Numerical Simulation Methods in Geophysics, Part 4: Other equations

1. MGPY+MGIN

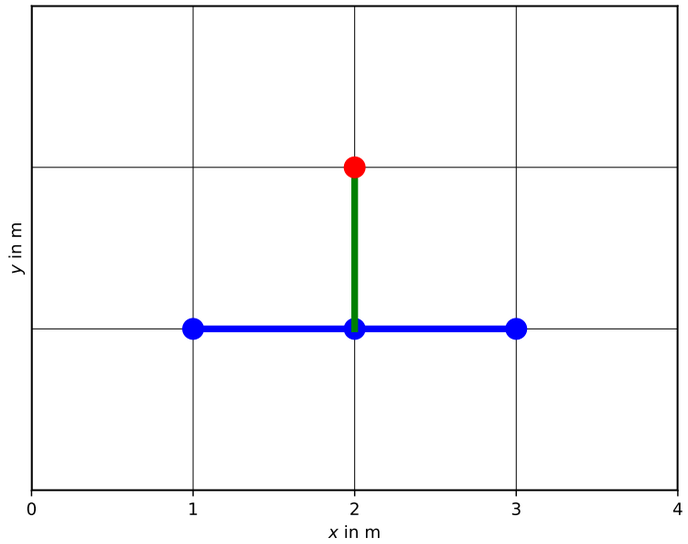
thomas.guenther@geophysik.tu-freiberg.de

TU BERGAKADEMIE FREIBERG

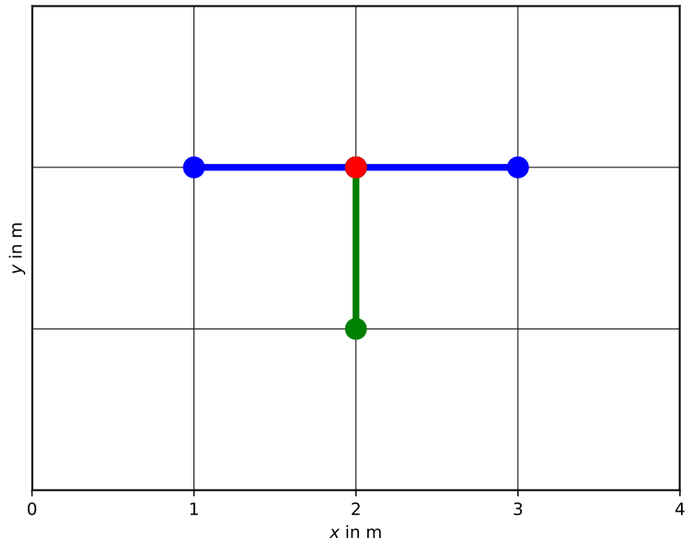
TUBAF

Die Ressourcenuniversität.
Seit 1765.

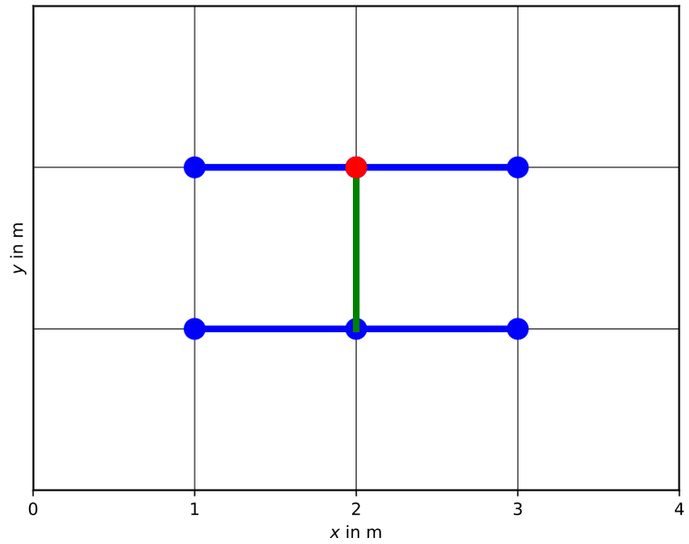
Finite Differences



Explicit



Implicit

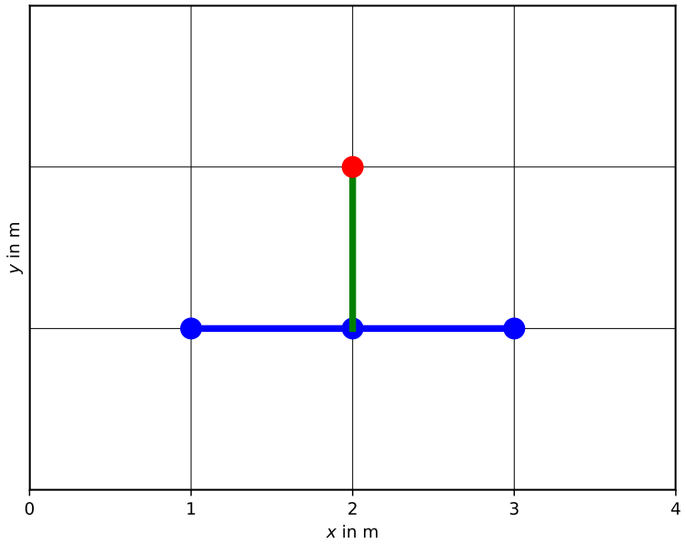


Mixed

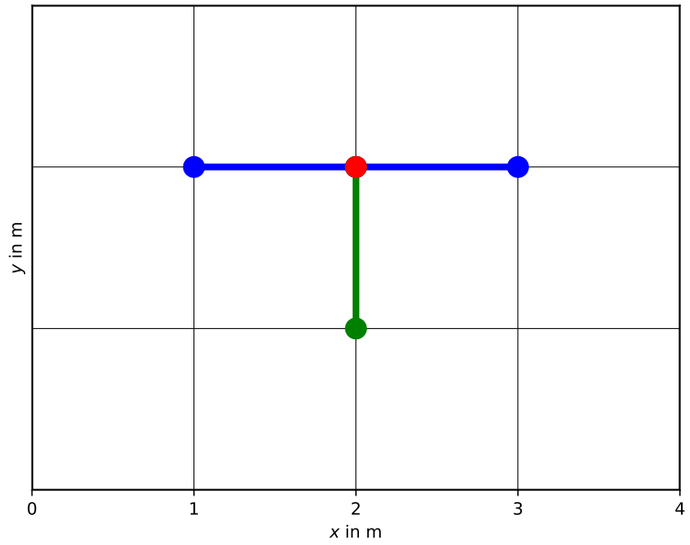
Recap Finite Differences

- elliptic (Poisson) or parabolic PDE problems
- replace partial differential operators ∂ by finite differences Δ
- transfer PDE into a matrix-vector equation $\mathbf{A}\mathbf{u} = \mathbf{b}$
- finite-difference stencil spatial or temporal
- spatial derivative $\Rightarrow$ stiffness $\mathbf{A}$, temporal $\Rightarrow$ mass matrix $\mathbf{M}$
- time-stepping explicit, implicit or mixed (stable & accurate)
- accuracy depends on discretization, parameter contrast

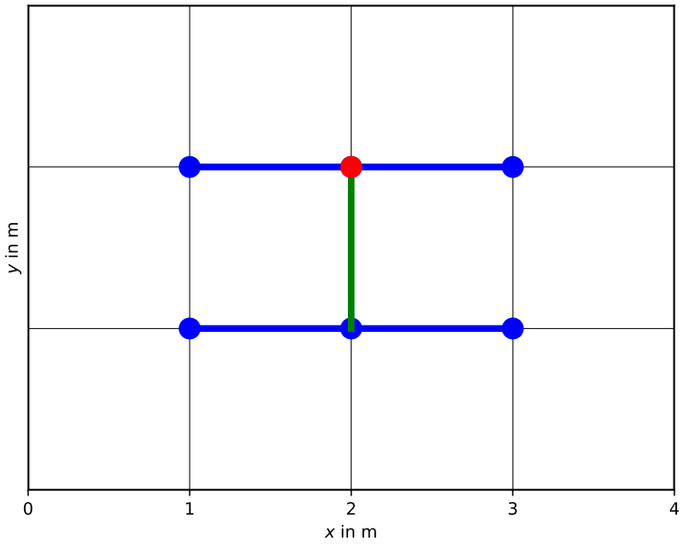
Time-stepping schemes



Explicit



Implicit



Mixed

Stability of time-stepping

Consider Newton cooling problem

$$\frac{\partial T}{\partial t} = -\frac{T}{\tau} \approx \frac{\mathrm{d}T}{\mathrm{d}t}$$

with solution

$$T(t) = T_0 \exp(-t/\tau)$$

Explicit Euler method

$$T_{i+1} = T_i - \frac{dt}{\tau} T_i = T_i(1 - \frac{dt}{\tau}) = T_0(1 - \frac{dt}{\tau})^{i+1}$$

$T_{i+1} < T_i$ requires $0 \leq 1 - dt/\tau < 1$

$$\Rightarrow 0 < dt \leq \tau$$

Important

Explicit method simple, but instable for $dt < \tau$

Implicit Euler method

$$\frac{T_{i+1} - T_i}{dt} = -\frac{T_{i+1}}{\tau}$$

$$T_{i+1} = T_i \frac{1}{1 + dt/\tau}$$

$$0 \leq \frac{1}{1 + dt/\tau} < 1$$

Caution

unconditionally stable (but maybe still inaccurate)

Mixed (Crank-Nicholson) method

$$\frac{T_{i+1} - T_i}{dt} = -\frac{T_{i+1} + T_i}{2\tau}$$

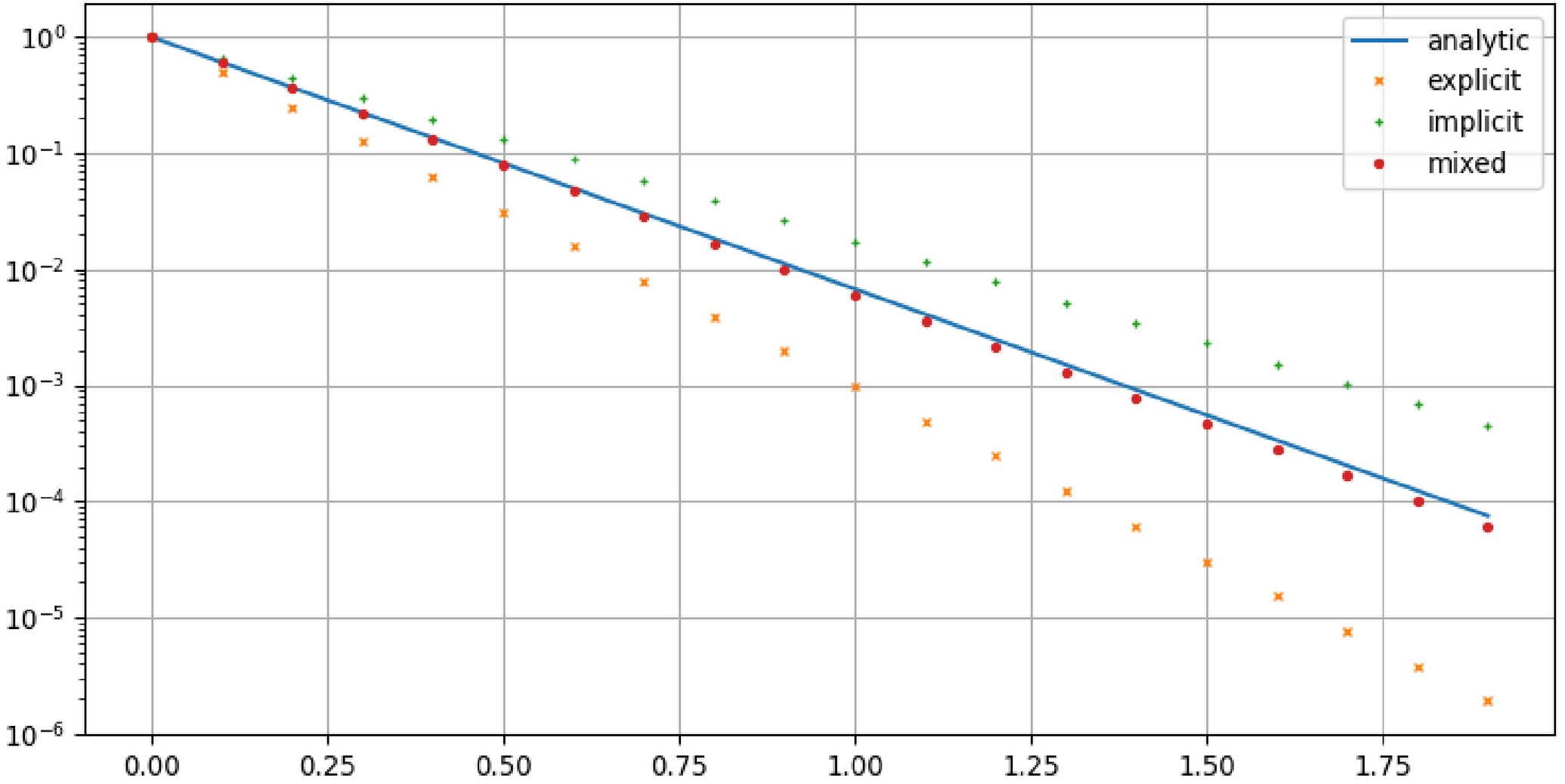
$$T_{i+1}(1 + dt/2\tau) = T_i(1 - dt/2\tau)$$

$$T_{i+1} = T_i \frac{1 - dt/2\tau}{1 + dt/2\tau}$$

Tip

always stable and decreasing for $dt < 2\tau$

Numerical comparison



Other equation types

- [x] Poisson equation (elliptic)
- [x] diffusion equation (parabolic)
- [ ] wave equation (hyperbolic)
- [ ] Helmholtz equation (elliptic)

Hyberbolic equations

Acoustic wave equation in 1D

$$\frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = 0$$

u..pressure/velocity/displacement, c..velocity

Damped (mixed parabolic-hyperbolic) wave equation

$$\frac{\partial^2 u}{\partial t^2} - a\frac{\partial u}{\partial t} - c^2 \frac{\partial^2 u}{\partial x^2} = 0$$

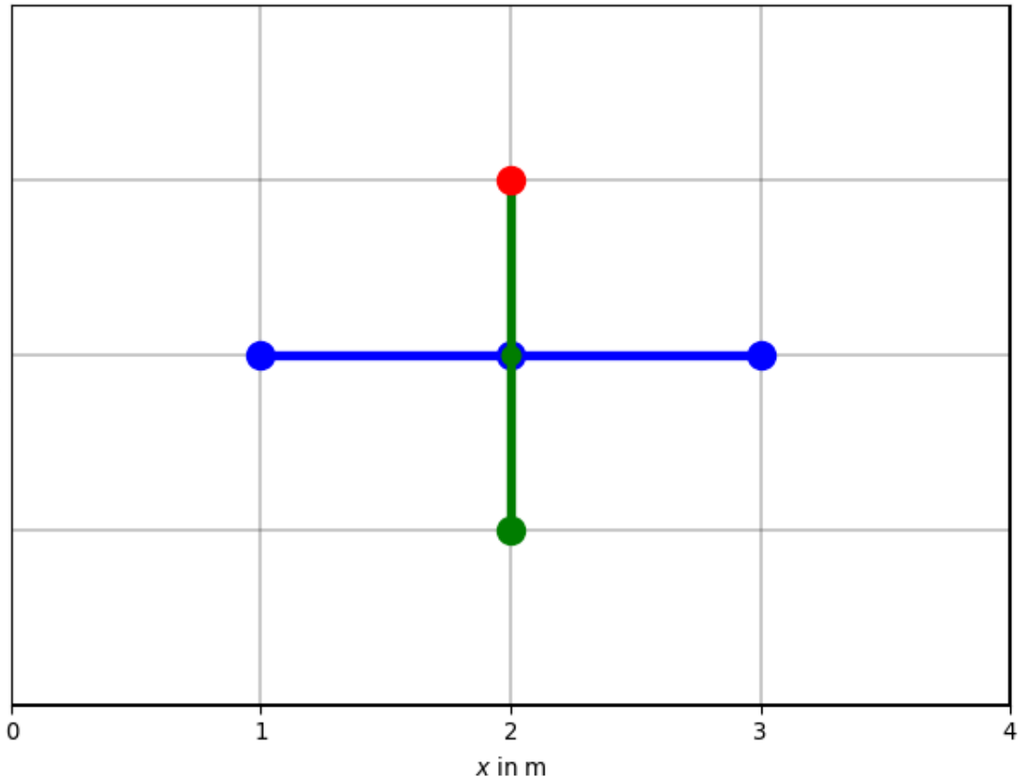
Discretization

$$\frac{\partial^2 u}{\partial t^2}^n \approx \frac{u^{n+1} - u^n}{\Delta t} - \frac{u^n - u^{n-1}}{\Delta t}$$

$$= \frac{u^{n+1} + u^{n-1} - 2u^n}{\Delta t^2} = c^2 \frac{\partial^2 u}{\partial x^2}^n$$

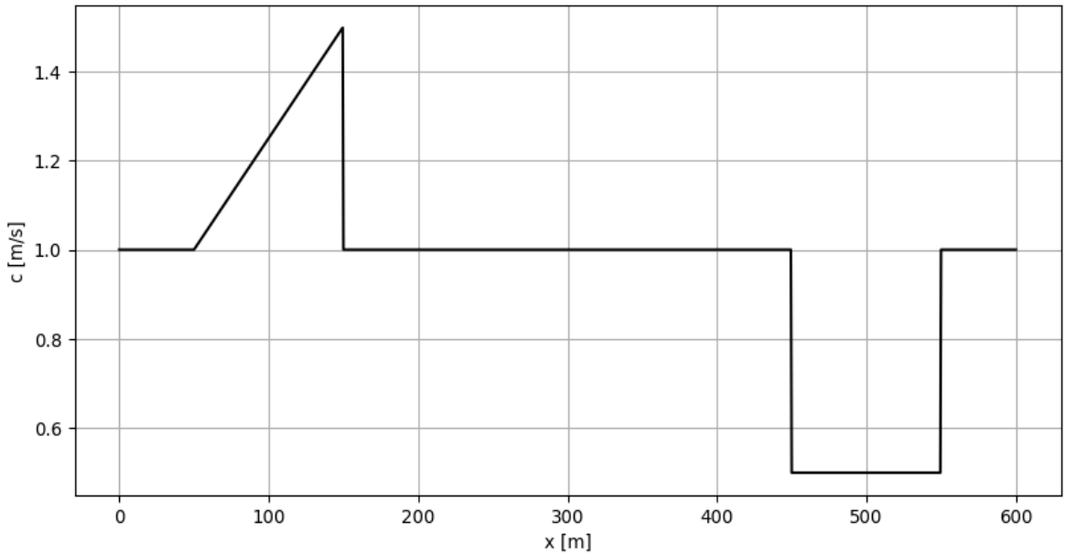
$$u^{n+1} = c^2 \Delta t^2 \frac{\partial^2 u}{\partial x^2}^n + 2u^n - u^{n-1}$$

$$\mathbf{u}^{n+1} = (2\mathbf{I} - \mathbf{A})\mathbf{u}^n - \mathbf{I}\mathbf{u}^{n-1}$$



Example: velocity distribution

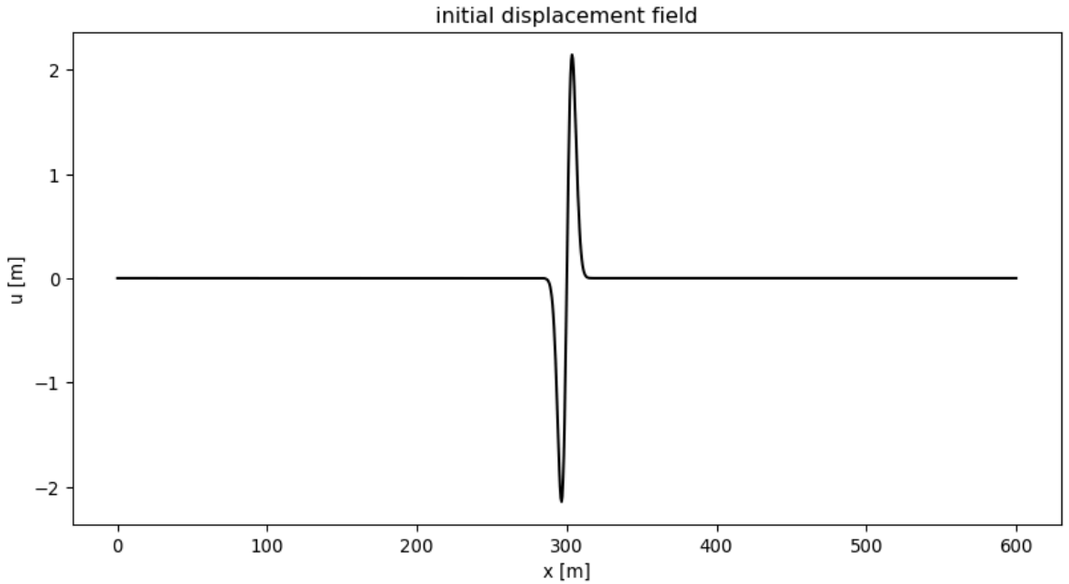
```
1  x=np.arange(0, 600.01, 0.5)
2  c = 1.0*np.ones_like(x) # velocity in m/s
3  c[100:300] = 1 + np.arange(0,0.5,0.0025)
4  c[900:1100] = 0.5  # low velocity zone
5
6  # Plot velocity distribution.
7  plt.plot(x,c,'k')
8  plt.xlabel('x [m]')
9  plt.ylabel('c [m/s]')
10 plt.grid()
```



Initial displacement

Derivative of Gaussian (Ricker wavelet)

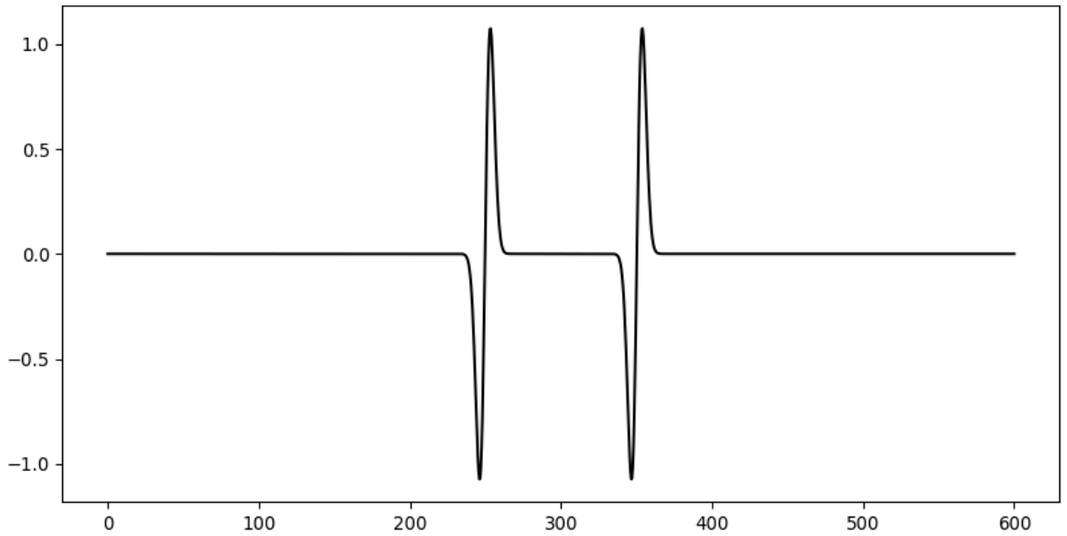
```
1  l=5.0
2
3  # Initial displacement field [m].
4  u=(x-300.0)*np.exp(-(x-300.0)**2/l**2)
5  # Plot initial displacement field.
6  plt.plot(x,u,'k')
7  plt.xlabel('x [m]')
8  plt.ylabel('u [m]')
9  plt.title('initial displacement field')
10 plt.show()
```

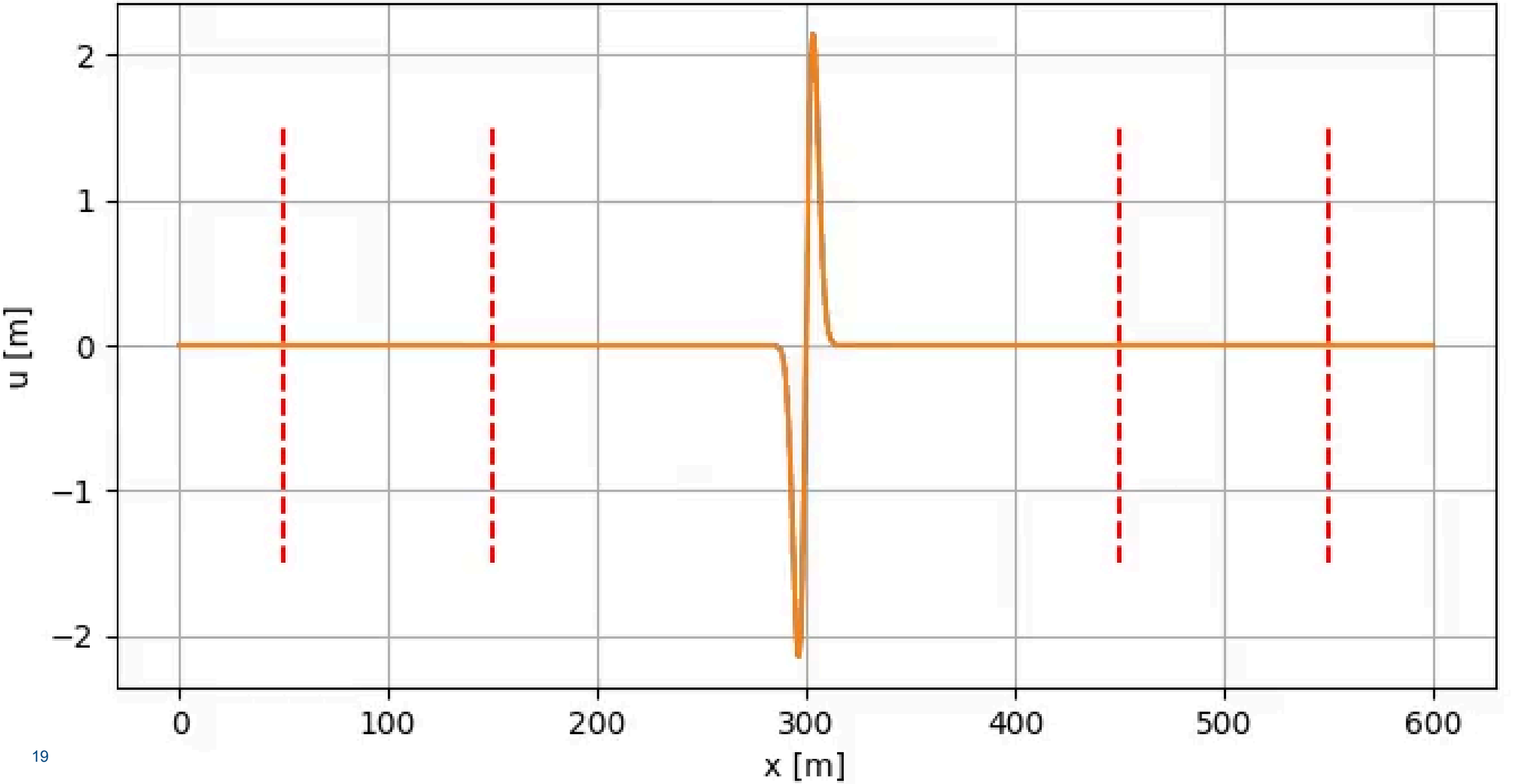


Time propagation

```
u_last=u
dt = 0.5
ddu = np.zeros_like(u)
dx = np.diff(x)
for i in range(100):
    dudx = np.diff(u)/dx
    ddu[1:-1] = np.diff(dudx)/dx[:-1]
    u_next = 2*u-u_last+ddu*c**2 * dt**2
    u_last = u
    u = u_next

plt.plot(x,u,'k')
```




2
1
0
−1
−2
u [m]
0
100
200
300
400
500
600
x [m]

Helmholtz equations

e.g. from Fourier assumption $u = u_0 e^{\imath \omega t}$

$$-\boldsymbol{\nabla} \cdot (a \boldsymbol{\nabla} u) + k^2 u = f$$

- Laplace operator assembled in FD system matrix $\mathbf{A}$
- additional terms with $u_i \Rightarrow$ add “mass” to stiffness

$$\Rightarrow (\mathbf{A} + k^2 \mathbf{I}) \mathbf{u} = \mathbf{b}$$

Wave equations in frequency domain

$$\frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = 0$$

separation approach: $u(z,t) = e^{\imath\omega t} X(x)$

$$\Rightarrow \frac{\partial^2 u}{\partial t^2} = -\omega^2 e^{\imath\omega t} X(x) = -\omega^2 u$$

$$\Rightarrow -\omega^2 u - c^2 \frac{\partial^2 u}{\partial x^2} = 0 \Rightarrow a \frac{\partial^2 X}{\partial x^2} + \omega^2 X = 0$$

Heat transfer in frequency domain

separation approach: $T(z,t) = e^{\imath\omega t} Z(z)$

$$\Rightarrow \frac{\partial T}{\partial t} = \imath\omega e^{\imath\omega t} Z(z) = \imath\omega T$$

$$\frac{\partial T}{\partial t} - a\frac{\partial^2 T}{\partial z^2} \Rightarrow -a\frac{\partial^2 T}{\partial z^2} + \imath\omega T = 0$$

$$\Rightarrow -ae^{\imath\omega t}\frac{\partial^2 Z}{\partial z^2} + \imath\omega e^{\imath\omega t} Z = 0$$

Heat transfer in frequency domain

separation approach: $T(z,t) = e^{\imath\omega t} Z(z)$

$$\Rightarrow \frac{\partial T}{\partial t} = \imath\omega e^{\imath\omega t} Z(z) = \imath\omega T$$

$$\frac{\partial T}{\partial t} - a\frac{\partial^2 T}{\partial z^2} \Rightarrow -a\frac{\partial^2 T}{\partial z^2} + \imath\omega T = 0$$

$$\Rightarrow -a\frac{\partial^2 Z}{\partial z^2} + \imath\omega Z = 0$$

Maxwells equations

- Faraday's law: currents & varying electric fields $\Rightarrow$ magnetic field

$$\boldsymbol{\nabla} \times \mathbf{H} = \frac{\partial \mathbf{D}}{\partial t} + \mathbf{j}$$

- Ampere's law: time-varying magnetic fields induce electric field

$$\boldsymbol{\nabla} \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

- $\boldsymbol{\nabla} \cdot \mathbf{D} = \varrho$ (charge $\Rightarrow$), $\boldsymbol{\nabla} \cdot \mathbf{B} = 0$ (no magnetic charge)
- material laws $\mathbf{D} = \epsilon \mathbf{E}$ and $\mathbf{B} = \mu \mathbf{H}$ (now ϵ/μ constant)

Maxwell in frequency domain

$$\mathbf{E} = \mathbf{E}_0 e^{\imath\omega t} \quad \text{or} \quad \mathbf{H} = \mathbf{H}_0 e^{\imath\omega t}$$

$$\boldsymbol{\nabla} \times \mathbf{H} = \imath\omega\epsilon\mathbf{E} + \sigma\mathbf{E}$$

$$\boldsymbol{\nabla} \times \mathbf{E} = -\imath\omega\mu\mathbf{H}$$

Helmholtz equation

see also Theory EM

take curl of one of the equations and insert in the other

$$\boldsymbol{\nabla} \times \boldsymbol{\nabla} \times \mathbf{E} + \imath\omega\mu\sigma\mathbf{E} - \omega^2\mu\epsilon\mathbf{E} = \boldsymbol{\nabla} \times \mathbf{j}_s$$

$$\boldsymbol{\nabla} \times \rho\boldsymbol{\nabla} \times \mathbf{H} + \imath\omega\mu\mathbf{H} - \omega^2\mu\epsilon\rho\mathbf{H} = 0$$

Quasi-static approximation

Assume: $\omega^2 \mu\epsilon < \omega\mu\sigma$, no sources ($\boldsymbol{\nabla} \cdot \mathbf{j}_s = 0$), + vector identity

$$\boldsymbol{\nabla} \times \boldsymbol{\nabla} \times \mathbf{F} = \boldsymbol{\nabla}\boldsymbol{\nabla} \cdot \mathbf{F} - \boldsymbol{\nabla}^2 \mathbf{F}$$

leads with $\boldsymbol{\nabla} \cdot \mathbf{E} = 0 = \boldsymbol{\nabla} \cdot \mathbf{B}$ to the vector Helmholtz PDE

$$-\boldsymbol{\nabla}^2 \mathbf{E} + \imath\omega\mu\sigma\mathbf{E} = 0$$

$$-\boldsymbol{\nabla} \cdot \sigma^{-1} \boldsymbol{\nabla} H_{xyz} + \imath\omega\mu H_{xyz} = 0$$

Electromagnetic fields in the Earth

Maxwell equations lead to diffusion equation

$$\frac{\partial^2 \mathbf{B}}{\partial z^2} = \mu_0 \sigma \frac{\partial \mathbf{B}}{\partial t}$$

A periodic excitation ($B_0 e^{\imath \omega t}$) leads to (cf. temperature diffusion)

$$B = B_0 e^{-z/d} \cos(\omega t - z/d)$$

with the skin depth $d = \sqrt{2/(\mu_0 \sigma \omega)} \approx 503\sqrt{\rho/f}$

Equation solvers

$$\mathbf{A}\mathbf{u} = \mathbf{b}$$

A represents PDE (incl. coefficients) and BC equations, $\mathbf{b}$ BC values

Sparse matrices

Up to now: regular (dense) array: save every element including 0

Sparse: Save only non-zero components (e.g. using `scipy.sparse`)

- COO - coordinate format
- CSC/CRS - compressed sparse column/row
- BSR - block sparse row format, …

Solve systems of equations

direct solvers

- Gauss elimination (expensive and dense)
- Cholesky (or ILU) decomposition

indirect solvers (fixed-point iteration)

- gradient methods: steepest descent, conjugate gradients
- preconditioning: incomplete factorizations (of submatrices)

Cholesky decomposition

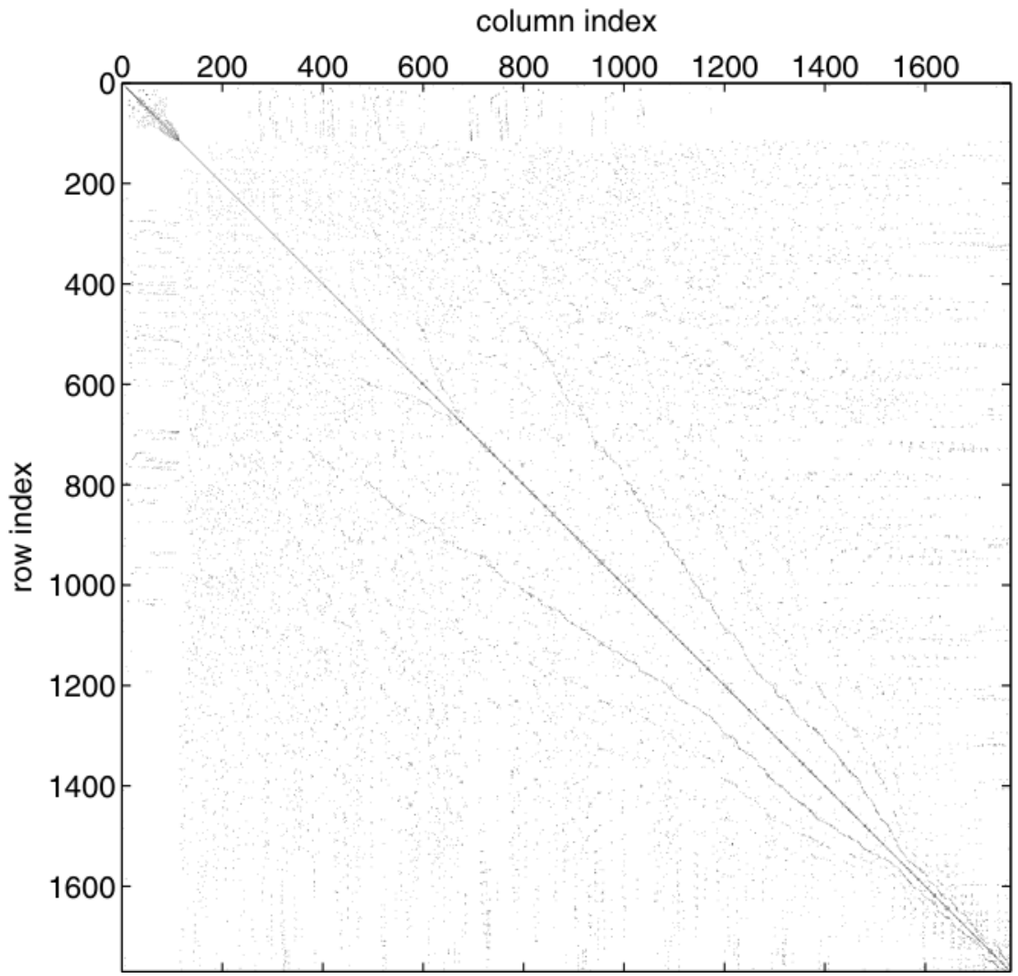
$$\mathbf{A} = \mathbf{L}\mathbf{L}^T \quad \text{or} \quad \mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T$$

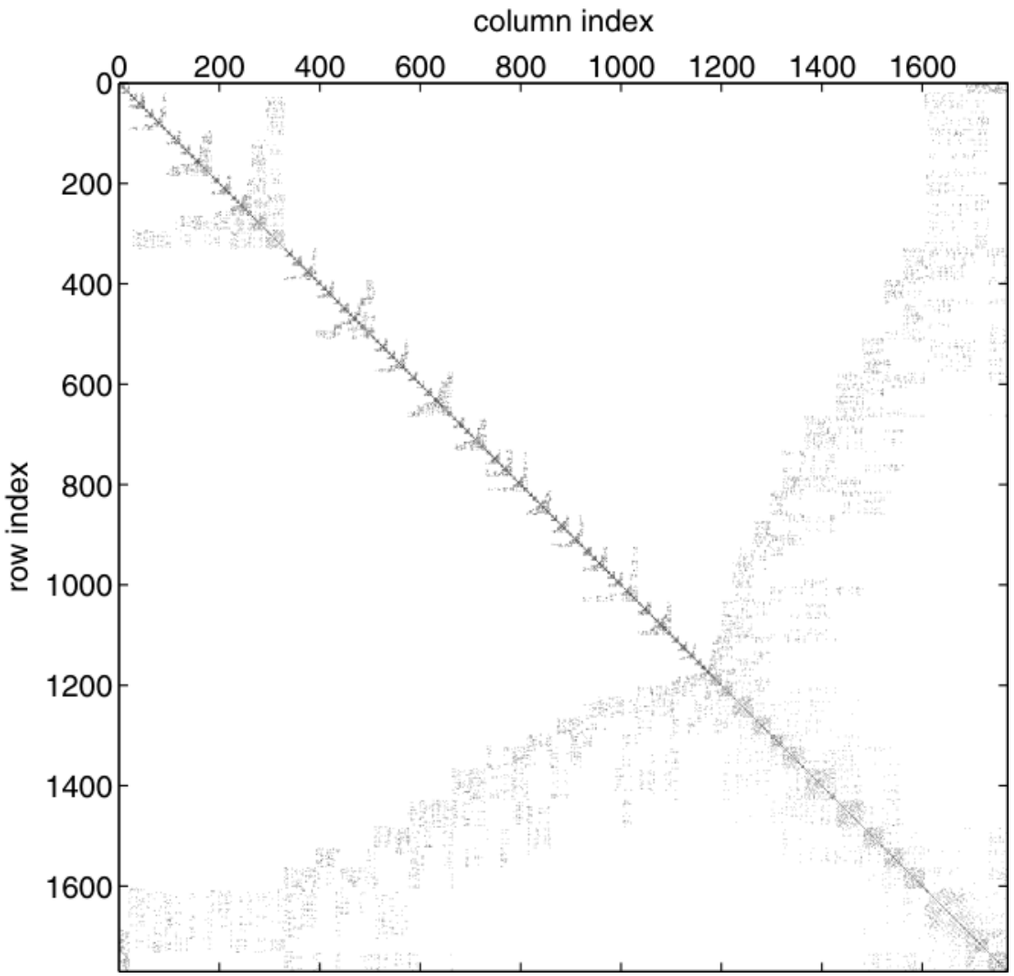
Particularly important if many right-hand sides need to be computed

- many transmitters (flying helicopter)
- time-stepping (same matrix in every step)
- computation of Jacobian matrix (inversion)

- do decomposition (generate sparse $\mathbf{L}$ matrix)
- back-substitution for every right-hand side

Reordering





original (left) & reordered (right) matrix (Rücker et al. 2006)

Iterative solvers - fixed-point iteration

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

split $\mathbf{A} = \mathbf{M} - \mathbf{N}$ ($\mathbf{M}$ easily invertible) and solve

$$\mathbf{M}\mathbf{x} = \mathbf{N}\mathbf{x} + \mathbf{b}$$

iterativily by $\mathbf{x}^{k+1} = \mathbf{M}^{-1}\left(\mathbf{N}\mathbf{x}^{k} + \mathbf{b}\right)$

e.g. $\mathbf{M} = \mathrm{diag}(\mathbf{A})$ (Jacobi method) or
$\mathbf{M} = \mathrm{tril}(\mathbf{A})$ (Gauss-Seidel method)

Gradient methods

minimize residual $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$

Steepest descent method

$$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$$

go towards $\mathbf{r}_0$ and minimize step length α

$$\mathbf{r}_1^T \mathbf{r}_0 = 0 = \mathbf{b} - \mathbf{A}(\mathbf{x}_0 + \alpha \mathbf{r}_0)$$

$$\Rightarrow \alpha = \frac{\mathbf{r}_0^T \mathbf{r}_0}{\mathbf{r}_0 \mathbf{A} \mathbf{r}_0}$$

Steepest descent algorithm

```
for i = 0,1, ...
```

$$\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i$$

$$\Rightarrow \alpha_i = \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{r}_i \mathbf{A} \mathbf{r}_i}$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{r}_i$$

Conjugate directions

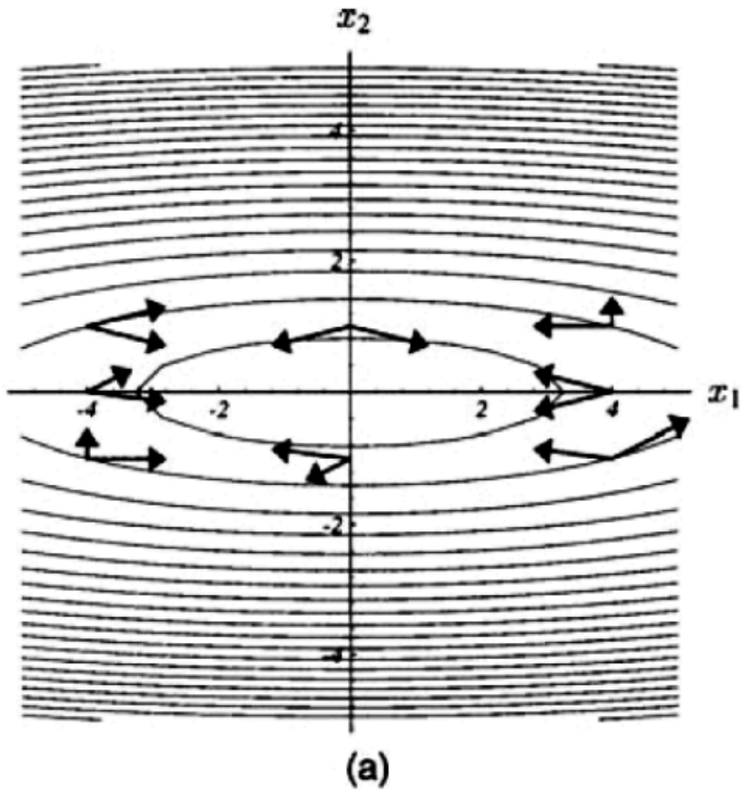
Set of orthogonalen directions $\mathbf{d}$ with

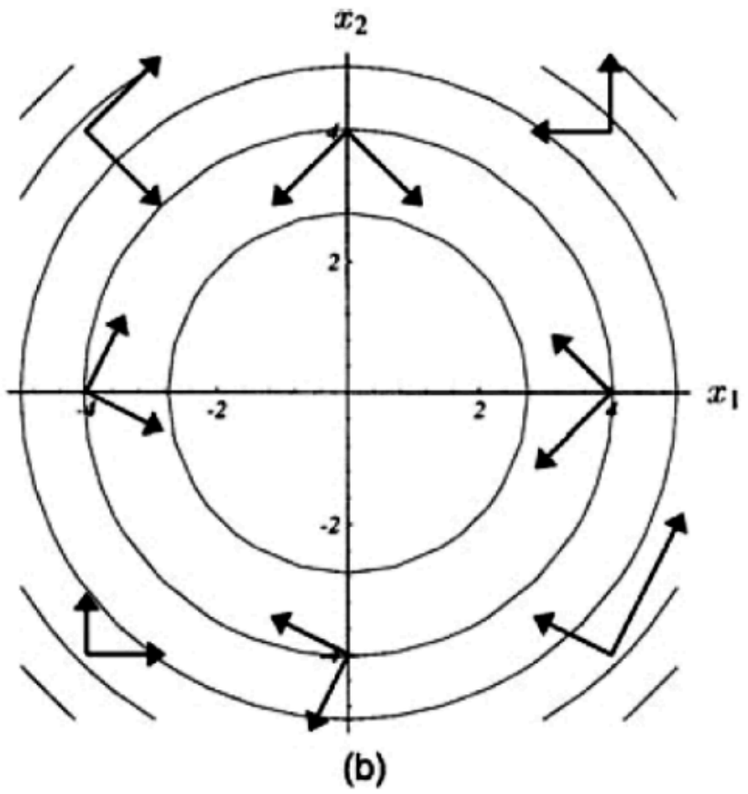
$$\mathbf{d}_i^T \mathbf{A} \mathbf{d}_j = 0 \qquad \forall i \neq j$$

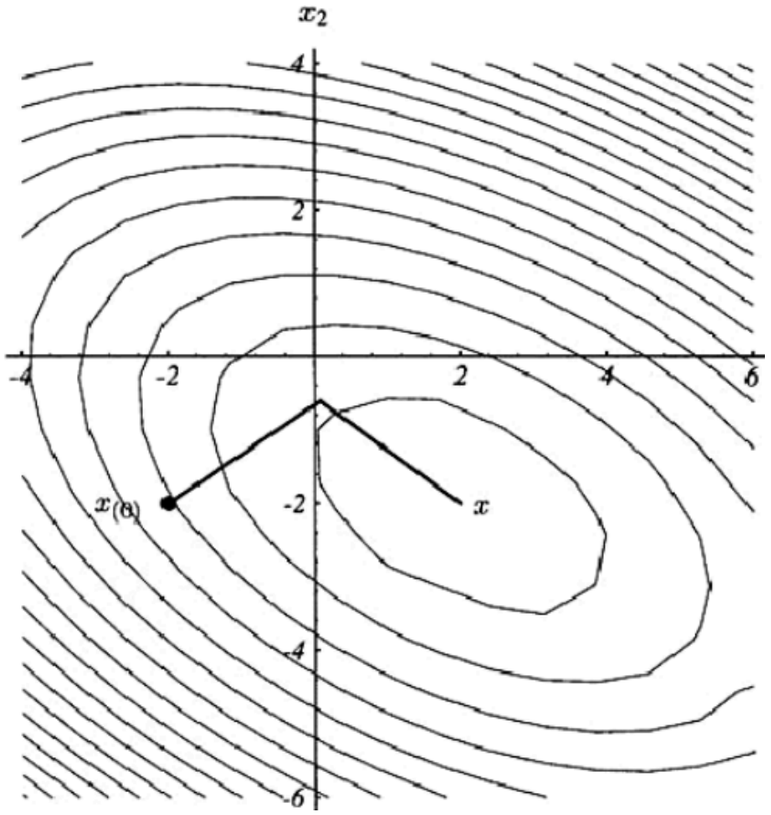
> **Note**
>
> every search direction is only used once

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{d}_i$$

Conjugate directions







Conjugate gradient (Hestenes&Stiefel, 1952)

$$\mathbf{d}_0 = \mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$$

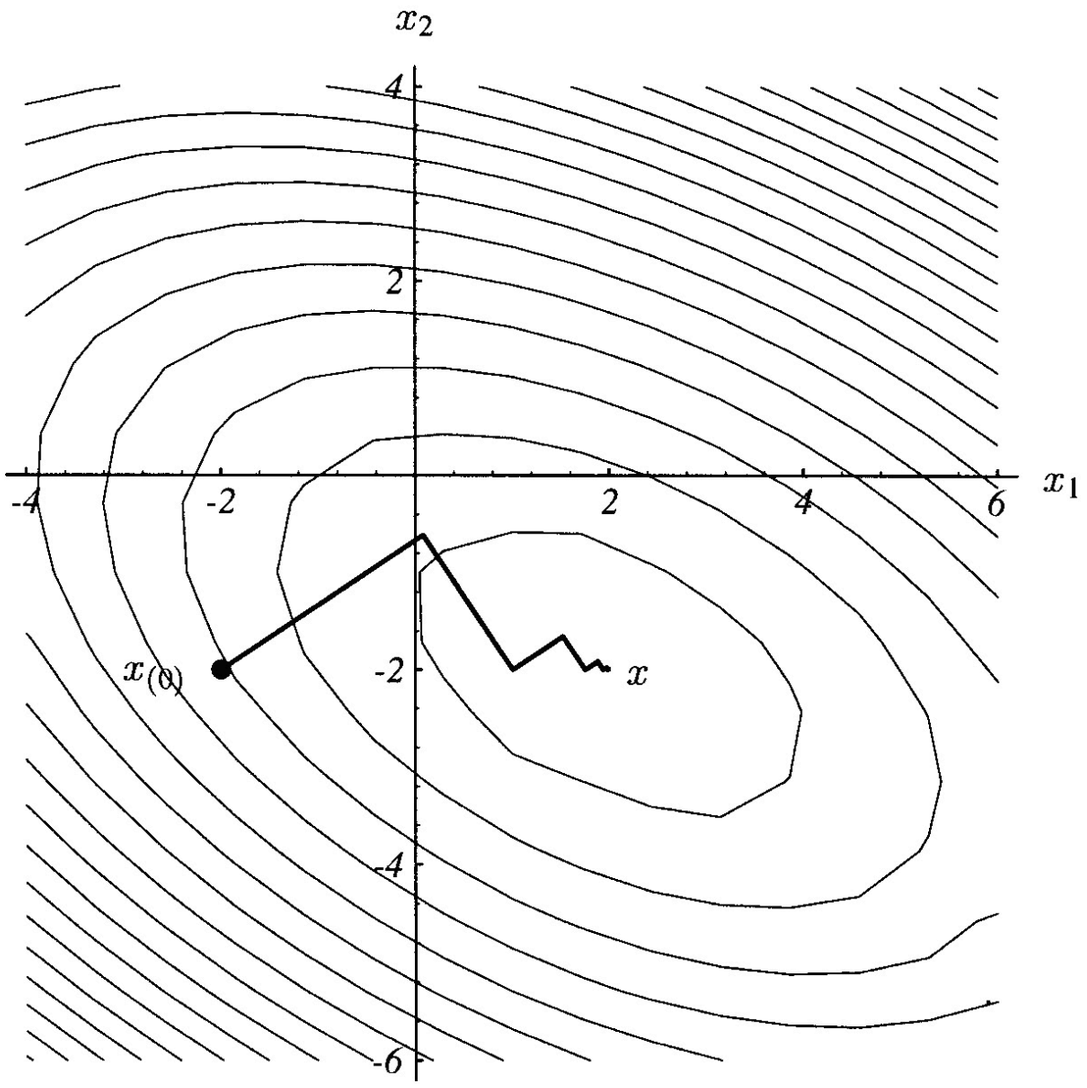
$$\alpha_i = \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{d}_i \mathbf{A} \mathbf{d}_i}$$

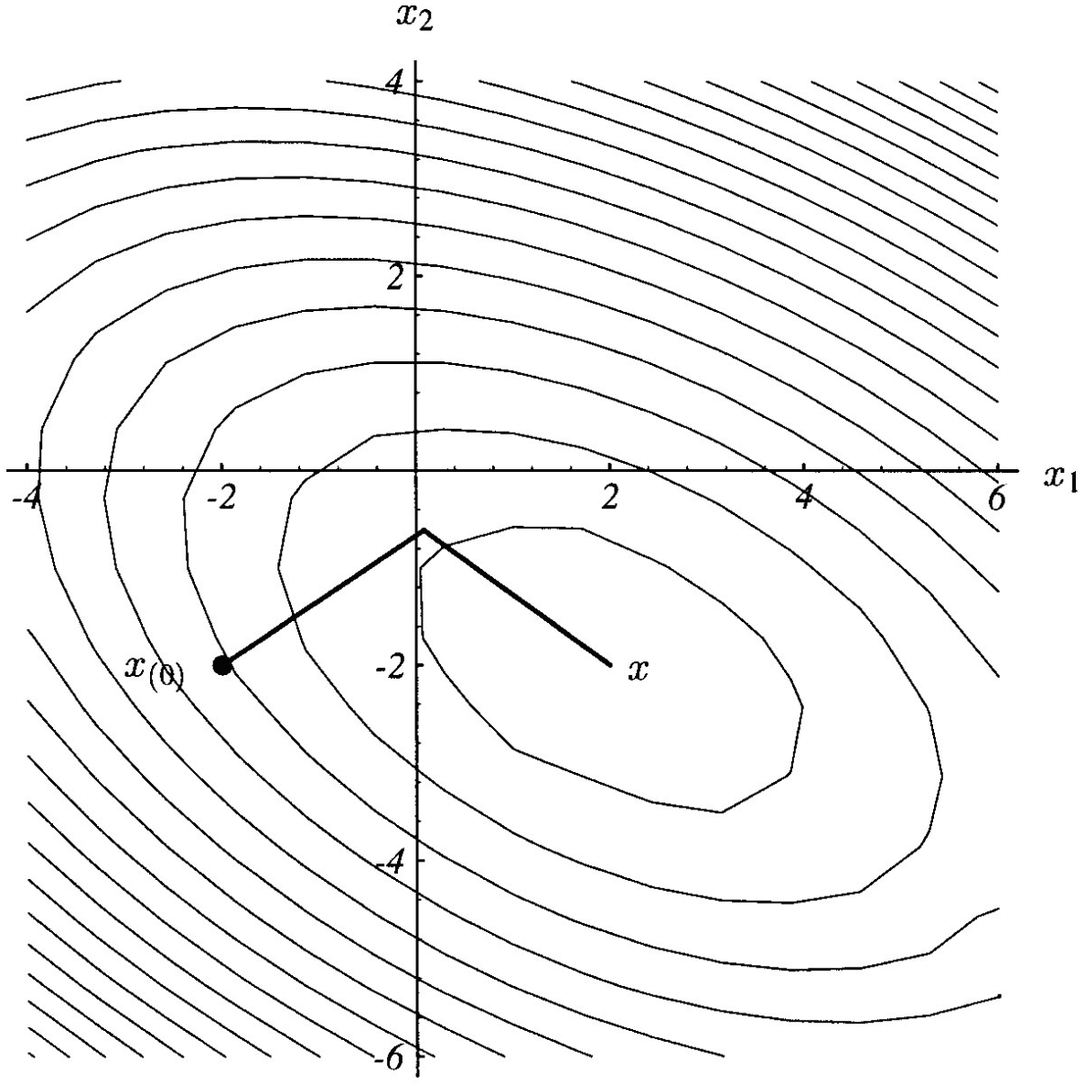
$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{d}_i$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha_i \mathbf{A} \mathbf{d}_i$$

$$d_{i+1} = \mathbf{r}_{i+1} + \mathbf{d}_{i+1} \frac{\mathbf{r}_{i+1}^T \mathbf{r}_{i+1}}{\mathbf{r}_i^T \mathbf{r}_i}$$

Steepest descent vs. conjugate gradient





Preconditioning

$\mathbf{Ax} = \mathbf{b}$ often badly conditioned (elongated) $\Rightarrow$ slow convergence

Idea: transform (precondition) equation system by preconditioner $\mathbf{K}$

$$\mathbf{K}^{-1}\mathbf{Ax} = \mathbf{K}^{-1}\mathbf{b}$$

Extreme cases: 1. $\mathbf{K} = \mathbf{I}$, cheap PC but no gain

2. $\mathbf{K} = \mathbf{A}$, perfect conditioning but expensive PC
   e.g. $\mathbf{K} = \mathrm{diag}\mathbf{A}$ or $\mathbf{K} = \epsilon\mathrm{tril}\mathbf{A}$

Incomplete Cholesky decomposition

$$\mathbf{A} \approx \mathbf{L}\mathbf{L}^T$$

with certain accuracy or sparsity