;s) = I$.
- Solve the initial value problems for $y(t;s)$ and $Z(t;s)$ approximately by a numerical method for initial value problems using the same discrete times t_i for both problems.

Simple Shooting

0. Given an initial guess $s^{(0)} \in \mathbb{R}^d$. Set $i = 0$.

1. Compute an approximation $\eta^{(i)}(t)$ for the solution $y^{(i)}$ of the initial value problem $y^{(i)'}(t) = f(t, y^{(i)}(t))$, $y^{(i)}(a) = s^{(i)}$. Set $F^{(i)} = r(s^{(i)}, \eta^{(i)}(b))$.

2. Using the same scheme and the same discrete times as in step **1** compute an approximation $\zeta^{(i)}(t)$ for the solution $Z^{(i)}$ of the initial value problem $Z^{(i)'}(t) = D_y f(t, \eta^{(i)}(t))Z^{(i)}(t)$, $Z^{(i)}(a) = I$. Set $D^{(i)} = D_u r(s^{(i)}, \eta^{(i)}(b)) + D_v r(s^{(i)}, \eta^{(i)}(b))\zeta^{(i)}(b)$.

3. Solve the linear system of equations $D^{(i)}\Delta s^{(i)} = -F^{(i)}$. Set $s^{(i+1)} = s^{(i)} + \Delta s^{(i)}$, increase i by 1 and return to step **1**.

Properties

- The initial value problems in step 1 have d unknowns.
- The initial value problems in step 2 have d^2 unknowns.
- The initial value problems in step 2 are linear.
- The linear systems of equations in step 2 have d equations and unknowns.
- Newton's method should be damped.
- If Newton's method converges, the convergence speed is quadratic.

A Warning Example

- Boundary value problem:
  $y'(t) = \left(\begin{smallmatrix} 0 & 1 \\ 110 & 1 \end{smallmatrix}\right) y(t)$, $y_1(0) = 1$, $y_1(10) = 1$
- Solution:
  $y(t) = c_1 e^{-10t} \left(\begin{smallmatrix} 1 \\ -10 \end{smallmatrix}\right) + c_2 e^{11t} \left(\begin{smallmatrix} 1 \\ 11 \end{smallmatrix}\right)$
  with $c_1 = \dfrac{e^{110} - 1}{e^{110} - e^{-100}}$, $c_2 = \dfrac{1 - e^{-100}}{e^{110} - e^{-100}}$
- The solution of the initial value problem with initial value s is:
  $y(t; s) = \dfrac{11 s_1 - s_2}{21} e^{-10t} \left(\begin{smallmatrix} 1 \\ -10 \end{smallmatrix}\right) + \dfrac{10 s_1 + s_2}{21} e^{11t} \left(\begin{smallmatrix} 1 \\ 11 \end{smallmatrix}\right)$
- Exact initial value:
  $s^* = \begin{pmatrix} 1 \\ -10+21 \cdot \frac{1-e^{-100}}{e^{110}-e^{-100}} \end{pmatrix}$
- The wrong initial value $\widetilde{s} = \left(\begin{smallmatrix} 1 \\ -10+10^{-9} \end{smallmatrix}\right)$ with a relative error of 10^{-10} yields $y_1(10; \widetilde{s}) \approx 10^{37}$.

Multiple Shooting

- Basic idea
- Properties

Observation

- Simple shooting breaks down since solutions corresponding to different initial values may run away with an exponential rate.
- This effect can be avoided by solving the initial value problems only on small time intervals.

Idea

- Subdivide the interval $[a, b]$ by choosing intermediate points $a = \tau_1 < \tau_2 < \ldots < \tau_m = b$.
- For $s_1, \ldots, s_m \in \mathbb{R}^d$ denote by $y(t; \tau_k, s_k)$ the solution of the initial value problem $y'(t) = f(t, y(t))$, $y(\tau_k; s_k) = s_k$.
- Define the piecewise function $\widetilde{y}$ by setting $\widetilde{y}(t) = y(t; \tau_k, s_k)$ for $\tau_k \le t < \tau_{k+1}$ and $1 \le k \le m-1$ and $\widetilde{y}(\tau_m) = s_m$.
- Then $\widetilde{y}$ solves the boundary value problem $y'(t) = f(t, y(t))$, $r(y(a), y(b)) = 0$ iff $y(\tau_{k+1}; \tau_k, s_k) = s_{k+1}$ for $1 \le k \le m-1$ and $r(s_1, s_m) = 0$.
- This corresponds to a system of equations $F(s_1, \ldots, s_m) = 0$ which can be solved with Newton's method.
- The evaluation of the derivative of F requires the solution of initial value problems on the intervals $[\tau_k, \tau_{k+1}]$.

Structure of DF

- DF has the structure $\begin{pmatrix} G_1 & -I & & & \\ & G_2 & -I & & 0 \\ & & \ddots & \ddots & \\ 0 & & & G_{m-1} & -I \\ A & 0 & & 0 & B \end{pmatrix}$
- Hence every Newton step requires the solution of a system of the form:
  $G_1\Delta s_1 - \Delta s_2 = -F_1$, ..., $G_{m-1}\Delta s_{m-1} - \Delta s_m = -F_{m-1}$,
  $A\Delta s_1 + B\Delta s_m = -F_m$
- Successive elimination of Δs_2, ..., Δs_m leads to
  $$(A + BG_{m-1}\dots G_1)\Delta s_1 = -F_m - B\sum_{j=1}^{m-1}(\prod_{i=j+1}^{m-1} G_i)F_j$$

Multiple Shooting I

0. Given m points $a = \tau_1 < \dots < \tau_m = b$ and m vectors $s_1^{(0)}, \dots, s_m^{(0)} \in \mathbb{R}^d$. Set $i = 0$.

1. Compute approximations $\eta^{(i,j)}(t)$, $1 \le j \le m-1$, to the solutions $y^{(i,j)}$ of the initial value problems
$y^{(i,j)'}(t) = f(t, y^{(i,j)}(t))$, $y^{(i,j)}(\tau_j) = s_j^{(i)}$ for $1 \le j \le m-1$.
Set $F_j^{(i)} = \eta^{(i,j)}(\tau_{j+1}) - s_{j+1}^{(i)}$ for $1 \le j \le m-1$
and $F_m^{(i)} = r(s_1^{(i)}, s_m^{(i)})$.

Multiple Shooting II

2. Using the same scheme and discrete times as in step **1**, compute approximations $\zeta^{(i,j)}(t)$ for the solutions $Z^{(i,j)}$ of the initial value problems
$Z^{(i,j)'}(t) = D_y f(t, \eta^{(i,j)}(t))Z^{(i,j)}(t)$, $Z^{(i,j)}(\tau_j) = I$ for $1 \le j \le m-1$.
Set $G_j^{(i)} = \zeta^{(i,j)}(\tau_{j+1})$ for $1 \le j \le m-1$
and $A^{(i)} = D_u r(s_1^{(i)}, s_m^{(i)})$,
$B^{(i)} = D_v r(s_1^{(i)}, s_m^{(i)})$.

Multiple Shooting III

3. Compute the matrix
$H^{(i)} = A^{(i)} + B^{(i)}G_{m-1}^{(i)} \cdot \dots \cdot G_1^{(i)}$
and the vector $\varphi^{(i)} = -F_m^{(i)} - B^{(i)}\sum_{j=1}^{m-1}(\prod_{l=j+1}^{m-1} G_l^{(i)})F_j^{(i)}$.
Solve the linear system of equations
$H^{(i)}\Delta s_1^{(i)} = \varphi^{(i)}$
and recursively compute the vectors
$\Delta s_{k+1}^{(i)} = G_k^{(i)}\Delta s_k^{(i)} + F_k^{(i)}$ for $1 \le k \le m-1$.
Set $s_k^{(i+1)} = s_k^{(i)} + \Delta s_k^{(i)}$ for $1 \le k \le m$, increase i by 1 and return to step **1**.

Properties

- With the same number of grid points on the total interval $[a, b]$, the initial value problems for the simple and multiple shooting require the same amount of work.
- The initial value problems on the sub-intervals can be solved in parallel .
- Lacking any further information, the intermediate points $\tau_1, \ldots, \tau_m$ may be chosen equidistant.

Finite Difference Methods

- Sturm-Liouville problem
- Difference quotients
- Difference discretization
- Properties

Sturm-Liouville Problem

- Given:
  - $p : [0,1] \to \mathbb{R}$ continuously differentiable function with $\underline{p} = \min_{0 \le x \le 1} p(x) > 0$
  - $q : [0,1] \to \mathbb{R}$ continuous function with $\underline{q} = \min_{0 \le x \le 1} q(x) > 0$
- Sought:

  Twice continuously differentiable function $u : [0,1] \to \mathbb{R}$ with
  $-(pu')' + qu = f$ in $(0,1)$ (differential equation)
  and $u(0) = 0$, $u(1) = 0$ (boundary condition)

Generalization

- Every Sturm-Liouville problem of the form
  $-(pu')' + qu = f$ in (a,b), $u(a) = \alpha$, $u(b) = \beta$
  can be transformed into an equivalent one with:
  $a = 0$, $b = 1$, $\alpha = 0$, $\beta = 0$.
- Look for a u of the form
  $u(x) = \alpha + \frac{\beta - \alpha}{b - a}(x - a) + v(\frac{x - a}{b - a})$
  with $v(0) = 0$, $v(1) = 0$
  and introduce a new variable by
  $t = \frac{x - a}{b - a}$.

Symmetric Difference Quotient

- The symmetric difference quotient is given by
  $\partial_h\varphi(x) = \frac{1}{h}\left[\varphi(x+\frac{h}{2}) - \varphi(x-\frac{h}{2})\right]$.
- Taylor's formula yields for every sufficiently differentiable function:
  $\partial_h\varphi(x) = \varphi'(x) + \frac{h^2}{24}\varphi'''(x+\theta h)$
  with a suitable $\theta \in (-\frac{1}{2}, \frac{1}{2})$.

Idea

- Replace derivatives by difference quotients ∂_h

$$\begin{aligned}
&\left(-(pu')'\right)(x)\\
&\approx \left(-\partial_h(pu')\right)(x)\\
&= \frac{1}{h}\left[p(x-\frac{h}{2})u'(x-\frac{h}{2}) - p(x+\frac{h}{2})u'(x+\frac{h}{2})\right]\\
&\approx \frac{1}{h}\left[p(x-\frac{h}{2})\partial_h u(x-\frac{h}{2}) - p(x+\frac{h}{2})\partial_h u(x+\frac{h}{2})\right]\\
&= \frac{1}{h^2}\left[p(x-\frac{h}{2})(u(x)-u(x-h)) - p(x+\frac{h}{2})(u(x+h)-u(x))\right]
\end{aligned}$$

- Impose the resulting equations only in grid points ih with $h = \frac{1}{n+1}$ and $1 \le i \le n$.

Finite Difference Discretization

- Choose a mesh size $h = \frac{1}{n+1}$.
- For $1 \le i \le n$ set $f_i = f(ih)$, $q_i = q(ih)$, $p_{i\pm\frac{1}{2}} = p(ih \pm \frac{h}{2})$
- Compute $u_0, \ldots, u_{n+1}$ such that
  $u_0 = 0$, $u_{n+1} = 0$
  and for $1 \le i \le n$
  $$f_i = -\frac{1}{h^2}p_{i-\frac{1}{2}}u_{i-1} + \left(\frac{1}{h^2}\left[p_{i-\frac{1}{2}} + p_{i+\frac{1}{2}}\right] + q_i\right)u_i - \frac{1}{h^2}p_{i+\frac{1}{2}}u_{i+1}$$
- Denote by u_h the continuous piecewise linear function which coincides at ih with u_i.

Properties

- The difference discretization gives rise to a linear system of equations with n equations for the n unknowns $u_1, \ldots, u_n$.
- The matrix is symmetric, positive definite and tridiagonal with positive diagonal elements and non-positive off-diagonal elements.
- The linear system admits a unique solution.
- The solution of the linear system with Gaussian elimination or Cholesky decomposition requires $O(n)$ operations.

Error Estimate

- Suppose that:
  - p is three times continuously differentiable.
  - The solution u of the Sturm-Liouville problem is four times continuously differentiable.
- Then the following error estimate is valid

  $\max_{0\le x\le 1} |u(x) - u_h(x)| \le ch^2$.
- The constant c depends on the lower bound $\underline{q}$ for q, the derivatives up to order 3 of p and the derivatives up to order 4 of u.

Variational Methods

- Basic idea
- Weak derivatives
- Sobolev spaces
- Finite element spaces
- Properties

Idea of the Variational Formulation

- Multiply the differential equation with a continuously differentiable function v with $v(0) = 0$, $v(1) = 0$

  $-(pu')'(x)v(x) + q(x)u(x)v(x) = f(x)v(x)$ for $0 \le x \le 1$.
- Integrate the result from 0 to 1

  $$\int_0^1 \left[-(pu')'(x)v(x) + q(x)u(x)v(x)\right]dx = \int_0^1 f(x)v(x)dx.$$
- Use integration by parts for the term containing derivatives

$$\begin{aligned} &-\int_0^1 (pu')'(x)v(x)dx \\ &= p(0)u'(0)v(0) - p(1)u'(1)v(1) + \int_0^1 p(x)u'(x)v'(x)dx \\ &= \int_0^1 p(x)u'(x)v'(x)dx. \end{aligned}$$

Problems

- The properties of the functions u and v must be stated more precisely to obtain a well-posed variational problem.
- Classical properties such as 'continuously differentiable' are too restrictive.
- The notion 'derivative' must be generalised.
- In view of the discrete problems, piecewise differentiable functions should be differentiable in the new weaker sense.

Weak Derivative

- Integration by parts yields for continuously differentiable functions u and v with $v(0) = 0$, $v(1) = 0$:

$$\int_0^1 u'(x)v(x)dx = u(1)v(1) - u(0)v(0) - \int_0^1 u(x)v'(x)dx.$$

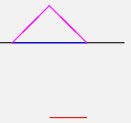
$$= -\int_0^1 u(x)v'(x)dx.$$

- The function u is said to be weakly differentiable with weak derivative w, if every continuously differentiable function v with $v(0) = 0$, $v(1) = 0$ satisfies

$$\int_0^1 w(x)v(x)dx = -\int_0^1 u(x)v'(x)dx.$$

Examples

- Every function which is continuously differentiable in the classical sense is weakly differentiable and its classical derivative coincides with the weak derivative.
- Every continuous piecewise differentiable function is weakly differentiable and its weak derivative is the piecewise classical derivative.
- $u(x) = 1 - |2x - 1|$ is weakly differentiable with weak derivative $w(x) = \begin{cases} 2 & \text{for } 0 < x < \frac{1}{2} \\ -2 & \text{for } \frac{1}{2} < x < 1 \end{cases}$.



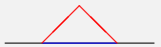
(Notice: The value $w(\frac{1}{2})$ is arbitrary.).

Sobolev Spaces

- $\|v\| = \left\{\int_0^1 |v(x)|^2 dx\right\}^{\frac{1}{2}}$ denotes the L^2-norm.
- $L^2(0,1)$ is the Lebesgue space of all functions v with finite L^2-norm $\|v\|$.
- $H^1(0,1)$ is the Sobolev space of all functions v in $L^2(0,1)$ which admit a weak derivative that is contained in $L^2(0,1)$.
- $H_0^1(0,1)$ is the Sobolev space of all functions v in $H^1(0,1)$ with $v(0) = 0$ und $v(1) = 0$.

Examples

- Every bounded function is in $L^2(0,1)$.
- $v(x) = \frac{1}{\sqrt{x}}$ is not in $L^2(0,1)$, since the integral of $\frac{1}{x} = v(x)^2$ is not finite.
- Every continuously differentiable function is in $H^1(0,1)$.
- Every continuous piecewise differentiable function is in $H^1(0,1)$.
- $v(x) = 1 - |2x - 1|$ is in $H_0^1(0,1)$.



- $v(x) = 2\sqrt{x}$ is not in $H^1(0,1)$, since the integral of $\frac{1}{x} = (v'(x))^2$ is not finite.
- Univariate functions in $H^1(0,1)$ are always continuous contrary to multivariate functions.

Variational Problem

Find $u \in H_0^1(0,1)$ such that for all $v \in H_0^1(0,1)$

$$\int_0^1 \left[p(x)u'(x)v'(x) + q(x)u(x)v(x)\right]dx = \int_0^1 f(x)v(x)dx.$$

Properties of the Variational Problem

- The variational problem admits a unique solution.
- The solution of the variational problem is the unique minimum in $H_0^1(0,1)$ of the energy function
$\frac{1}{2}\int_0^1 \left[p(x)u'(x)^2 + q(x)u(x)^2\right]dx - \int_0^1 f(x)u(x)dx.$

Finite Element Spaces

- $\mathcal{T}$ denotes an arbitrary partition of the interval $(0,1)$ into non-overlapping sub-intervals.

- $k \geq 1$ denotes an arbitrary polynomial degree.
- $S^{k,0}(\mathcal{T})$ is the finite element space of all continuous functions which are piecewise polynomials of degree at most k on the intervals in $\mathcal{T}$.
- $S_0^{k,0}(\mathcal{T})$ is the finite element space of all functions v in $S^{k,0}(\mathcal{T})$ with $v(0) = 0$ and $v(1) = 0$.

Finite Element Problem

Find $u_\mathcal{T} \in S_0^{k,0}(\mathcal{T})$ (trial function) such that for all $v_\mathcal{T} \in S_0^{k,0}(\mathcal{T})$ (test function)

$$\int_0^1 \left[p(x)u_\mathcal{T}'(x)v_\mathcal{T}'(x) + q(x)u_\mathcal{T}(x)v_\mathcal{T}(x)\right]dx = \int_0^1 f(x)v_\mathcal{T}(x)dx.$$

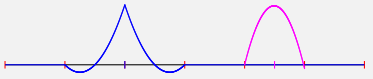
Properties of the Finite Element Problem

- The finite element problem admits a unique solution.
- The solution of the finite element problem is the unique minimum in $S_0^{k,0}(\mathcal{T})$ of the energy function.
- After choosing a basis for $S_0^{k,0}(\mathcal{T})$, the finite element problem amounts to a linear system of equations with $k \cdot \sharp\mathcal{T} - 1$ unknowns and a symmetric positive definite tridiagonal matrix (stiffness matrix).
- Standard choices of k are 1 (linear elements) or 2 (quadratic elements).
- One usually uses a nodal basis for $S_0^{k,0}(\mathcal{T})$.

Nodal Basis Functions

- Linear elements: Those functions which take the value 1 at exactly one endpoint of an interval and which vanish at all other endpoints of intervals.

- Quadratic elements: Those functions which take the value 1 at exactly one endpoint of an interval or midpoint of an interval and which vanish at all other endpoints and midpoints of intervals.



Error Estimates

- Denote by $h_\mathcal{T}$ the maximal length of the intervals in $\mathcal{T}$.
- Then the following error estimates hold for the solutions u of the variational problem and $u_\mathcal{T}$ of the finite element problem:

  $\|u' - u'_\mathcal{T}\| \leq c_1 h_\mathcal{T}$

  $\|u - u_\mathcal{T}\| \leq c_2 h_\mathcal{T}^2$
- The constants c_1 and c_2 only depend on the lower bound $\underline{p}$ for p, derivatives up to order 1 of p, the maximal value of q and derivatives up to order 2 of u.

Prerequisites for Finite Element and Finite Volume Methods

- Sobolev Spaces
- Finite Element Methods
- Finite Volume Methods

Sobolev Spaces

- Basic idea
- Integration by parts
- Weak derivatives
- Sobolev spaces
- Properties of Sobolev spaces
- Supplements

Reaction-Diffusion Equation

$$-\operatorname{div}(A\nabla u) + \alpha u = f \quad \text{in } \Omega$$
$$u = 0 \quad \text{on } \Gamma$$

- Ω a polyhedron in $\mathbb{R}^d$ with $d = 2$ or $d = 3$
- $A(x)$ a symmetric positive definite, $d \times d$ matrix for every x in Ω
- $\alpha(x)$ a non-negative number for every x in Ω

Divergence Theorem

- Divergence:
$$\operatorname{div} \mathbf{w} = \sum_{i=1}^{d} \frac{\partial w_i}{\partial x_i}$$
- Divergence Theorem:
$$\int_\Omega \operatorname{div} \mathbf{w} dx = \int_\Gamma \mathbf{w} \cdot \mathbf{n} dS$$

Integration by Parts in Several Dimensions I

- The divergence theorem applied to $\mathbf{w} = v(A\nabla u)$ yields
$$\int_\Omega v \operatorname{div}(A\nabla u) dx + \int_\Omega \nabla v \cdot A\nabla u dx$$
$$= \int_\Omega \operatorname{div}(vA\nabla u) dx = \int_\Omega \operatorname{div} \mathbf{w} dx = \int_\Gamma \mathbf{w} \cdot \mathbf{n} dS$$
$$= \int_\Gamma v\mathbf{n} \cdot A\nabla u dS.$$
- If $v = 0$ on Γ, this implies
$$\int_\Omega \nabla v \cdot A\nabla u dx = -\int_\Omega v \operatorname{div}(A\nabla u) dx.$$

Idea of the Variational Formulation

- Multiply the differential equation with a continuously differentiable function v with $v = 0$ on Γ
  $-\operatorname{div}(A\nabla u)(x)v(x) + \alpha(x)u(x)v(x) = f(x)v(x)$ for $x \in \Omega$.
- Integrate the result over Ω
  $\int_\Omega [-\operatorname{div}(A\nabla u)v + \alpha uv]dx = \int_\Omega fvdx.$
- Use integration by parts for the term containing derivatives
  $-\int_\Omega \operatorname{div}(A\nabla u)vdx = \int_\Omega \nabla v \cdot A\nabla u dx.$

Problems

- The properties of the functions u and v must be stated more precisely to obtain a well-posed variational problem.
- Classical properties such as 'continuously differentiable' are too restrictive.
- The notion 'derivative' must be generalised.
- In view of the discrete problems, piecewise differentiable functions should be differentiable in the new weaker sense.

Integration by Parts in Several Dimensions II

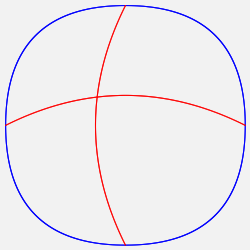
- The divergence theorem applied to $\mathbf{w} = uv\mathbf{e}_i$ ($\mathbf{e}_i$ i-th unit vector with i-th component 1 and vanishing remaining components) yields
  $$\int_\Omega \frac{\partial u}{\partial x_i} v dx + \int_\Omega u \frac{\partial v}{\partial x_i} dx$$
  $$= \int_\Omega \frac{\partial (uv)}{\partial x_i} dx = \int_\Omega \operatorname{div} \mathbf{w} dx = \int_\Gamma \mathbf{w} \cdot \mathbf{n} dS$$
  $$= \int_\Gamma uv\mathbf{n}_i dS.$$
- If $u = 0$ or $v = 0$ on Γ, this implies
  $\int_\Omega \frac{\partial u}{\partial x_i} v dx = -\int_\Omega u \frac{\partial v}{\partial x_i} dx.$

Weak Derivative

- The function u is said to be weakly differentiable w.r.t. x_i with weak derivative w_i, if every continuously differentiable function v with $v = 0$ on Γ satisfies
  $\int_\Omega w_i v dx = -\int_\Omega u \frac{\partial v}{\partial x_i} dx.$
- If u is weakly differentiable w.r.t. to all variables $x_1, \ldots, x_d$, we call u weakly differentiable and write ∇u for the vector $(w_1, \ldots, w_d)$ of the weak derivatives.

Examples

- Every function which is continuously differentiable in the classical sense is weakly differentiable and its classical derivative coincides with the weak derivative.
- Every continuous piecewise differentiable function is weakly differentiable and its weak derivative is the piecewise classical derivative.



Sobolev Spaces

- $\|v\| = \left\{ \int_\Omega |v|^2 dx \right\}^{\frac{1}{2}}$ denotes the L^2-norm.
- $L^2(\Omega)$ is the Lebesgue space of all functions v with finite L^2-norm $\|v\|$.
- $H^1(\Omega)$ is the Sobolev space of all functions v in $L^2(\Omega)$, which are weakly differentiable and for which $|\nabla v|$, the Euclidean norm of ∇v, is in $L^2(\Omega)$.
- $H_0^1(\Omega)$ is the Sobolev space of all functions v in $H^1(\Omega)$ with $v = 0$ on Γ.

Examples

- Every bounded function is in $L^2(\Omega)$.
- Every continuously differentiable function is in $H^1(\Omega)$.
- A piecewise differentiable function is in $H^1(\Omega)$, if and only if it is globally continuous.
- Functions in $H^1(\Omega)$ must not admit point values.

Example: Radially Symmetric Functions in $\mathbb{R}^2$

- Ω circle with radius 1 centred at the origin
- $v_\alpha(x,y) = \left(x^2+y^2\right)^{\frac{\alpha}{2}}$ with $\alpha \in \mathbb{R}$
- $\int_\Omega v_\alpha^2 dxdy = 2\pi \int_0^1 r^{2\alpha} r dr < \infty$
  $\iff 2\alpha + 1 > -1 \iff \alpha > -1$
- $\int_\Omega |\nabla v_\alpha|^2 dxdy = 2\pi \int_0^1 \alpha^2 r^{2\alpha-2} r dr < \infty$
  $\iff 2\alpha - 1 > -1 \iff \alpha > 0$
- $v_\alpha \in H^1(\Omega) \iff \alpha > 0$
- $v(x) = \ln(|\ln(\sqrt{x^2+y^2})|)$ is in $H^1(\Omega)$ but has no finite value at the origin.

Example: Radially Symmetric Functions in $\mathbb{R}^3$

- Ω ball with radius 1 centred at the origin
- $v_\alpha(x,y,z) = \left(x^2+y^2+z^2\right)^{\frac{\alpha}{2}}$ with $\alpha \in \mathbb{R}$
- $\int_\Omega v_\alpha^2 dxdydz = 4\pi \int_0^1 r^{2\alpha} r^2 dr < \infty$
  $\iff \quad 2\alpha + 2 > -1 \quad \iff \quad \alpha > -\frac{3}{2}$
- $\int_\Omega |\nabla v_\alpha|^2 dxdy = 4\pi \int_0^1 \alpha^2 r^{2\alpha-2} r^2 dr < \infty$
  $\iff \quad 2\alpha > -1 \quad \iff \quad \alpha > -\frac{1}{2}$
- $v_\alpha \in H^1(\Omega) \quad \iff \quad \alpha > -\frac{1}{2}$
- $v(x) = \left(x^2+y^2+y^2\right)^{-\frac{1}{8}}$ is in $H^1(\Omega)$ but has no finite value at the origin.

Variational Problem

Find $u \in H_0^1(\Omega)$ such that for all $v \in H_0^1(\Omega)$

$$\int_\Omega \left[\nabla v \cdot A \nabla u + \alpha u v\right] dx = \int_\Omega f v dx.$$

Properties of the Variational Problem

- The variational problem admits a unique solution.
- The solution of the variational problem is the unique minimum in $H_0^1(\Omega)$ of the energy function
  $\frac{1}{2}\int_\Omega \left[\nabla u \cdot A \nabla u + \alpha u^2\right] dx - \int_\Omega f u dx.$

Convective Derivatives

- A convective derivative $\mathbf{a} \cdot \nabla u$ gives rise to the additional term $\int_\Omega \mathbf{a} \cdot \nabla u v$ on the left-hand side of the variational problem.
- Then the solution of the variational problem cannot be interpreted as the minimum of an energy function.

Neumann Boundary Condition

- The boundary condition $\mathbf{n} \cdot A\nabla u = g$ on $\Gamma_N \subset \Gamma$ is called Neumann or natural boundary condition.
- It prescribes the flux or traction.
- It gives rise to the additional term $\int_{\Gamma_N} gv$ on the right-hand side of the variational problem.

Weak Divergence

- A vector-field $\mathbf{u} : \Omega \subset \mathbb{R}^d \to \mathbb{R}^d$ is said to have the weak divergence $w : \Omega \to \mathbb{R}$ if every continuously differentiable scalar function v satisfies

$$\int_\Omega wv = -\int_\Omega \mathbf{u} \cdot \nabla v$$

- If $\mathbf{u}$ has the weak divergence w, one writes $w = \operatorname{div} \mathbf{u}$.
- If $\mathbf{u}$ is continuously differentiable, it has a weak divergence which coincides with the classical divergence.

$H(\operatorname{div};\Omega)$

- $H(\operatorname{div};\Omega) = \{\mathbf{u} : \Omega \to \mathbb{R}^d : \mathbf{u} \in L^2(\Omega)^d \text{ and } \operatorname{div} \mathbf{u} \in L^2(\Omega)\}$
- A piecewise differentiable vector-field is in $H(\operatorname{div};\Omega)$, if and only if its normal component is continuous across interfaces.
- The space $H(\operatorname{div};\Omega)$ plays a crucial role in mixed formulations of linearized elasticity which avoid the locking phenomenon.

Finite Element Spaces

- Partitions
- Finite element spaces
- Local and global degrees of freedom
- Nodal basis functions
- Evaluation of the nodal basis functions
- Evaluation of integrals
- Supplements

Reaction-Diffusion Equation

Find $u \in H_0^1(\Omega)$ such that for all $v \in H_0^1(\Omega)$

$$\int_\Omega \left[\nabla v \cdot A \nabla u + \alpha u v\right] dx = \int_\Omega f v dx.$$

Basic Idea

- Subdivide Ω into non-overlapping simple sub-domains called elements such as triangles, parallelograms, tetrahedra of parallelepipeds, ... (partition).
- In the variational problem replace the space $H_0^1(\Omega)$ by a finite dimensional subspace consisting of continuous functions which are element-wise polynomials (finite element space).
- This gives rise to a linear system of equations for the approximation $u_\mathcal{T}$ of the solution u of the differential equation.

Partition

$\mathcal{T} = \{K_i : 1 \le i \le N_\mathcal{T}\}$ denotes a partition of Ω with the following properties:

- Ω is the union of all elements K in $\mathcal{T}$.
- Admissibility: Any two elements K and K' in $\mathcal{T}$ are either disjoint or share a vertex or a complete edge or, if $d = 3$, a complete face.

  admissible  not admissible 
- Affine equivalence: Every element K is a triangle or parallelogram, if $d = 2$, or a tetrahedron or parallelepiped, if $d = 3$.

Remarks

- Curved boundaries can be approximated by piecewise straight lines or planes.
- The admissibility is necessary to ensure the continuity of the finite element functions and thus the inclusion of the finite element spaces in $H_0^1(\Omega)$.
- If the admissibility is violated, the continuity of the finite element functions must be enforced which leads to a more complicated implementation.
- Partitions can also consist of general quadrilaterals or hexahedra which leads to a more complicated implementation.

Finite Element Spaces

- $R_k(\widehat{K}) = \begin{cases} \text{span}\{x_1^{\alpha_1} \cdot \ldots \cdot x_d^{\alpha_d} : \alpha_1 + \ldots + \alpha_d \leq k\} \\ \widehat{K} \text{ reference simplex} \\ \text{span}\{x_1^{\alpha_1} \cdot \ldots \cdot x_d^{\alpha_d} : \max\{\alpha_1, \ldots, \alpha_d\} \leq k\} \\ \widehat{K} \text{ reference cube} \end{cases}$
- $R_k(K) = \{\widehat{p} \circ F_K^{-1} : \widehat{p} \in \widehat{R}_k\}$
- $S^{k,-1}(\mathcal{T}) = \{v : \Omega \to \mathbb{R} : v\big|_K \in R_k(K) \text{ for all } K \in \mathcal{T}\}$
- $S^{k,0}(\mathcal{T}) = S^{k,-1}(\mathcal{T}) \cap C(\overline{\Omega})$
- $S_0^{k,0}(\mathcal{T}) = S^{k,0}(\mathcal{T}) \cap H_0^1(\Omega)$
  $= \{v \in S^{k,0}(\mathcal{T}) : v = 0 \text{ on } \Gamma\}$

Remarks

- The global continuity ensures that $S^{k,0}(\mathcal{T}) \subset H^1(\Omega)$.
- The polynomial degree k may vary from element to element; this leads to a more complicated implementation.

Discrete Problem

Find $u_\mathcal{T} \in S_0^{k,0}(\mathcal{T})$ (trial function) such that for all $v_\mathcal{T} \in S_0^{k,0}(\mathcal{T})$ (test function)

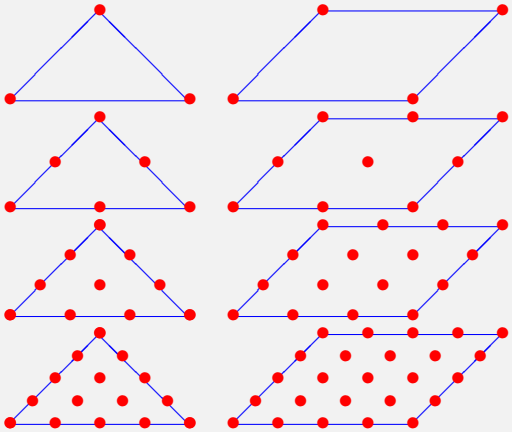
$$\int_\Omega \left[\nabla v_\mathcal{T} \cdot A \nabla u_\mathcal{T} + \alpha u_\mathcal{T} v_\mathcal{T}\right] dx = \int_\Omega f v_\mathcal{T} dx.$$

Properties of the Discrete Problem

- The discrete problem admits a unique solution.
- The solution of the discrete problem is the unique minimum in $S_0^{k,0}(\mathcal{T})$ of the energy function $\frac{1}{2}\int_\Omega \left[\nabla u \cdot A \nabla u + \alpha u^2\right] dx - \int_\Omega f u dx$.
- After choosing a basis for $S_0^{k,0}(\mathcal{T})$ the discrete problem amounts to a linear system of equations with $\approx k^d N_\mathcal{T}$ ($N_\mathcal{T} = \sharp\mathcal{T}$) equations and unknowns.

Element-Wise Degrees of Freedom $\mathcal{N}_{K,k}$

- $k = 1$
- $k = 2$
- $k = 3$
- $k = 4$

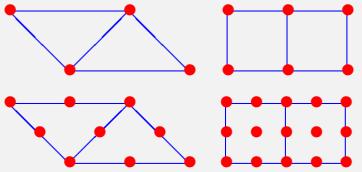


Global Degrees of Freedom $\mathcal{N}_{\mathcal{T},k}$

- $\mathcal{N}_{\mathcal{T},k} = \bigcup_{K \in \mathcal{T}} \mathcal{N}_{K,k}$

$k = 1$

$k = 2$

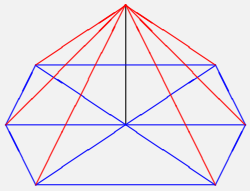


- The functions in $S^{k,0}(\mathcal{T})$ are uniquely defined by their values in $\mathcal{N}_{\mathcal{T},k}$ thanks to the admissibility of $\mathcal{T}$.

Nodal Basis Functions

The nodal basis function associated with a vertex $z \in \mathcal{N}_{\mathcal{T},k}$ is uniquely defined by he conditions

- $\lambda_{z,k} \in S^{k,0}(\mathcal{T})$,
- $\lambda_{z,k}(z) = 1$,
- $\lambda_{z,k}(y) = 0$ for all $y \in \mathcal{N}_{\mathcal{T},k} \setminus \{z\}$.



Properties

- $\{\lambda_{z,k} : z \in \mathcal{N}_{\mathcal{T},k}\}$ is a basis for $S^{k,0}(\mathcal{T})$.
- $\{\lambda_{z,k} : z \in \mathcal{N}_{\mathcal{T},k} \setminus \Gamma\}$ is a basis for $S_0^{k,0}(\mathcal{T})$.
  (Degrees of freedom on the boundary Γ are suppressed.)
- $\lambda_{z,k}$ vanishes outside the union of all elements that share the vertex z.
- The stiffness matrix is sparse.

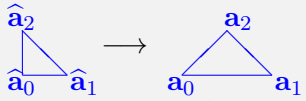
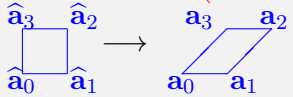
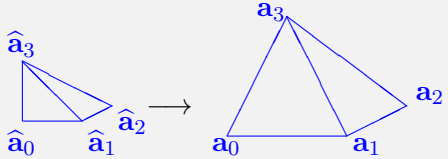
Evaluation of the Nodal Basis Functions by Transformation to a Reference Element

- Reference elements $\widehat{K}$ 
- Determine the nodal basis functions $\widehat{\lambda}_{\widehat{z},k}$ for the reference element $\widehat{K}$.
- Determine an affine transformation of the reference element $\widehat{K}$ onto the current element K
  $\widehat{K} \ni \widehat{x} \mapsto x = b_K + B_K \widehat{x} \in K$.
- Express $\lambda_{z,k}$ in terms of $\widehat{\lambda}_{\widehat{z},k}$ using the affine transformation
  $\lambda_{z,k}(x) = \widehat{\lambda}_{\widehat{z},k}(\widehat{x})$.

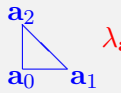
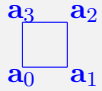
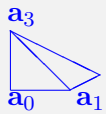
Examples for $\widehat{\lambda}_{\widehat{z},k}$

- Reference triangle ◺
  - $k = 1$ Vertices $1 - x - y$, x, y
  - $k = 2$
    Vertices $(1 - x - y)(1 - 2x - 2y)$, $x(2x - 1)$, $y(2y - 1)$
    Midpoints of edges $4x(1 - x - y)$, $4xy$, $4y(1 - x - y)$
- Reference square □
  - $k = 1$ Vertices $(1 - x)(1 - y)$, $x(1 - y)$, xy, $(1 - x)y$
  - $k = 2$
    Vertices $(1 - 2x)(1 - x)(1 - 2y)(1 - y)$,
    $x(2x - 1)(1 - 2y)(1 - y)$, $x(2x - 1)y(2y - 1)$,
    $(1 - 2x)(1 - x)y(2y - 1)$
    Midpoints of edges $4x(1 - x)(1 - y)(1 - 2y)$,
    $4x(2x - 1)y(1 - y)$,
    $4x(1 - x)y(2y - 1)$, $4y(1 - y)(1 - 2x)(1 - x)$
    Barycentre $16x(1 - x)y(1 - y)$

Examples for Affine Transformations

- 
  $b_K = \mathbf{a}_0$, $B_K = (\mathbf{a}_1 - \mathbf{a}_0\,,\, \mathbf{a}_2 - \mathbf{a}_0)$
- 
  $b_K = \mathbf{a}_0$, $B_K = (\mathbf{a}_1 - \mathbf{a}_0\,,\, \mathbf{a}_3 - \mathbf{a}_0)$
- 
  $b_K = \mathbf{a}_0$, $B_K = (\mathbf{a}_1 - \mathbf{a}_0\,,\, \mathbf{a}_2 - \mathbf{a}_0\,,\, \mathbf{a}_3 - \mathbf{a}_0)$
- Similar formulae hold for parallelepipeds.

Evaluation Using the Element Geometry ($k = 1$)

- 
  $\lambda_{\mathbf{a}_i,1}(x) = \dfrac{\det(x - \mathbf{a}_{i+1}\,,\, \mathbf{a}_{i+2} - \mathbf{a}_{i+1})}{\det(\mathbf{a}_i - \mathbf{a}_{i+1}\,,\, \mathbf{a}_{i+2} - \mathbf{a}_{i+1})}$
- 
  $\lambda_{\mathbf{a}_i,1}(x) = \dfrac{\det(x - \mathbf{a}_{i+2}\,,\, \mathbf{a}_{i+3} - \mathbf{a}_{i+2})}{\det(\mathbf{a}_i - \mathbf{a}_{i+2}\,,\, \mathbf{a}_{i+3} - \mathbf{a}_{i+2})} \cdot \dfrac{\det(x - \mathbf{a}_{i+2}\,,\, \mathbf{a}_{i+1} - \mathbf{a}_{i+2})}{\det(\mathbf{a}_i - \mathbf{a}_{i+2}\,,\, \mathbf{a}_{i+1} - \mathbf{a}_{i+2})}$
- 
  $\lambda_{\mathbf{a}_i,1}(x) = \dfrac{\det(x - \mathbf{a}_{i+1}, \mathbf{a}_{i+2} - \mathbf{a}_{i+1}, \mathbf{a}_{i+3} - \mathbf{a}_{i+1})}{\det(\mathbf{a}_i - \mathbf{a}_{i+1}, \mathbf{a}_{i+2} - \mathbf{a}_{i+1}, \mathbf{a}_{i+3} - \mathbf{a}_{i+1})}$
- Parallelepipeds similarly with 3 factors corresponding to 3 tetrahedra
- All indices must be taken modulo the number of element vertices.

Evaluation Using the Element Geometry ($k \geq 2$)

- Every $\lambda_{z,k}$ can be represented as a suitable product of first order nodal basis functions $\lambda_{\mathbf{a}_i,1}$ associated with the element vertices.
- Example: triangle, $k = 2$
  - Vertex $\mathbf{a}_i$
    $\lambda_{\mathbf{a}_i,2} = \lambda_{\mathbf{a}_i}[\lambda_{\mathbf{a}_i} - \lambda_{\mathbf{a}_{i+1}} - \lambda_{\mathbf{a}_{i+2}}]$
  - Midpoint z of the edge with endpoints $\mathbf{a}_i$ und $\mathbf{a}_{i+1}$
    $\lambda_{z,2} = 4\lambda_{\mathbf{a}_i}\lambda_{\mathbf{a}_{i+1}}$
- Example: parallelogram, $k = 2$
  - Vertex $\mathbf{a}_i$
    $\lambda_{\mathbf{a}_i,2} = \lambda_{\mathbf{a}_i}[\lambda_{\mathbf{a}_i} - \lambda_{\mathbf{a}_{i+1}} + \lambda_{\mathbf{a}_{i+2}} - \lambda_{\mathbf{a}_{i+3}}]$
  - Midpoint z of the edge with endpoints $\mathbf{a}_i$ und $\mathbf{a}_{i+1}$
    $\lambda_{z,2} = 4\lambda_{\mathbf{a}_i}[\lambda_{\mathbf{a}_{i+1}} - \lambda_{\mathbf{a}_{i+2}}]$
  - Barycentre z
    $\lambda_{z,2} = 16\lambda_{\mathbf{a}_0}\lambda_{\mathbf{a}_2}$

Evaluation of Integrals

- The exact evaluation of the integrals appearing in the entries of the stiffness matrix and load vector often is too expensive or even impossible.
- The integrals are therefore approximately evaluated using a suitable quadrature formula:
  $$\int_K \varphi dx \approx Q_k(\varphi) = \sum_{q \in \mathcal{Q}_K} c_q \varphi(q).$$
- In order to avoid that this spoils the accuracy of the finite element discretization, the quadrature formula must have the order $2k-2$ (k element degree):
  $$\int_K \varphi dx = Q_K(\varphi) \text{ for all } \varphi \in R_{2k-2}(K).$$
- Order 0 is sufficient for linear elements; order 2 is sufficient for quadratic elements.

Examples of Quadrature Formulae

- Triangle:
  - order 1:
    - $\mathcal{Q}_K$ barycentre of K,
    - $c_q = |K|$
  - order 2:
    - $\mathcal{Q}_K$ midpoints of edges of K,
    - $c_q = \frac{1}{3}|K|$ for all q
- Parallelogram:
  - order 1:
    - $\mathcal{Q}_K$ barycentre of K,
    - $c_q = |K|$
  - order 3:
    - $\mathcal{Q}_K$ vertices, midpoints of edges and barycentre of K,
    - $c_q = \begin{cases} \frac{1}{36}|K| & \text{if } q \text{ is a vertex} \\ \frac{4}{36}|K| & \text{if } q \text{ is a midpoint of an edge} \\ \frac{16}{36}|K| & \text{if } q \text{ is the barycentre} \end{cases}$

Neumann Boundary Condition

- The Neumann boundary condition $\mathbf{n} \cdot A\nabla u = g$ on $\Gamma_N \subset \Gamma$ gives rise to
  - an additional term $\int_{\Gamma_N} gv dS$ on the right-hand side of the variational problem,
  - an additional term $\int_{\Gamma_N} gv_\mathcal{T} dS$ on the right-hand side of the discrete problem.
- The additional entries of the load vector are taken into account when sweeping through the elements.
- Degrees of freedom associated with points on the Neumann boundary Γ_N are additional unknowns.

Convective Derivatives

- Convective derivatives lead to a non-symmetric stiffness matrix.
- They often give rise to unphysical oscillations of the numerical solution.
- To avoid these oscillations special modifications such as upwinding or streamline Petrov-Galerkin stabilization must be introduced.

Finite Volume Methods

- Systems in divergence form
- Finite volume discretization
- Finite volume meshes
- Numerical fluxes
- Relation to finite element methods

Systems in Divergence Form

- Domain: $\Omega \subset \mathbb{R}^d$
- Source: $\mathbf{g} : \mathbb{R}^m \times \Omega \times (0,\infty) \to \mathbb{R}^m$
- Mass: $\mathbf{M} : \mathbb{R}^m \to \mathbb{R}^m$
- Flux: $\underline{\mathbf{F}} : \mathbb{R}^m \to \mathbb{R}^{m\times d}$
- Initial value: $\mathbf{U}_0 : \Omega \to \mathbb{R}^m$
- Problem: Find $\mathbf{U} : \Omega \times (0,\infty) \to \mathbb{R}^m$ such that under suitable boundary conditions

$$\begin{aligned} \frac{\partial \mathbf{M}(\mathbf{U})}{\partial t} + \operatorname{div} \underline{\mathbf{F}}(\mathbf{U}) &= \mathbf{g}(\mathbf{U}, x, t) && \text{in } \Omega \times (0,\infty) \\ \mathbf{U}(\cdot, 0) &= \mathbf{U}_0 && \text{in } \Omega \end{aligned}$$

- $\operatorname{div} \underline{\mathbf{F}}(\mathbf{U}) = \Big(\sum_{j=1}^{d} \frac{\partial \underline{\mathbf{F}}(\mathbf{U})_{i,j}}{\partial x_j} \Big)_{1\le i\le m}$

Advective and Viscous Fluxes

- The flux $\underline{\mathbf{F}}$ splits into two components:
  $\underline{\mathbf{F}} = \underline{\mathbf{F}}_{\text{adv}} + \underline{\mathbf{F}}_{\text{visc}}$.
- $\underline{\mathbf{F}}_{\text{adv}}$ is called advective flux and does not contain any derivatives.
- $\underline{\mathbf{F}}_{\text{visc}}$ is called viscous flux and contains spatial derivatives.
- The advective flux models transport or convection phenomena.
- The viscous flux models diffusion phenomena.

Examples

- Linear parabolic equations of 2nd order:
  - $\frac{\partial u}{\partial t} - \operatorname{div}(A\nabla u) + \mathbf{a}\cdot\nabla u + \alpha u = f$
  - $m = 1$
  - $\mathbf{U} = u$
  - $\mathbf{M}(\mathbf{U}) = u$
  - $\underline{\mathbf{F}}_{\text{adv}}(\mathbf{U}) = \mathbf{a}u$
  - $\underline{\mathbf{F}}_{\text{visc}}(\mathbf{U}) = -A\nabla u$
  - $\mathbf{g}(\mathbf{U}) = f - \alpha u + (\operatorname{div}\mathbf{a})u$
- Euler equations
- Compressible Navier-Stokes equations
- Burger's equation

Finite Volume Discretization

First Step

- Choose a time step $\tau > 0$.
- Choose a partition $\mathcal{T}$ of Ω into arbitrary non-overlapping polyhedra.
- Fix $n \in \mathbb{N}^*$ and $K \in \mathcal{T}$.
- Integrate the system over $K \times [(n-1)\tau, n\tau]$:

$$\int_{(n-1)\tau}^{n\tau}\int_K \frac{\partial \mathbf{M}(\mathbf{U})}{\partial t}dxdt + \int_{(n-1)\tau}^{n\tau}\int_K \operatorname{div}\underline{\mathbf{F}}(\mathbf{U})dxdt = \int_{(n-1)\tau}^{n\tau}\int_K \mathbf{g}(\mathbf{U},x,t)dxdt$$

Finite Volume Discretization

Second Step

Apply integration by parts to the terms on the left-hand side:

$$\int_{(n-1)\tau}^{n\tau}\int_K \frac{\partial \mathbf{M}(\mathbf{U})}{\partial t}dxdt = \int_K \mathbf{M}(\mathbf{U}(x,n\tau))dx - \int_K \mathbf{M}(\mathbf{U}(x,(n-1)\tau))dx$$

$$\int_{(n-1)\tau}^{n\tau}\int_K \operatorname{div}\underline{\mathbf{F}}(\mathbf{U})dxdt = \int_{(n-1)\tau}^{n\tau}\int_{\partial K} \underline{\mathbf{F}}(\mathbf{U})\cdot\mathbf{n}_K dSdt$$

Finite Volume Discretization

Third Step

- Assume that $\mathbf{U}$ is piecewise constant w.r.t space and time.
- Denote by $\mathbf{U}_K^n$ and $\mathbf{U}_K^{n-1}$ the value of $\mathbf{U}$ on K at times $n\tau$ and $(n-1)\tau$:

$$\int_K \mathbf{M}(\mathbf{U}(x,n\tau))dx \approx |K|\mathbf{M}(\mathbf{U}_K^n)$$

$$\int_K \mathbf{M}(\mathbf{U}(x,(n-1)\tau))dx \approx |K|\mathbf{M}(\mathbf{U}_K^{n-1})$$

$$\int_{(n-1)\tau}^{n\tau}\int_{\partial K} \underline{\mathbf{F}}(\mathbf{U})\cdot\mathbf{n}_K dSdt \approx \tau\int_{\partial K} \underline{\mathbf{F}}(\mathbf{U}_K^{n-1})\cdot\mathbf{n}_K dS$$

$$\int_{(n-1)\tau}^{n\tau}\int_K \mathbf{g}(\mathbf{U},x,t)dxdt \approx \tau|K|\mathbf{g}(\mathbf{U}_K^{n-1},x_K,(n-1)\tau)$$

Finite Volume Discretization

Fourth Step

Approximate the boundary integral for the flux by a numerical flux:

$$\tau \int_{\partial K} \underline{\mathbf{F}}(\mathbf{U}_K^{n-1}) \cdot \mathbf{n}_K dS$$
$$\approx \tau \sum_{\substack{K' \in \mathcal{T} \\ \partial K \cap \partial K' \in \mathcal{E}}} |\partial K \cap \partial K'| \mathbf{F}_{\mathcal{T}}(\mathbf{U}_K^{n-1}, \mathbf{U}_{K'}^{n-1})$$

Resulting Finite Volume Method

- For every element $K \in \mathcal{T}$ compute

$$\mathbf{U}_K^0 = \frac{1}{|K|} \int_K \mathbf{U}_0(x).$$

- For $n = 1, 2, \ldots$ successively compute for every element $K \in \mathcal{T}$

$$\mathbf{M}(\mathbf{U}_K^n) = \mathbf{M}(\mathbf{U}_K^{n-1}) - \tau \sum_{\substack{K' \in \mathcal{T} \\ \partial K \cap \partial K' \in \mathcal{E}}} \frac{|\partial K \cap \partial K'|}{|K|} \mathbf{F}_{\mathcal{T}}(\mathbf{U}_K^{n-1}, \mathbf{U}_{K'}^{n-1}) + \tau \mathbf{g}(\mathbf{U}_K^{n-1}, x_K, (n-1)\tau).$$

Possible Modifications

- The time step may be variable.
- The partition of Ω may differ from time step to time step.
- The approximation of $\mathbf{U}_K^n$ may not be constant.

Open Tasks

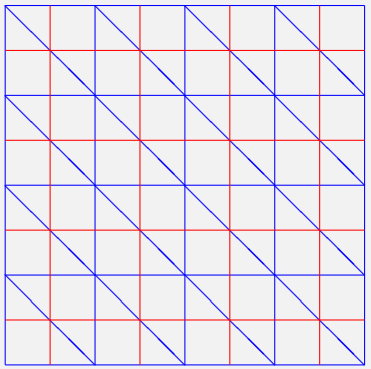
- Construct the partition $\mathcal{T}$.
- Construct the numerical flux $\underline{\mathbf{F}}_{\mathcal{T}}$.
- Take boundary conditions into account.

Construction of the Partition

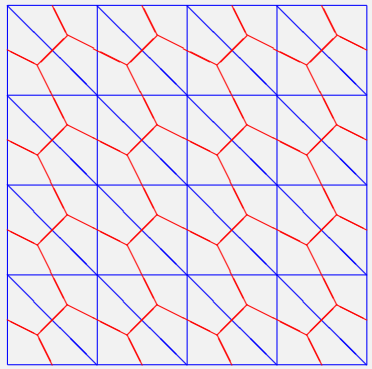
- Often the partition $\mathcal{T}$ is constructed as a dual mesh corresponding to an admissible primal finite element mesh $\widetilde{\mathcal{T}}$.
- In two space dimensions ($d = 2$) there are two major approaches for the construction of dual meshes:
  - For every element $\widetilde{K} \in \widetilde{\mathcal{T}}$ draw the perpendicular bisectors.
  - Connect the barycentre of every element $\widetilde{K} \in \widetilde{\mathcal{T}}$ with the midpoints of its edges.

Perpendicular Bisectors and Barycentres

Perpendicular Bisectors

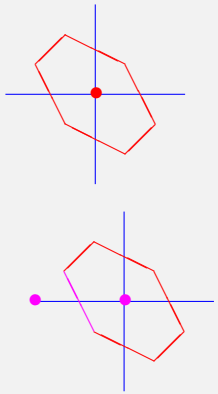


Barycentres



Properties of Dual Meshes

- Every element in $K \in \mathcal{T}$ corresponds to an element vertex x_K of $\widetilde{\mathcal{T}}$ and vice versa.
- For every edge E of $\mathcal{T}$ there are two element vertices $x_{E,1}$, $x_{E,2}$ of $\widetilde{\mathcal{T}}$ such that the line segment $\overline{x_{E,1}\, x_{E,2}}$ intersects E.



Advantages and Disadvantages of Perpendicular Bisectors

- The line segment $\overline{x_{E,1}\, x_{E,2}}$ and the edge E are perpendicular.
- The perpendicular bisectors of a triangle may intersect in a point outside of the triangle. The intersection of the perpendicular bisectors is inside the triangle, if and only if the triangle is acute.
- The perpendicular bisectors of a quadrilateral may not intersect at all. The perpendicular bisectors of a quadrilateral intersect in a common point, if and only if the quadrilateral is a rectangle.
- The construction with perpendicular bisectors is restricted to two space dimensions.

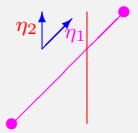
Construction of the Numerical Fluxes

Notations and Assumptions

- Assume that $\mathcal{T}$ is a dual mesh corresponding to a primal finite element mesh $\widetilde{\mathcal{T}}$.
- For every edge or face E of $\mathcal{T}$ denote by
  - K_1 and K_2 the adjacent volumes,
  - $\mathbf{U}_1$, $\mathbf{U}_2$ the values $\mathbf{U}_{K_1}^{n-1}$ and $\mathbf{U}_{K_2}^{n-1}$,
  - x_1, x_2 the element vertices in $\widetilde{\mathcal{T}}$ such that the line segment $\overline{x_1\, x_2}$ intersects E.
- Split the numerical flux $\underline{\mathbf{F}}_{\mathcal{T}}(\mathbf{U}_1, \mathbf{U}_2)$ into a viscous numerical flux $\underline{\mathbf{F}}_{\mathcal{T},\mathrm{visc}}(\mathbf{U}_1, \mathbf{U}_2)$ and an advective numerical flux $\underline{\mathbf{F}}_{\mathcal{T},\mathrm{adv}}(\mathbf{U}_1, \mathbf{U}_2)$.

Approximation of Viscous Fluxes

- Introduce a local coordinate system $\eta_1, \ldots, \eta_d$ such that η_1 is parallel to $\overline{x_1\, x_2}$ and such that the remaining coordinates are tangential to E.



- Express all derivatives in $\underline{\mathbf{F}}_{\mathrm{visc}}$ in terms of derivatives corresponding to the new coordinate system.
- Suppress all derivatives except those corresponding to η_1.
- Replace derivatives corresponding to η_1 by difference quotients of the form $\frac{\varphi_1-\varphi_2}{|x_1-x_2|}$.

Spectral Decomposition of Advective Fluxes

- Denote by $C(\mathbf{V}) = D(\underline{\mathbf{F}}_{\mathrm{adv}}(\mathbf{V}) \cdot \mathbf{n}_{K_1}) \in \mathbb{R}^{m\times m}$ the derivative of $\underline{\mathbf{F}}_{\mathrm{adv}}(\mathbf{V}) \cdot \mathbf{n}_{K_1}$ w.r.t. $\mathbf{V}$.
- Assume that this matrix can be diagonalized (Euler and Navier-Stokes equations fulfil this assumption.)

$$Q(\mathbf{V})^{-1} C(\mathbf{V}) Q(\mathbf{V}) = \Delta(\mathbf{V})$$

with an invertible matrix $Q(\mathbf{V}) \in \mathbb{R}^{m\times m}$ and a diagonal matrix $\Delta(\mathbf{V}) \in \mathbb{R}^{m\times m}$.

- Set $z^+ = \max\{z, 0\}$, $z^- = \min\{z, 0\}$ and

$$\Delta(\mathbf{V})^{\pm} = \mathrm{diag}\big(\Delta(\mathbf{V})_{11}^{\pm}, \ldots, \Delta(\mathbf{V})_{mm}^{\pm}\big),$$
$$C(\mathbf{V})^{\pm} = Q(\mathbf{V})\Delta(\mathbf{V})^{\pm} Q(\mathbf{V})^{-1}.$$

Approximation of Advective Fluxes

- Steger-Warming

$$\mathbf{F}_{\mathcal{T},\mathrm{adv}}(\mathbf{U}_1, \mathbf{U}_2) = C(\mathbf{U}_1)^+ \mathbf{U}_1 + C(\mathbf{U}_2)^- \mathbf{U}_2$$

- van Leer

$$\begin{aligned}
&\mathbf{F}_{\mathcal{T},\mathrm{adv}}(\mathbf{U}_1, \mathbf{U}_2) \\
&= \Big[\frac{1}{2}C(\mathbf{U}_1) + C\big(\frac{1}{2}(\mathbf{U}_1+\mathbf{U}_2)\big)^+ - C\big(\frac{1}{2}(\mathbf{U}_1+\mathbf{U}_2)\big)^-\Big]\mathbf{U}_1 \\
&\quad + \Big[\frac{1}{2}C(\mathbf{U}_2) - C\big(\frac{1}{2}(\mathbf{U}_1+\mathbf{U}_2)\big)^+ + C\big(\frac{1}{2}(\mathbf{U}_1+\mathbf{U}_2)\big)^-\Big]\mathbf{U}_2
\end{aligned}$$

Properties

- Both approximations require the computation of $D\underline{\mathbf{F}}_{\text{adv}}(\mathbf{V}) \cdot \mathbf{n}_{K_1}$ together with its eigenvalues and eigenvectors for suitable values of $\mathbf{V}$.
- The approach of van Leer usually is more costly than the one of Steger-Warming since it requires three evaluations of $C(\mathbf{V})$ instead of two.
- This extra cost can be avoided for the Euler and Navier-Stokes equations since these have the particular structure $\underline{\mathbf{F}}_{\text{adv}}(\mathbf{V}) \cdot \mathbf{n}_{K_1} = C(\mathbf{V})\mathbf{V}$.

A One-Dimensional Example

- Burger's equation: $\dfrac{\partial u}{\partial t} + u\dfrac{\partial u}{\partial x} = 0$
- $\underline{\mathbf{F}}_{\text{adv}}(u) = \frac{1}{2}u^2$, $C(u) = u$, $C(u)^{\pm} = u^{\pm}$
- Steger-Warming:

$$\underline{\mathbf{F}}_{\mathcal{T},\text{adv}}(u_1, u_2) = \begin{cases} u_1^2 & \text{if } u_1 \geq 0, u_2 \geq 0 \\ u_1^2 + u_2^2 & \text{if } u_1 \geq 0, u_2 \leq 0 \\ u_2^2 & \text{if } u_1 \leq 0, u_2 \leq 0 \\ 0 & \text{if } u_1 \leq 0, u_2 \geq 0 \end{cases}$$

- van Leer:

$$\underline{\mathbf{F}}_{\mathcal{T},\text{adv}}(u_1, u_2) = \begin{cases} u_1^2 & \text{if } u_1 \geq -u_2 \\ u_2^2 & \text{if } u_1 \leq -u_2 \end{cases}$$

TVD and ENO Schemes

- The convergence analysis of finite volume methods is based on compactness arguments, in particular the concept of compensated compactness.
- This requires to bound the total variation of the numerical approximation and to avoid unphysical oscillations.
- This leads to the concept of total variation diminishing TVD and essentially non-oscillating ENO schemes.
- Corresponding material may be found under the names of Enquvist, LeVeque, Osher, Roe, Tadmor, ...

Relation to Finite Element Methods

- Suppose that $\mathcal{T}$ is a dual mesh corresponding to a primal finite element mesh $\widetilde{\mathcal{T}}$.
- Then there is a one-to-one correspondence between piecewise constant functions associated with $\mathcal{T}$ and continuous piecewise linear functions associated with $\widetilde{\mathcal{T}}$:

$$S^{0,-1}(\mathcal{T})^m \ni \mathbf{U}_{\mathcal{T}} \leftrightarrow \widetilde{\mathbf{U}}_{\widetilde{\mathcal{T}}} \in S^{1,0}(\widetilde{\mathcal{T}})^m$$
$$\mathbf{U}_{\mathcal{T}}|_K = \widetilde{\mathbf{U}}_{\widetilde{\mathcal{T}}}(x_K) \qquad \text{for all } K \in \mathcal{T}.$$

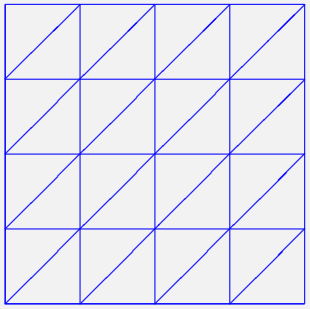
Efficient Solvers for Linear Systems of Equations

- Properties of Direct and Iterative Solvers
- Classical Iterative Solvers
- Conjugate Gradient Methods
- Multigrid Methods
- Indefinite Problems

Properties of Direct and Iterative Solvers

- A typical model problem
- Properties of the stiffness matrix
- Consequences for direct and iterative solvers

A Typical Model Problem



- Poisson equation
  $-\Delta u = f$ in Ω, $u = 0$ on Γ
- $\Omega = (0,1)^2$
- Courant triangulation
  consisting of $2n^2$ isosceles right-angled triangles with short sides of length $h = n^{-1}$
- Linear finite elements
- Number N of unknowns is of order $n^2 = h^{-2}$.

Properties of the Stiffness Matrix

- It is symmetric positive definite.
- It has 5 non-zero elements per row.
- It has bandwidth $h^{-1} \approx N^{\frac{1}{2}}$.
- Gaussian elimination requires N^2 operations.
- A matrix-vector multiplication requires $5N$ operations.
- Its smallest eigenvalue is of order 1.
- Its largest eigenvalue is of order $h^{-2} \approx N$.

Typical Properties of Direct Solvers

- They require $O(N^{2-\frac{1}{d}})$ storage for a discrete problem with N unknowns in d space dimensions.
- They require $O(N^{3-\frac{2}{d}})$ operations.
- They yield the exact solution of the discrete problem up to rounding errors.
- They yield an approximation for the differential equation with an $O(h^\alpha) = O(N^{-\frac{\alpha}{d}})$ error (typically: $\alpha \in \{1, 2\}$).

Typical Properties of Classical Iterative Solvers

- They require $O(N)$ storage.
- They require $O(N)$ operations per iteration.
- Their convergence rate deteriorates with an increasing condition number of the discrete problem which usually is $O(h^{-2}) = O(N^{\frac{2}{d}})$.
- In order to reduce an initial error by a factor 0.1 one usually needs the following numbers of operations:
  - $O(N^{1+\frac{2}{d}})$ with the Gauß-Seidel algorithm,
  - $O(N^{1+\frac{1}{d}})$ with the conjugate gradient (CG-) algorithm,
  - $O(N^{1+\frac{1}{2d}})$ with the CG-algorithm with Gauß-Seidel preconditioning.

Comparison of Solvers

Arithmetic Operations

Example: Linear finite elements on a Courant triangulation for the Poisson equation in the unit square; initial error is reduced by the factor 0.05

h	Gaussian el.	GS	CG	PCG	MG
$\frac{1}{16}$	$7.6 \cdot 10^5$	$2.6 \cdot 10^5$	$2.7 \cdot 10^4$	$1.6 \cdot 10^4$	$1.2 \cdot 10^4$
$\frac{1}{32}$	$2.8 \cdot 10^7$	$4.5 \cdot 10^6$	$2.2 \cdot 10^5$	$8.6 \cdot 10^4$	$4.9 \cdot 10^4$
$\frac{1}{64}$	$9.9 \cdot 10^8$	$7.6 \cdot 10^7$	$1.9 \cdot 10^6$	$5.0 \cdot 10^5$	$2.1 \cdot 10^5$
$\frac{1}{128}$	$3.3 \cdot 10^{10}$	$1.2 \cdot 10^9$	$1.5 \cdot 10^7$	$3.2 \cdot 10^6$	$8.4 \cdot 10^5$

Comparison of Solvers

Iterations

Example: Linear finite elements on a Courant triangulation for the Poisson equation in the unit square; initial error is reduced by the factor 0.05

h	GS	CG	PCG	MG
$\frac{1}{16}$	236	12	4	1
$\frac{1}{32}$	954	23	5	2
$\frac{1}{64}$	3820	47	7	2
$\frac{1}{128}$	15287	94	11	1

Comparison of Solvers

Iterations and Convergence Rates

Example: Adaptively refined linear finite element discretization of a reaction-diffusion equation in the unit square with an interior layer; initial error is reduced by the factor 0.05

	CG		PCG		MG	
DOF	It.	κ	It.	κ	It.	κ
9	4	0.10	3	0.2	4	0.3
47	10	0.60	7	0.5	3	0.3
185	24	0.80	12	0.7	5	0.2
749	49	0.90	21	0.8	5	0.4
2615	94	0.95	37	0.9	6	0.4
5247	130	0.96	55	0.9	5	0.4

Conclusion

- Direct solvers need too much storage and computer time.
- It suffices to compute an approximate solution of the discrete problem which, compared to the solution of the differential equation, has an error similar in size to the one of the exact solution of the discrete problem.
- Iterative solvers are superior if one arrives at improving their convergence rate and at finding good initial guesses.

Classical Iterative Solvers

- Taking advantage of nested grids
- Richardson, Jacobi and Gauss-Seidel algorithms
- Comparisons

Nested Grids

- Often one has to solve a sequence of discrete problems $L_k u_k = f_k$ corresponding to increasingly more accurate discretizations.
- Usually there is a natural interpolation operator $I_{k-1,k}$ which maps functions associated with the $(k-1)$-st discrete problem into those corresponding to the k-th discrete problem.
- Then the interpolate of any reasonable approximate solution of the $(k-1)$-st discrete problem is a good initial guess for any iterative solver applied to the k-th discrete problem.
- Often it suffices to reduce the initial error by a factor 0.1.

Nested Iteration

- Compute
$$\widetilde{u}_0 = u_0 = L_0^{-1} f_0.$$
- For $k = 1, \ldots$ compute an approximate solution $\widetilde{u}_k$ for $u_k = L_k^{-1} f_k$ by applying m_k iterations of an iterative solver for the problem
$$L_k u_k = f_k$$
with starting value $I_{k-1,k}\widetilde{u}_{k-1}$.
- m_k is implicitly determined by the stopping criterion
$$\|f_k - L_k\widetilde{u}_k\| \leq \varepsilon \|f_k - L_k(I_{k-1,k}\widetilde{u}_{k-1})\|.$$

The Setting

- We have to solve a linear system $Lu = f$ with N unknowns.
- L is symmetric positive definite.
- κ denotes the condition number of L, i.e. the ratio of the largest over the smallest eigenvalue of L.
- $\kappa \approx N^{\frac{2}{d}}$

Richardson Iteration

- Iteration step: $u \mapsto u + \frac{1}{\omega}(f - Lu)$
- ω is called relaxation parameter.
- ω must be comparable in size to the largest eigenvalue of L.
- The convergence rate is $\frac{\kappa-1}{\kappa+1} \approx 1 - N^{-\frac{2}{d}}$.

Jacobi Iteration

- Iteration step: $u \mapsto u + D^{-1}(f - Lu)$
- D is the diagonal of L.
- The convergence rate is $\frac{\kappa-1}{\kappa+1} \approx 1 - N^{-\frac{2}{d}}$.
- The algorithm corresponds to sweeping through the equations and solving the i-th equation for the i-th unknown without modifying previous or subsequent equations.

Gauß-Seidel Iteration

- Iteration step: Sweep through the equations, solve the i-th equation for the i-th unknown and immediately insert the new value of the i-th unknown in all subsequent equations.
- The convergence rate is $\frac{\kappa-1}{\kappa+1} \approx 1 - N^{-\frac{2}{d}}$.

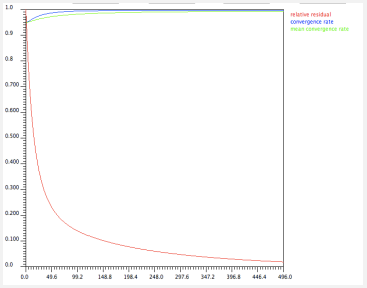
SSOR Iteration

- Iteration step:
  - Sweep through the equations first in increasing order, then in decreasing order.
  - Solve the i-th equation for the i-th unknown and write the result in the form "old value plus increment".
  - The new approximation for the i-th unknown then is the old one plus a factor (usually 1.5) times the increment.
  - Immediately insert the new value of the i-th unknown in all subsequent equations.
- The convergence rate is $\frac{\kappa-1}{\kappa+1} \approx 1 - N^{-\frac{2}{d}}$.

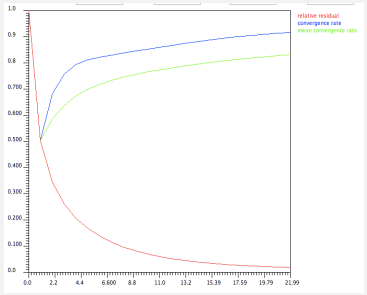
Comparison of Classical Iterative Solvers

Poisson equation on the unit square,
linear finite elements on Courant triangulation with $h = \frac{1}{64}$

Richardson
convergence rate 0.992



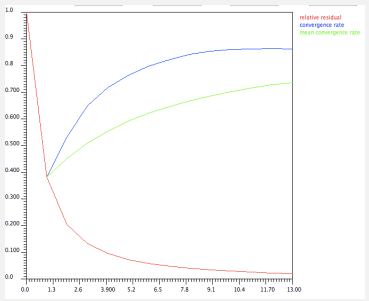
Jacobi
convergence rate 0.837



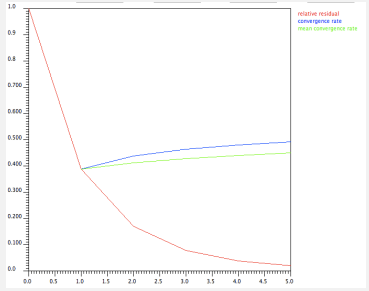
Comparison of Classical Iterative Solvers

Poisson equation on the unit square,
linear finite elements on Courant triangulation with $h = \frac{1}{64}$

Gauß-Seidel
convergence rate 0.752



SSOR
convergence rate 0.513



Conjugate Gradient Methods

- Gradient algorithm
- Conjugate gradient algorithm
- Preconditioning
- Examples

The Setting

- We have to solve a linear system $Lu = f$ with N unknowns.
- L is symmetric positive definite.
- κ denotes the condition number of L, i.e. the ratio of the largest over the smallest eigenvalue of L.
- $\kappa \approx N^{\frac{2}{d}}$

Idea of the Gradient Algorithm

- The solution of $Lu = f$ is equivalent to the minimization of the quadratic functional $J(u) = \frac{1}{2}u \cdot (Lu) - f \cdot u$.
- The negative gradient $-\nabla J(v) = f - Lv$ of J at v gives the direction of the steepest descent.
- Given an approximation v and a search direction $d \neq 0$, J attains its minimum on the line $t \mapsto v + td$ at the point $t^* = \frac{(f-Lv)\cdot d}{d\cdot(Ld)}$.

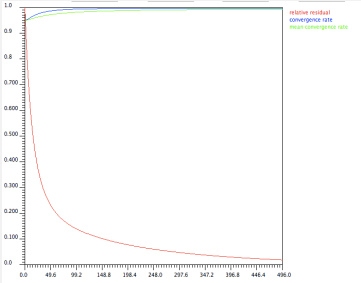
Gradient Algorithm

- Iteration step: Given the actual iterate u
  - compute the residual $r = f - Lu$,
  - replace u by $u + \frac{r\cdot r}{r\cdot Lr}r$.
- The gradient algorithm corresponds to a Richardson iteration with an automatic and optimal choice of the relaxation parameter.
- The convergence rate is $\frac{\kappa-1}{\kappa+1} \approx 1 - N^{-\frac{2}{d}}$.

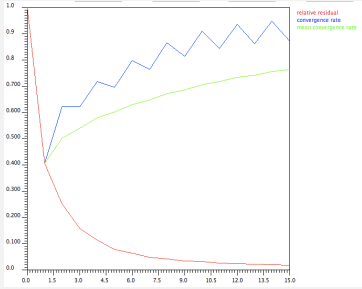
Comparison of Richardson and Gradient Algorithms

Poisson equation on the unit square,
linear finite elements on Courant triangulation with $h = \frac{1}{64}$

Richardson
convergence rate 0.992



Gradient
convergence rate 0.775



Idea of the CG-Algorithm

- The gradient algorithm slows down since the search directions become nearly parallel.
- The algorithm speeds up when choosing the successive search directions L-orthogonal, i.e. $d_i \cdot (Ld_{i-1}) = 0$.
- L-orthogonal search directions can be computed during the algorithm by a suitable three-term recursion.

The CG-Algorithm

0. Given: an initial guess u_0 for the solution, and a tolerance $\varepsilon > 0$.

1. Compute $r_0 = f - Lu_0$, $d_0 = r_0$, $\gamma_0 = r_0 \cdot r_0$. Set $i = 0$.

2. If $\gamma_i < \varepsilon^2$ return u_i as approximate solution; stop. Otherwise go to step **3**.

3. Compute $s_i = Ld_i$, $\alpha_i = \frac{\gamma_i}{d_i \cdot s_i}$, $u_{i+1} = u_i + \alpha_i d_i$, $r_{i+1} = r_i - \alpha_i s_i$, $\gamma_{i+1} = r_{i+1} \cdot r_{i+1}$, $\beta_i = \frac{\gamma_{i+1}}{\gamma_i}$, $d_{i+1} = r_{i+1} + \beta_i d_i$. Increase i by 1 and go to step **2**.

Properties

- The CG-algorithm only requires matrix-vector multiplications and inner products.
- The convergence rate is $\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \approx 1 - N^{-\frac{1}{d}}$.
- The CG-algorithm can only be applied to symmetric positive definite matrices, it breaks down for non-symmetric or indefinite matrices.

The Idea of Pre-Conditioning

- Instead of the original system $Lu = f$ solve the equivalent system $\widehat{L}\widehat{u} = \widehat{f}$ with $\widehat{L} = H^{-1}LH^{-t}$, $\widehat{f} = H^{-1}f$, $\widehat{u} = H^t u$ and an invertible square matrix H.
- Choose the matrix H such that:
  - The condition number of $\widehat{L}$ is much smaller than the one of L.
  - Systems of the form $Cv = d$ with $C = HH^t$ are much easier to solve than the original system $Lu = f$.
- Apply the conjugate gradient algorithm to the new system $\widehat{L}\widehat{u} = \widehat{f}$ and express everything in terms of the original quantities L, f, and u.

The PCG-Algorithm

0. Given: an initial guess u_0 for the solution, and a tolerance $\varepsilon > 0$.

1. Compute $r_0 = f - Lu_0$, solve $Cz_0 = r_0$ and compute $d_0 = z_0$, $\gamma_0 = r_0 \cdot z_0$. Set $i = 0$.

2. If $\gamma_i < \varepsilon^2$ return u_i as approximate solution; stop. Otherwise go to step **3**.

3. Compute $s_i = Ld_i$, $\alpha_i = \frac{\gamma_i}{d_i \cdot s_i}$, $u_{i+1} = u_i + \alpha_i d_i$, $r_{i+1} = r_i - \alpha_i s_i$, solve $Cz_{i+1} = r_{i+1}$ and compute $\gamma_{i+1} = r_{i+1} \cdot z_{i+1}$, $\beta_i = \frac{\gamma_{i+1}}{\gamma_i}$, $d_{i+1} = z_{i+1} + \beta_i d_i$. Increase i by 1 and go to step **2**.

Properties

- The convergence rate of the PCG-algorithm is $\frac{\sqrt{\widehat{\kappa}}-1}{\sqrt{\widehat{\kappa}}+1}$ where $\widehat{\kappa}$ is the condition number of $\widehat{L}$.
- Good choices of C, e.g. SSOR-preconditioning, yield $\widehat{\kappa} = N^{\frac{1}{d}}$ and corresponding convergence rates of $1 - N^{-\frac{1}{2d}}$.

SSOR-Preconditioning

0. Given: r and a relaxation parameter $\omega \in (0, 2)$. Sought: $z = C^{-1}r$.

1. Set $z = 0$.

2. For $i = 1, \ldots, N$ compute

$$z_i = z_i + \omega L_{ii}^{-1}\{r_i - \sum_{j=1}^{N} L_{ij} z_j\}.$$

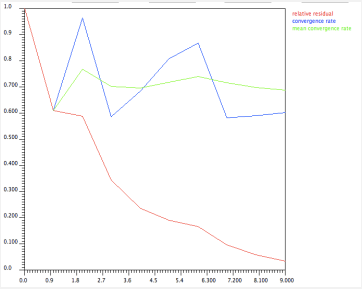
3. For $i = N, \ldots, 1$ compute

$$z_i = z_i + \omega L_{ii}^{-1}\{r_i - \sum_{j=1}^{N} L_{ij} z_j\}.$$

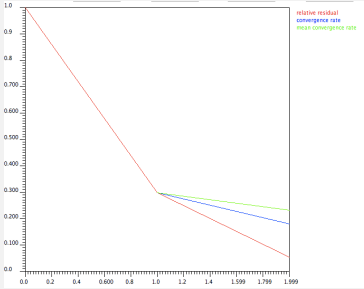
Comparison of CG and PCG Algorithms

Poisson equation on the unit square,
linear finite elements on Courant triangulation with $h = \frac{1}{64}$

CG
convergence rate 0.712



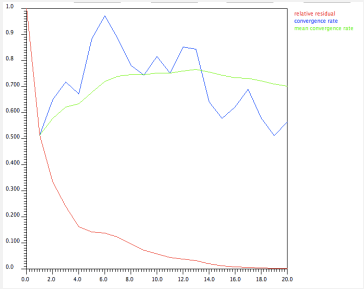
SSOR-PCG
convergence rate 0.376



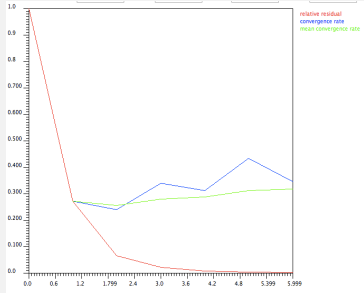
Comparison of CG and PCG Algorithms

Poisson equation on the unit square,
linear finite elements on Courant triangulation with $h = \frac{1}{128}$

CG
convergence rate 0.723



SSOR-PCG
convergence rate 0.377



The Multigrid Algorithm

- The multigrid idea
- Multigrid algorithm
- Restriction, prolongation and smoothing
- Convergence

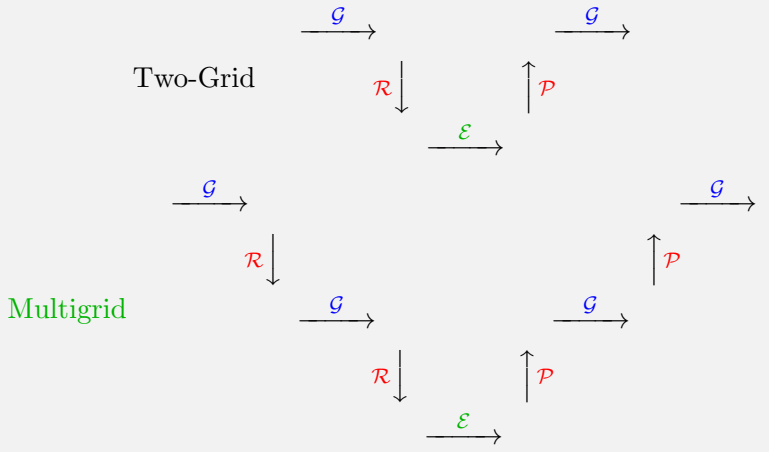
The Basic Idea of the Multigrid Algorithm

- Classical iterative methods such as the Gauß-Seidel algorithm quickly reduce highly oscillatory error components.
- Classical iterative methods such as the Gauß-Seidel algorithm are very poor in reducing slowly oscillatory error components.
- Slowly oscillating error components can well be resolved on coarser meshes with fewer unknowns.

The Basic Two-Grid Algorithm

- Perform several steps of a classical iterative method on the current grid.
- Correct the current approximation as follows:
  - Compute the current residual.
  - Restrict the residual to the next coarser grid.
  - Exactly solve the resulting problem on the coarse grid.
  - Prolongate the coarse-grid solution to the next finer grid.
- Perform several steps of a classical iterative method on the current grid.

Schematic Form



Basic Ingredients

- A sequence $\mathcal{T}_k$ of increasingly refined meshes with associated discrete problems $L_k u_k = f_k$.
- A smoothing operator M_k, which should be easy to evaluate and which at the same time should give a reasonable approximation to L_k^{-1}.
- A restriction operator $R_{k,k-1}$, which maps functions on a fine mesh $\mathcal{T}_k$ to the next coarser mesh $\mathcal{T}_{k-1}$.
- A prolongation operator $I_{k-1,k}$, which maps functions from a coarse mesh $\mathcal{T}_{k-1}$ to the next finer mesh $\mathcal{T}_k$.

The Multigrid Algorithm

0. Given: the actual level k, parameters μ, ν_1, and ν_2, the matrix L_k, the right-hand side f_k, an initial guess u_k. Sought: improved approximate solution u_k.

1. If $k = 0$ compute $u_0 = L_0^{-1} f_0$; stop.

2. (Pre-smoothing) Perform ν_1 steps of the iterative procedure $u_k \mapsto u_k + M_k(f_k - L_k u_k)$.

3. (Coarse grid correction)

3.1 Compute $f_{k-1} = R_{k,k-1}(f_k - L_k u_k)$ and set $u_{k-1} = 0$.

3.2 Perform μ iterations of the MG-algorithm with parameters $k-1$, μ, ν_1, ν_2, L_{k-1}, f_{k-1}, u_{k-1} and denote the result by u_{k-1}.

3.3 Update $u_k \mapsto u_k + I_{k-1,k} u_{k-1}$.

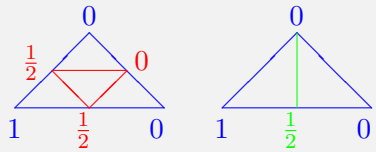
4. (Post-smoothing) Perform ν_2 steps of the iterative procedure $u_k \mapsto u_k + M_k(f_k - L_k u_k)$.

Typical Choices of Parameters

- $\mu = 1$ V-cycle or
  $\mu = 2$ W-cycle
- $\nu_1 = \nu_2 = \nu$ or
  $\nu_1 = \nu$, $\nu_2 = 0$ or
  $\nu_1 = 0$, $\nu_2 = \nu$
- $1 \leq \nu \leq 4$.

Prolongation and Restriction

- The prolongation is typically determined by the natural inclusion of the finite element spaces, i.e. a finite element function corresponding to a coarse mesh is expressed in terms of the finite element basis functions corresponding to the fine mesh.



- The restriction is typically determined by inserting finite element basis functions corresponding to the coarse mesh in the variational form of the discrete problem corresponding to the fine mesh.

Smoothing

- Gauß-Seidel iteration
- SSOR iteration:
  - Perform a forward Gauß-Seidel sweep with over-relaxation as pre-smoothing.
  - Perform a backward Gauß-Seidel sweep with over-relaxation as post-smoothing.
- ILU smoothing:
  - Perform an incomplete lower upper decomposition of L_k by suppressing all fill-in.
  - The result is an approximate decomposition $\mathcal{L}_k\mathcal{U}_k \approx L_k$.
  - Compute $v_k = M_k u_k$ by solving the system $\mathcal{L}_k\mathcal{U}_k v_k = u_k$.

Number of Operations

- Assume that
  - one smoothing step requires $O(N_k)$ operations,
  - the prolongation requires $O(N_k)$ operations,
  - the restriction requires $O(N_k)$ operations,
  - $\mu \leq 2$,
  - $N_k > \mu N_{k-1}$,
- then one iteration of the multigrid algorithm requires $O(N_k)$ operations.

Convergence Rate

- The convergence rate is uniformly less than 1 for all meshes.
- The convergence rate is bounded by $\frac{c}{c+\nu_1+\nu_2}$ with a constant which only depends on the shape parameter of the meshes.
- Numerical experiments yield convergence rates less than 0.1.

Indefinite Problems

- CG-type algorithms
- Multigrid algorithms

CG Algorithm for Non-Symmetric or Indefinite Problems

- The CG algorithm typically breaks down when applied to non-symmetric or indefinite problems (stiffness matrix has eigenvalues with positive as well as negative real part).
- A naive solution is to apply the CG algorithm to the symmetric positive definite system of normal equations $L^T Lu = L^T f$.
- This doubles the number of iterations since the passage to the normal equations squares the condition number.
- A preferable solution are specialised variants of the CG algorithm such as the stabilised bi-conjugate gradient algorithm (Bi-CG-Stab algorithm).

Bi-CG-Stab Algorithm

0. Given: an initial guess u_0 and a tolerance $\varepsilon > 0$.

1. Compute $r_0 = b - Lu_0$ and set $\overline{r}_0 = r_0$, $v_{-1} = 0$, $p_{-1} = 0$, $\alpha_{-1} = 1$, $\rho_{-1} = 1$, $\omega_{-1} = 1$, and $i = 0$.

2. If $r_i \cdot r_i < \varepsilon^2$ return u_i as approximate solution; stop. Otherwise go to step **3**.

3. Compute $\rho_i = \overline{r}_i \cdot r_i$, $\beta_{i-1} = \frac{\rho_i \alpha_{i-1}}{\rho_{i-1}\omega_{i-1}}$. If $|\beta_{i-1}| < \varepsilon$ there may be a break-down; stop. Otherwise compute $p_i = r_i + \beta_{i-1}\{p_{i-1} - \omega_{i-1}v_{i-1}\}$, $v_i = Lp_i$, $\alpha_i = \frac{\rho_i}{\overline{r}_0 \cdot v_i}$. If $|\alpha_i| < \varepsilon$ there may be a break-down; stop. Otherwise compute $s_i = r_i - \alpha_i v_i$, $t_i = Ls_i$, $\omega_i = \frac{t_i \cdot s_i}{t_i \cdot t_i}$, $u_{i+1} = u_i + \alpha_i p_i + \omega_i s_i$, $r_{i+1} = s_i - \omega_i t_i$. Augment i by 1 and go to step **2**.

Properties

- The Bi-CG-Stab algorithm aims at a simultaneous solution of the original problem $Lu = f$ as well of the adjoint problem $L^T v = f$.
- The algorithm only needs the stiffness matrix L of the original problem.
- It only requires inner products and matrix vector multiplications.
- The Bi-CG-Stab algorithm may be preconditioned; possible methods for preconditioning are the SSOR iteration or the ILU decomposition applied to the symmetric part $\frac{1}{2}(L + L^T)$ of L.

Multigrid Algorithms for Non-Symmetric or Indefinite Problems

- Multigrid algorithms can directly be applied to non-symmetric or indefinite problems.
- Eventually one as to resort to a specialised smoother.
- The Richardson iteration applied to the normal equations is a robust smoother which however yields convergence rates of about 0.8.
- The ILU decomposition is a robust smoother too, but more costly and yields convergence rates of about 0.5.

Linear and Non-Linear Optimization Problems

- Linear Optimization Problems
- Unconstrained Non-Linear Optimization Problems
- Constrained Non-Linear Optimization Problems. Optimality
- Constrained Non-Linear Optimization Problems. Algorithms
- Global Optimization Problems

Linear Optimization Problems

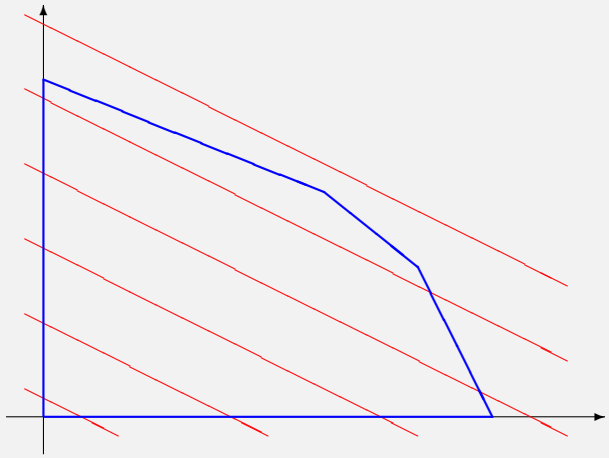
- Motivation
- Forms of linear optimization problems
- The Simplex algorithm
- Dual problems
- Complexity of the Simplex algorithm
- Interior point methods

A Motivating Example

- A small company produces two models of shoes.
- The net profit is 16 $ and 32 $, resp. per shoe.
- The required material is 6dm^2 and 15dm^2, resp. per shoe; there are 4500dm^2 available per month.
- The required machine-time is 4h and 5h, resp. per shoe; the available total time is 2000h per month.
- The required man-time is 20h and 10h, resp. per shoe; the available total time is 8000h per month.
- The company wants to maximize its profit, this lead to the optimization problem:

  maximize $16x + 32y$ subject to the constraints
  $6x + 15y \leq 4500$, $4x + 5y \leq 2000$, $20x + 10y \leq 8000$, $x \geq 0$, $y \geq 0$.

Geometric Interpretation of the Example



$16x + 32y = const$ set of constraints

General Form of Linear Optimization Problems

- Given:
  - two integers $1 \leq m < n$
  - a vector $c \in \mathbb{R}^n$
  - a matrix $A \in \mathbb{R}^{m \times n}$
  - vectors $\underline{b}, \overline{b} \in [\mathbb{R} \cup \{-\infty, \infty\}]^m$
  - vectors $\ell, u \in [\mathbb{R} \cup \{-\infty, \infty\}]^n$
- Sought:

  a minimum of the function $\mathbb{R}^n \ni x \mapsto c^t x \in \mathbb{R}$
  subject to the constraints
  - $\underline{b} \leq Ax \leq \overline{b}$
  - $\ell \leq x \leq u$
- All inequalities have to hold for all components of the corresponding vectors.

Standard Form of Linear Optimization Problems

- Given:
  - two integers $1 \leq m < n$
  - a vector $c \in \mathbb{R}^n$
  - a matrix $A \in \mathbb{R}^{m \times n}$
  - vector $b \in \mathbb{R}^m$
- Sought:

  a minimum of the function $\mathbb{R}^n \ni x \mapsto c^t x \in \mathbb{R}$
  subject to the constraints
  - $Ax = b$
  - $x \geq 0$
- The set $\mathcal{P} = \{x \in \mathbb{R}^n : Ax = b\ ,\ x \geq 0\}$ is called the set of admissible vectors associated with the optimization problem.

Simplex Form of Linear Optimization Problems

- Given:
  - two integers $1 \leq m < n$
  - a vector $c \in \mathbb{R}^n$
  - a matrix $A \in \mathbb{R}^{m \times n}$
  - vector $b \in \mathbb{R}^m$
- Sought:

  a maximum of the function $\mathbb{R} \ni z \mapsto z \in \mathbb{R}$

  subject to the constraints
  - $Ax = b$
  - $c^t x + z = 0$
  - $x \geq 0$

Equivalence of the Various Forms of Linear Optimization Problems

- The function $x \mapsto c^t x$ is minimal, if and only if the function $x \mapsto (-c)^t x$ is maximal.

  Hence, it is sufficient to consider minimization problems.
- The equality $y = b$ is equivalent to the two inequalities $y \leq b$ and $y \geq b$.
- An inequality $y \leq b$ is equivalent to equality $y + z = b$ plus the inequality $z \geq 0$.

  The vector z is called slack vector.

Properties

- The set $\mathcal{P}$ of admissible vectors is a simplex.
- If the set $\mathcal{P}$ is empty, the optimization problem is not solvable.
- If the function $x \mapsto c^t x$ is not bounded from below on $\mathcal{P}$, the optimization problem is not solvable.
- If the set $\mathcal{P}$ is not empty and bounded, the optimization problem admits a solution.
- The solution may not be unique.
- Every solution is attained at a vertex of the set $\mathcal{P}$.

Basic Idea of the Simplex Algorithm

- Given a vertex of $\mathcal{P}$ find a neighbouring vertex with a smaller value for $c^t x$.
- If such a neighbour does not exist, the current vertex solves the optimization problem.
- A vector $x \in \mathbb{R}^n$ is a vertex of $\mathcal{P}$, if it has m non-negative components and $n - m$ vanishing components and solves the system $Ax = b$.
- When freezing $n - m$ components of x to zero, the system $Ax = b$ reduces to a linear system of m equations and m unknowns involving only those columns of A which correspond to the unfrozen components of x.

Tasks

- Find a vertex of $\mathcal{P}$.
- Decide whether a given vertex is optimal.
- Find a neighbouring vertex with a smaller value of $c^t x$.

Finding a Vertex

- Given an index set $J = \{j_1, \ldots, j_m\} \subset \{1, \ldots, n\}$.
  - Set $\overline{x}_k = 0$ for all $k \notin J$.
  - Denote by A_J the $m \times m$ matrix which is obtained by discarding all columns of A corresponding to indices not contained in J.
  - Solve the linear system of equations $A_J y = b$.
  - Set $\overline{x}_{j_i} = y_i$ for $i = 1, \ldots, m$.
  - If $\overline{x}_j \geq 0$ for all $j \in J$, $\overline{x}$ is a vertex of $\mathcal{P}$.
- If $\overline{x}$ is a vertex of $\mathcal{P}$, set
  - $\overline{A} = A_J^{-1} A$,
  - $\overline{b} = A_J^{-1} b$,
  - $\beta = -c^t \overline{x}$,
  - $\widetilde{c}_i = c_{j_i}$, $1 \leq i \leq m$, and $\overline{c}^t = -\widetilde{c}^t \overline{A} + c^t$.

Checking for Optimality and Solvability

- Given a vertex $\overline{x}$ of $\mathcal{P}$.
- If $\overline{c}_k \geq 0$ for all $k \notin J$, $\overline{x}$ solves the optimization problem.
- If, for all $s \notin J$ with $\overline{c}_s < 0$, the corresponding column of $\overline{A}$ is non-positive, the optimization problem has no solution.

Finding a Neighbour with a Larger Value of $c^t x$

- Given a vertex $\overline{x}$ of $\mathcal{P}$ which is not optimal and which guarantees the solvability of the optimization problem.
- Choose an index $s \notin J$ such that $\overline{c}_s < 0$ and such that $\overline{a}$, the s-th column of $\overline{A}$, has a positive entry.
- Find an index $r \in \{1, \ldots, m\}$ such that $\overline{a}_r > 0$ and such that $\frac{\overline{b}_r}{\overline{a}_r}$ is minimal among all fractions $\frac{\overline{b}_j}{\overline{a}_j}$ with positive denominator.
- Remove the r-th entry from the index set J and put s into J.
- Update $\overline{x}$, $\overline{A}$, $\overline{b}$, β and $\overline{c}$.

Comments

- The update can be performed by dividing the r-th row of the matrix by $\overline{a}_r$ and subtracting the result from the other rows of that matrix.
- The simplex algorithm may run into a cycle since different index sets J may lead to the same value of $c^t x$.
- The cycling can be avoided by introducing a suitable ordering of the vectors x.
- The first index set J can be determined by applying the simplex algorithm to a suitable auxiliary optimization problem which has unit vectors as vertices.

Complexity of the Simplex Algorithm

- Every step of the Simplex algorithm requires $O((m+1)(n+1-m))$ operations.
- The Simplex algorithm stops after at most $\binom{n}{m}$ iterations with a solution or the information that the optimization problem has no solution.
- In the worst case the overall complexity is $O(2^{\frac{n}{2}}\left(\frac{n}{2}\right)^2)$ operations.

Dual Problem

- Every vertex yields an upper bound for the function $c^t x$.
- To obtain a lower bound for $c^t x$ one has to consider the dual optimization problem:

  Find a maximum of the function $\mathbb{R}^m \ni y \mapsto b^t y \in \mathbb{R}$ subject to the constraint $A^t y \leq c$.
- The minimal value of $c^t x$ and the maximal value of $b^t y$ are identical.
- The dual problem can be solved with a variant of the Simplex algorithm which works with the original data A, b and c.

Idea of Interior Point Methods

- The Simplex algorithm sweeps through the boundary of $\mathcal{P}$.
- Interior point methods sweep through the interior of $\mathcal{P}$.
- They try to simultaneously solve the original and the dual optimization problem.
- They reformulate both problems as a system of algebraic equations to which Newton's method is applied.
- They yield an approximation with error ε with a complexity of $O(\sqrt{n}\ln(\frac{n}{\varepsilon}))$ operations.
- This approximation is projected to a close-by vertex of $\mathcal{P}$ and a few steps of the Simplex algorithm then yield the exact solution.

Basic Form of Interior Point Methods

- Given a vector x denote by X the diagonal matrix which has the components of x as its diagonal entries.
- Consider the optimization problem $\min\{c^t x : Ax = b, x \geq 0\}$ and the corresponding dual problem $\max\{b^t y : A^t y + s = c, s \geq 0\}$.
- Then (x^*, y^*, s^*) solves both problems if and only if $\Psi_0(x^*, y^*, s^*) = 0$ where

$$\Psi_0(x, y, s) = \begin{pmatrix} Ax - b \\ A^t y + s - c \\ Xs \end{pmatrix}$$

- Apply Newton's method to this system of algebraic equations.

Improved Form of Interior Point Methods

- The derivative $D\Psi_0(x, y, s)$ becomes nearly singular when (x, y, s) approaches the solution (x^*, y^*, s^*).
- To stabilize the derivative, apply Newton's method to

$$\Psi_\mu(x, y, s) = \begin{pmatrix} Ax - b \\ A^t y + s - c \\ Xs - \mu \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \end{pmatrix}$$

and let tend μ to 0 in a judicious way.

Unconstrained Non-Linear Optimization Problems

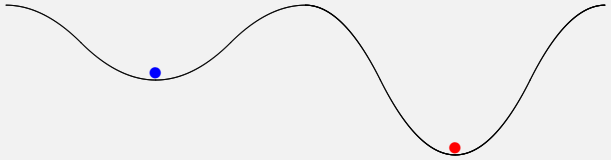
- Problem setting
- Newton's method
- Minimization methods in one dimension
- Minimization methods in several dimensions

Problem Setting

- Given:
  - a non-empty set D,
  - a function $f : D \to \mathbb{R}$
- Sought:
  - a minimizer of f, i.e. $x \in D$ with $f(x) \leq f(y)$ for all $y \in D$
- Short-hand notation:

$\min\{f(x) : x \in D\}$

Local vs. Global Minima



- Ideally, we are looking for a global minimum.
- In most cases we have to be satisfied with a local minimum.

Optimality Conditions

- If f is differentiable, then every local minimum is a critical point, i.e. satisfies $Df(x) = 0$.
- If f is twice differentiable, x is a critical point and the Hessian $D^2 f$ is positive definite, then x is a local minimum.

Newton's Method

0. Given: initial guess x_0 and tolerance ε.
Set $n = 0$.

1. If $\|Df(x_n)\| \leq \varepsilon$, go to step **(3)**.

2. Solve the linear system
$D^2 f(x_n) z_n = -Df(x_n)$,
set
$x_{n+1} = x_n + z_n$,
increase n by 1 and got to step **(1)**.

3. Check whether $D^2 f(x_n)$ is positive definite.

Difficulties

- Newton's method at best yields a critical point, its result may be a maximum or a saddle-point.
- The algorithm requires second order derivatives.
- Checking the positive definiteness of a matrix is expensive.
- A critical point may be a local minimum although $D^2 f$ is only positive semi-definite, e.g. $f(x) = x^4$.

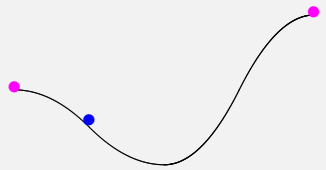
Goals

- Develop algorithms which at least find a local minimum.
- Develop algorithms which need as few derivatives as possible.
- Embed Newton's method into a larger class of algorithms to gain more flexibility and insight.
- In view of future applications, develop efficient algorithms for line search, i.e. for the minimization of functions of one variable.

One-Dimensional Minimization by Bisection

Idea

- Assume that the function $f : [a, b] \to \mathbb{R}$ is continuous and that there is a point $x \in (a, b)$ with $f(x) \leq \min\{f(a), f(b)\}$.



- Then f admits a local minimum $\eta \in (a, b)$ and $f'(\eta) = 0$ if f is differentiable.
- Determine the midpoint u of the smaller one of the two intervals $[a, x]$ and $[x, b]$ and suitably choose three points out of $\{a, x, u, b\}$.

One-Dimensional Minimization by Bisection

Algorithm

0. Given: points $a_0 < x_0 < b_0$ with $f(x_0) \leq \min\{f(a_0), f(b_0\}$, tolerance $\varepsilon < 0$. Set $k = 0$.

1. Compute $u_k = \begin{cases} \frac{1}{2}(b_k + x_k) & \text{if } x_k \leq \frac{1}{2}(a_k + b_k), \\ \frac{1}{2}(a_k + x_k) & \text{if } x_k > \frac{1}{2}(a_k + b_k). \end{cases}$

If $f(x_k) \leq f(u_k)$, set $x_{k+1} = x_k$ and

$$a_{k+1} = \begin{cases} a_k & \text{if } x_k \leq \frac{1}{2}(a_k + b_k), \\ u_k & \text{if } x_k > \frac{1}{2}(a_k + b_k), \end{cases}, \quad b_{k+1} = \begin{cases} u_k & \text{if } x_k \leq \frac{1}{2}(a_k + b_k), \\ b_k & \text{if } x_k > \frac{1}{2}(a_k + b_k). \end{cases}$$

If $f(u_k) < f(x_k)$, set $x_{k+1} = u_k$ and

$$a_{k+1} = \begin{cases} x_k & \text{if } x_k \leq \frac{1}{2}(a_k + b_k), \\ a_k & \text{if } x_k > \frac{1}{2}(a_k + b_k), \end{cases}, \quad b_{k+1} = \begin{cases} b_k & \text{if } x_k \leq \frac{1}{2}(a_k + b_k), \\ x_k & \text{if } x_k > \frac{1}{2}(a_k + b_k). \end{cases}$$

2. Increase k by 1. If $b_k - a_k < \varepsilon$ stop. Otherwise return to step **(1)**.

One-Dimensional Minimization by Bisection

Properties

- $a_k < x_k < b_k$ for all k.
- $f(x_k) \leq \min\{f(a_k), f(b_k)\}$ for all k.
- $b_k - a_k \leq (\frac{3}{4})^{k-1}(b_0 - a_0)$ for all k.
- For every prescribed tolerance, the algorithm yields an interval with length less than the tolerance which contains a local minimum of f.
- If f is differentiable, the common limit point η of the sequences a_k, b_k and x_k is a critical point of f, i.e. $f'(\eta) = 0$.
- If f is twice differentiable $f''(\eta) \geq 0$.

General Descent Algorithm

0. Given: parameters $0 < c_1 \leq c_2 < 1$, $0 < \gamma \leq 1$ and initial guess $x_0 \in \mathbb{R}^n$. Set $k = 0$.

1. If $Df(x_k) = 0$ stop, otherwise proceed with step **(2)**.

2. Choose a search direction $s_k \in \mathbb{R}^n$ with $\|s_k\| = 1$ and $-Df(x_k)s_k \geq \gamma \|Df(x_k)\|$.

3. Choose a step size $\lambda_k > 0$ such that $f(x_k + \lambda_k s_k) \leq f(x_k) + \lambda_k c_1 Df(x_k)s_k$ and $Df(x_k + \lambda_k s_k)s_k \geq c_2 Df(x_k)s_k$.

4. Set $x_{k+1} = x_k + \lambda_k s_k$, increase k by 1 and return to step **(1)**.

Choice of the Search Direction

- Smaller values of γ give more flexibility in the choice of the search direction.
- In the limiting case $\gamma \to 0$, the only condition is that the search direction must not be orthogonal to the negative gradient $-Df(x_k)$.
- The choice $s_k = -\frac{1}{\|Df(x_k)\|} Df(x_k)$ is feasible for all values of γ and corresponds to the damped Newton method.
- When applied to $f(x) = \frac{1}{2}x^t Ax - b^t x$ with a symmetric positive definite matrix A, the general descent algorithm with a suitable choice of search directions covers the gradient algorithm and (preconditioned) conjugate gradient algorithms.

Choice of the Step Size

- Exact line search: The step size λ_k is chosen such that it minimizes the function $t \mapsto f(x_k + ts_k)$ on the positive real line.
- Armijo line search: Fix a constant $\sigma > 0$, determine $\lambda^*_{k,0}$ such that $\lambda^*_{k,0} \geq \sigma \|Df(x_k)\|$ and determine the smallest integer j_k satisfying $f(x_k + 2^{-j_k}\lambda^*_{k,0}s_k) \leq f(x_k) + 2^{-j_k}c_1 Df(x_k)s_k$.
  Set $\lambda_k = 2^{-j_k}\lambda^*_{k,0}$ or $\lambda_k = 2^{-i^*}\lambda^*_{k,0}$ with $f(x_k + 2^{-i^*}\lambda^*_{k,0}s_k) = \min_i f(x_k + 2^{-i}\lambda^*_{k,0}s_k)$.

Properties of the General Descent Algorithm

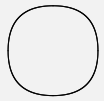
- The sequence $f(x_k)$ is monotonically decreasing.
- The sequence x_k admits at least one accumulation point.
- Every accumulation point of the sequence x_k is a critical point of f.

Constrained Non-Linear Optimization Problems. Optimality

- Convex optimization problems
- Optimality conditions for convex optimization problems
- General non-linear optimization problems
- Optimality conditions for general non-linear optimization problems

Convex Sets and Functions

- A set $C \subset \mathbb{R}^n$ is called convex, if for all $x, y \in C$ and all $\lambda \in [0,1]$ the point $\lambda x + (1-\lambda)y$ is contained in C too.



convex set

non-convex set

- A function $f : C \to \mathbb{R}^n$ on a convex set is called convex, if for all $x, y \in C$ and all $\lambda \in [0,1]$ the inequality $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$ is valid.

convex function

non-convex function

Convex Optimization Problems

- Given:
  - integers $m \geq 1$ and p with $0 \leq p \leq m$,
  - a convex set $C \subset \mathbb{R}^n$,
  - a convex function $f : C \to \mathbb{R}$,
  - convex functions $f_1, \ldots, f_p : C \to \mathbb{R}$,
  - affine functions $f_{p+1}, \ldots, f_m : C \to \mathbb{R}$.
- Sought:
  - a minimum of f under
  - the inequality constraints $f_i(x) \leq 0$ for $1 \leq i \leq p$ and
  - the equality constraints $f_j(x) = 0$ for $p+1 \leq j \leq m$
- The particular cases $p = 0$, no inequality constraints, and $p = m$, no equality constraints, are admitted.

Karush-Kuhn-Tucker Conditions for Convex Optimization Problems

- Assume that $C = \mathbb{R}^n$ and that the functions f and $f_1, \ldots, f_m$ are differentiable.
- Then $x^* \in \mathbb{R}^n$ solves the convex optimization problem, if and only if there is a $y^* \in \mathbb{R}^m$ such that

$$Df(x^*) + \sum_{i=1}^{m} y_i^* Df_i(x^*) = 0,$$

$$\begin{aligned} f_i(x^*)y_i^* &= 0, \quad 1 \leq i \leq p, \\ f_i(x^*) &\leq 0, \quad 1 \leq i \leq p, \\ y_i^* &\geq 0, \quad 1 \leq i \leq p, \\ f_j(x^*) &= 0, \quad p+1 \leq j \leq m \end{aligned}$$

Lagrange Function

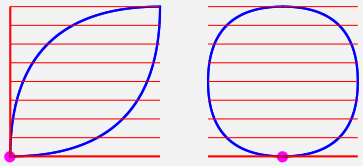
- Set $D = \{y \in \mathbb{R}^m : y_i \geq 0 \text{ for } 1 \leq i \leq p\}$.
- The function $\mathcal{L} : C \times D \to \mathbb{R}$ with
  $\mathcal{L}(x, y) = f(x) + \sum_{j=1}^{m} y_j f_j(x)$
  is called the Lagrange function of the convex optimization problem.
- $x^* \in C$ is a solution of the convex optimization problem if and only if there is $y^* \in D$ such that (x^*, y^*) is saddle point of $\mathcal{L}$, i.e.
  $\mathcal{L}(x, y^*) \geq \mathcal{L}(x^*, y^*) \geq \mathcal{L}(x^*, y)$ for all $(x, y) \in C \times D$.

General Non-Linear Optimization Problems

- Given:
  - integers $m \geq 1$ and p with $0 \leq p \leq m$,
  - a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$,
  - differentiable functions $f_1, \ldots, f_p : \mathbb{R}^n \to \mathbb{R}$,
  - differentiable functions $f_{p+1}, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$.
- Sought:
  - a minimum of f under
  - the inequality constraints $f_i(x) \leq 0$ for $1 \leq i \leq p$ and
  - the equality constraints $f_j(x) = 0$ for $p + 1 \leq j \leq m$
- The particular cases $p = 0$, no inequality constraints, and $p = m$, no equality constraints, are admitted.
- Set $S = \{x \in \mathbb{R}^n : f_i(x) \leq 0, 1 \leq i \leq p, f_j(x) = 0, p + 1 \leq j \leq m\}$.

Tangent Cones

- The tangent cone $T(S; x)$ of a set $S \subset \mathbb{R}^n$ at a point $x \in S$ is the collection of all vectors $v \in \mathbb{R}^n$ for which there is a sequence λ_k of non-negative real numbers and a sequence x_k of points in S such that $x_k \to x$ and $\lambda_k(x_k - x) \to v$.



- $T(S; x) = \mathbb{R}^n$ if x is an interior point of S.
- $T(S; x)$ is the classical tangent space if x is a boundary point of S and if the boundary of S is smooth at x.

Cone Condition

- Assume that $x^* \in S$ is a local minimum of f and that f is differentiable at x^*, then $Df(x^*)v \geq 0$ holds for all $v \in T(S; x)$.
- The cone condition is the sharpest condition for solutions of general non-linear optimization problems.
- The cone condition is of limited practical use since in general the computation of the tangent cone is too expensive, hence it is replaced by weaker more practical conditions.

Karush-Kuhn-Tucker Conditions for General Non-Linear Optimization Problems

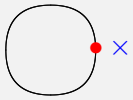
- Assume that:
  - $x^* \in S$ is a local minimum of f,
  - the gradients $Df_{p+1}(x^*), \ldots, Df_m(x^*)$ are linearly independent,
  - there is a vector $s \in \mathbb{R}^n$ with $Df_j(x^*)s = 0$ for all $p+1 \leq j \leq m$ and $Df_i(x^*)s < 0$ for all those i with $1 \leq i \leq m$ and $f_i(x^*) = 0$.
- Then there is a vector $y^* \in \mathbb{R}^m$ such that (x^*, y^*) is a saddle point of the Lagrange function $\mathcal{L}$ and
  - $Df(x^*) + \sum_{i=1}^m y_i^* Df_i(x^*) = 0$,
  - $f_i(x^*)y_i^* = 0, \quad 1 \leq i \leq p$,
  - $f_i(x^*) \leq 0, \quad 1 \leq i \leq p$,
  - $y_i^* \geq 0, \quad 1 \leq i \leq p$,
  - $f_j(x^*) = 0, \quad p+1 \leq j \leq m$.

Constrained Non-Linear Optimization Problems. Algorithms

- Projection methods
- Penalty methods
- SQP methods
- Derivative-free methods

Projection onto Convex Sets

- Assume that $S \subset \mathbb{R}^n$ is convex
- For every $x \in \mathbb{R}^n$ there is a unique point $P_S(x) \in S$, its projection, which is closest to x, i.e. $\|x - P_S(x)\| \leq \|x - y\|$ for all $y \in S$.



- The projection $P_S(x)$ is uniquely characterized by the property $(x - P_S(x))^t(y - P_S(x)) \leq 0$ for all $y \in S$.
- The projection $P_S(x)$ satisfies $(P_S(y) - P_S(x))^t(y - x) \geq \|P_S(y) - P_S(x)\|^2$ and $\|P_S(y) - P_S(x)\| \leq \|x - y\|$ for all $x, y \in \mathbb{R}^n$.

Projection Method

0. Given: a convex set $S \subset \mathbb{R}^n$, an initial guess $x_0 \in S$ and parameters $\beta, \mu \in (0,1)$ and $\gamma > 0$.
Set $k = 0$.

1. Compute $Df(x_k)$.

2. If $Df(x_k)v \geq 0$ for all $v \in T(S; x_k)$ stop, otherwise proceed with step **(3)**.

3. Find the smallest integer m_k such that $z_k = P_S(x_k - \beta^{m_k}\gamma Df(x_k))$ satisfies $f(z_k) \leq f(x_k) + \mu Df(x_k)(z_k - x_k)$.
Set $x_{k+1} = z_k$, increase k by 1 and return to step **(1)**.

Properties

- The algorithm is a damped Newton's method combined with a projection onto the set S.
- The practicability of the algorithm hinges on the computability of the tangent cones and the ability to check the cone condition $Df(x_k)v \geq 0$.
- Every accumulation point x^* of the generated sequence x_k satisfies the cone condition $Df(x^*)v \geq 0$ for all $v \in T(S; x^*)$.

Basic Idea of Penalty Methods

- 'Penalize' the constraints.
- Solve unconstrained optimization problems incorporating the 'penalization'.
- If the penalty vanishes for the solution of the auxiliary unconstrained problem we have found a solution of the original constrained problem.
- Successively increase the penalty and hope that the solutions of the auxiliary problems converge to a solution of the original constrained problem.
- Either all constraints are penalized by a penalty function or only inequality constraints are penalized by a barrier function.

Penalty Functions

- A function $\ell : \mathbb{R}^n \to \mathbb{R}$ is called a penalty function for the non-empty set $S \subset \mathbb{R}^n$ if $\ell(x) > 0$ for all $x \notin S$ and $\ell(x) = 0$ for all $x \in S$.
- The function

$$\ell(x) = \sum_{i=1}^{p} (f_i(x)^+)^\alpha + \sum_{j=p+1}^{m} |f_j(x)|^\alpha$$

with $\alpha > 0$ and $z^+ = \max\{z, 0\}$ is a penalty function for the set $S = \{x \in \mathbb{R}^n : f_i(x) \leq 0, 1 \leq i \leq p, f_j(x) = 0, p+1 \leq j \leq m\}$ associated with a general non-linear optimization problem.

Penalty Algorithm with General Penalty Function

0. Given: initial guesses $x_0 \in \mathbb{R}^n$ and $r_0 > 0$, a continuous function $f : \mathbb{R}^n \to \mathbb{R}$, a non-empty closed set $S \subset \mathbb{R}^n$ and a penalty function ℓ for S.
Set $k = 0$.

1. Compute an approximation x_k for a local minimum of $p(x, r_k) = f(x) + r_k \ell(x)$

2. If $x_k \in S$ stop.
Otherwise set $r_{k+1} = 2r_k$, increase k by 1 and return to step **(1)**.

Properties

- For sufficiently large r the function $p(x, r)$ admits a local minimum.
- The sequence x_k converges to a local minimum $x^* \in S$ of the function f.

Augmented Lagrange Function

- inequality constraints: $f_i(x) \leq 0$ for $1 \leq i \leq p$
- equality constraints: $f_j(x) = 0$ for $p+1 \leq j \leq m$
- $z^+ = \max\{z, 0\}$
- Augmented Lagrange function

$$\Lambda(x, y, r) = f(x) + \sum_{i=1}^{p} \frac{1}{2} r_i \left[\left(f_i(x) + \frac{y_i}{r_i} \right)^+ \right]^2 + \sum_{j=p+1}^{m} \frac{1}{2} r_j \left[f_j(x) + \frac{y_j}{r_j} \right]^2 - \sum_{k=1}^{m} \frac{1}{2} \frac{y_k^2}{r_k}$$

Penalty Algorithm with Augmented Lagrange Function

0. Given: a vector $r \in (\mathbb{R}_+^*)^m$ and an initial guess $y_0 \in (\mathbb{R}_+)^p \times \mathbb{R}^{m-p}$.
Set $k = 0$.

1. Determine a local minimum x_k of the augmented Lagrange function $x \mapsto \Lambda(x, y_k, r)$.

2. If (x_k, y_k) satisfies the Karush-Kuhn-Tucker conditions stop. Otherwise proceed with step **(3)**.

3. Set
$y_{k+1,i} = (r_i f_i(x_k) + y_{k,i})^+$ for $1 \leq i \leq p$,
$y_{k+1,j} = r_j f_j(x_k) + y_{k,j}$ for $p+1 \leq j \leq m$.
Increase k by 1 and return to step **(1)**.

Properties

- If $r = (\rho, \ldots, \rho)^t$ with a sufficiently large ρ, the algorithm converges to a saddle point of the Lagrange function $\mathcal{L}$.
- The convergence is linear.
- Convergence speed improves with increasing ρ.

Barrier Functions

- A function $\mathcal{B} : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ is called barrier function if it has the following properties:
  - $\mathcal{B}(t) = \infty$ for all $t \leq 0$.
  - $\mathcal{B}$ is monotonically decreasing.
  - $\mathcal{B}$ is convex.
  - $\mathcal{B}$ is continuously differentiable on $\mathbb{R}_+^*$.
  - $\lim\limits_{t \to 0+} \mathcal{B}(t) = \infty$.
  - $\lim\limits_{t \to 0+} \mathcal{B}'(t) = -\infty$.
- $\mathcal{B}(t) = \begin{cases} -\ln t & \text{for } t > 0 \\ \infty & \text{for } t \leq 0 \end{cases}$ and $\mathcal{B}(t) = \begin{cases} t^{-\alpha} & \text{for } t > 0 \\ \infty & \text{for } t \leq 0 \end{cases}$ with $\alpha > 0$ are barrier functions.

Barrier Algorithm for Convex Optimization

0. Given: convex functions f and $f_1, \ldots, f_p$ and affine functions $f_{p+1}, \ldots, f_m$, a barrier function $\mathcal{B}$ and an initial guess $x_0 \in \mathbb{R}^n$ with $f_j(x_0) = 0$ for $p+1 \leq j \leq m$.
Choose $\mu_0 > 0$ and $d_0 \in (\mathbb{R}_+^*)^p$ with $f_i(x_0) < d_{i,0}$ for $1 \leq i \leq p$.
Set $k = 0$.

1. Choose $\lambda_k \in (0,1)$ with $f_i(x_k) < \lambda_k d_{i,k}$ for $1 \leq i \leq p$. Set $\mu_{k+1} = \lambda_k \mu_k$, $d_{k+1} = \lambda_k d_k$.

2. Starting with x_k apply a line search to the problem

$$\min_x \{f(x) + \mu \sum_{i=1}^{p} \mathcal{B}(d_i - f_i(x)) : f_j(x) = 0\ ,\ p+1 \leq j \leq m\}$$

with result x_{k+1}. Increase k by 1 and return to **(1)**.

Basic Idea of the Sequential Quadratic Programming Algorithm

- Replace the Lagrange Function $\mathcal{L}$ by a second order approximation.
- Linearize the constraints.
- Successively solve constrained optimization problems with a quadratic object function and affine constraints.

SQP Algorithm

0. Given: initial guesses $x_0 \in \mathbb{R}^n$, $y_0 \in (\mathbb{R}_+^*)^p \times \mathbb{R}^{m-p}$.
Compute $B_0 = D^2 f(x_0) + \sum\limits_{i=1}^{m} y_{0,i} D^2 f_i(x_0)$ and set $k = 0$.

1. Find a solution (s, y) for the Karush-Kuhn-Tucker conditions of the auxiliary problem

$$\min_s \left\{ Df(x_k)s + \frac{1}{2} s^t B_k s : f_i(x_k) + Df_i(x_k)s \leq 0, 1 \leq i \leq p, f_j(x_k) + Df_j(x_k)s = 0, p+1 \leq j \leq m \right\}.$$

2. Set $x_{k+1} = x_k + s$, $y_{k+1} = y_k + y$.

3. Compute $B_{k+1} = D^2 f(x_{k+1}) + \sum\limits_{i=1}^{m} y_{k+1,i} D^2 f_i(x_{k+1})$, increase k by 1 and return to step **(1)**.

Properties

- The SQP algorithm is locally quadratically convergent.
- If the B_k are replaced by approximations in a suitable quasi Newton type, the convergence still is linear.

Basic Idea of the Simplex Method of Nelder and Mead

- Minimize a function f over $\mathbb{R}^n$.
- Take into account eventual constraints by setting $f(x) = \infty$ if x violates the constraints.
- Choose $n+1$ points $x_0, \ldots, x_n$ generating $\mathbb{R}^n$.
- Sort these points by increasing size of f.
- Reflect x_n at the barycentre of $x_0, \ldots, x_{n-1}$ and eventually expand or contract the image x' depending on the values $f(x_0), \ldots, f(x_n)$ and $f(x')$.
- Replace an appropriate member of the list $x_0, \ldots, x_n$ by x'.

Simplex Method of Nelder and Mead.

0. Given: points $x_0, \ldots, x_n \in \mathbb{R}^n$ generating $\mathbb{R}^n$ sorted by increasing size of f, tolerance $\varepsilon > 0$.

1. If the standard deviation of the f-values is less than ε stop.

2. Compute $c = \frac{1}{n}\sum_{i=0}^{n-1} x_i$, $x_r = 2c - x_n$ and $f(x_r)$.

3. Decide:

3.1 If $f(x_0) \le f(x_r) \le f(x_{n-1})$ replace x_n by x_r (reflection).

3.2 If $f(x_r) < f(x_0)$ compute $x_e = 2x_r - c$ and $f(x_e)$. If $f(x_e) < f(x_r)$ replace x_r by x_e.
Replace x_n by x_r (expansion).

3.3 If $f(x_r) > f(x_{n-1})$ compute $x_c = \begin{cases} c+\frac{1}{2}(x_n-c) & \text{if } f(x_r)\ge f(x_n) \\ c+\frac{1}{2}(x_r-c) & \text{if } f(x_r)<f(x_n) \end{cases}$ and $f(x_c)$.
If $f(x_c) < \min\{f(x_n), f(x_r)\}$ replace x_n by x_c, otherwise compute $x_i = \frac{1}{2}(x_0 + x_i)$ for $1 \le i \le n$ (contraction).

4. Re-sort $x_0, \ldots, x_n$ by increasing size of f and return to **(1)**.

Properties

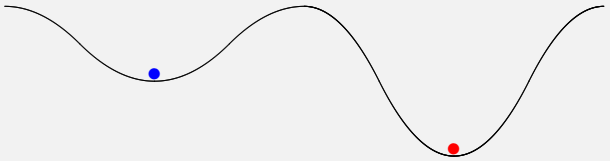
- The algorithm is very cheap since it does not require the computation of any derivative.
- The algorithm is very slow.
- There is no convergence proof.
- The algorithm is very robust.
- The algorithm may yield suitable initial guesses for the algorithms presented previously.

Global Optimization Problems

- Problem setting
- Structure of global optimization algorithms
- Ingredients
- Concluding remarks

Problem Setting

- All algorithms considered so far at best yield a local minimum.
- We want to find a global minimum of even all of them.



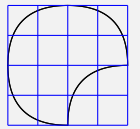
- This difficulty only arises for non-convex optimization problems since a convex function has at most one local minimum which is the global minimum.

Structure of Global Optimization Algorithms

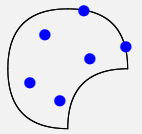
- Try several candidates for a global minimum.
- Eventually replace candidates by the result of a local search, i.e. apply one of the previously described algorithms with a given candidate as initial guess.
- Eventually iterate on lists of candidates.
- Eventually perturb candidates.
- Algorithms differ by
  - the initial choice of candidates,
  - the method for updating the list of candidates,
  - the form of perturbation,
  - the amount of randomness,
  - the work invested in local searches.

Initial Choice of Candidates

- Deterministic: Cover the domain $S \subset \mathbb{R}^n$ of admissible points x by a uniform mesh.



- Random: Cover the domain $S \subset \mathbb{R}^n$ of admissible points x by a random mesh according to a chosen probability measure, e.g. uniform distribution.



- In both approaches eventually construct several lists of candidates by iteratively reducing the mesh size.

Updating Lists of Candidates

- Replace candidates by the result of a local search.
- Replace candidates by a perturbation.
- With a small probability also accept candidates with a larger value of f, e.g. simulated annealing:
  x' with $f(x') > f(x)$ is allowed to replace x with probability $e^{\frac{f(x)-f(x')}{T}}$.
- Update lists by branch and bound techniques.

Perturbation of Candidates

- Normalize all points such that all their co-ordinates are represented by an N-bit string.
- Given a candidate pick one of its components by random and flip one of its bits by random.
- Example: $N = 4$, $x = 15 = 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$,
  $x' = 11 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$ is a perturbation of x
  $x'' = 7 = 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$ is a perturbation of x
  $\widetilde{x} = 9 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$ is no perturbation of x

Concluding Remarks

- Each algorithm has its own benefits and drawbacks.
- The choice of an efficient algorithm requires knowledge of the particular structure of the given optimization problem.
- There is no efficient black-box algorithm.

References

A. Quarteroni, R. Sacco and F. Saleri
Numerical Mathematics
Springer, 2000

D. Braess
Finite Elements. Theory, Fast Solvers, and Applications in Solid Mechanics
Cambridge University Press, 2001

E. M. T. Hendrix and B. G. Tóth
Introduction to Nonlinear and Global Optimization
Springer, 2010