1. Numerical Treatment of PDE 1

- here: restriction to the elliptic case, Poisson equation \(-\Delta u = f\)
- several principal approaches:
  
  - **finite differences (FD)**: direct approximation of each derivative (cf. ODE)

  ![Finite differences diagram]

  \[ f(x_i) \approx \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \]

  - **finite volumes (FV)**: direct implementation of conservation laws on small volumes; esp. for flows

  ![Finite volumes diagram]

  mass conservation:

  \[ \text{flux}_1 + \text{flux}_2 - \text{flux}_3 - \text{flux}_4 = 0 \]
2. Numerical Treatment of PDE 2

• principal approaches (continued):

  – **finite elements (FEM)**: variational approach, study some weak form of the PDE

    \[- \int_{\Omega} \sum_{i} u_{i} \nabla \varphi_{i} \nabla \psi_{j} = \int_{\Omega} f \psi_{j}, \quad \forall j\]

• rough characterization:

  – **FD**: straightforward, easy to implement, poor theoretical background
  
  – **FEM**: more complicated to implement, but rich mathematical theory available
3. Finite Difference Methods 1

- Place a (in general regular) grid on the given domain.

\[ x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_{N-1} \quad x_N \]

\[ h_x \]

- Replace derivatives by difference quotients:
  
  – first derivatives: forward, backward, or central differences

\[
\frac{\partial u}{\partial x}(x_k) \triangleq \frac{u(x_{k+1}) - u(x_k)}{h_x}, \quad \frac{u(x_k) - u(x_{k-1})}{h_x}, \quad \frac{u(x_{k+1}) - u(x_{k-1})}{2h_x}
\]

- second derivatives: standard 3-point-stencil

\[
\frac{\partial^2 u}{\partial x^2}(x_k) \triangleq \frac{u(x_{k+1}) - 2u(x_k) + u(x_{k-1})}{h_x^2}
\]

- Laplacian in 2D or 3D, resp: 5-point- or 7-point-stencil, resp.

- in each inner grid point:

  * set up one difference equation

  * compute one degree of freedom per (scalar) unknown function

- concrete equation near boundary depends on bound. cond.
4. Finite Difference Methods 2

• a simple example: Poisson equation on unit square
  \[-\Delta u = f \text{ on } ]0, 1[^2\]

• use an equidistant square grid with \( h = h_x = h_y = 1/N \)

• number of inner grid points/unknowns: \( M = (N - 1)^2 \)

• linear system \( A_h u_h = f_h \) with
  
  – M×M-matrix \( A_h \) (the difference equations)
  
  – M-vectors \( f_h \) (right-hand side) and \( u_h \) (desired values of \( u \))

• \( A_h \) is a sparse matrix (band structure):
  
  – **Dirichlet b.c.**:: all diagonal elements are 4, per row between 2 and 4 non-zeroes (-1); boundary values to right-hand side
  
  – **Neumann b.c.**:: diagonal elements are 2 or 3 near the boundary and 4 elsewhere; a corresponding number of -1’s (2 to 4) in each row; pairs (1,-1) along boundary to r.h.s.
5. Finite Difference Methods 3

- order of accuracy: \( \| \, u_h - u \, \| = O(h^2) = O(N^{-2}) \)

- curse of dimension: for that, we need \( O(N^d) \) points

- possibilities of an improvement:
  - use higher-order stencils (involving more than two neighbouring grid points in each direction): matrix gets a denser structure
  - use a locally refined (adaptive) grid (problem: what to do with hanging nodes where coarser and finer subregions meet?)
6. Finite Element Methods 1

- no direct discretization of derivatives, but use some *weak form* of the PDE instead of the PDE itself

- basic steps:
  
  - **subdivision** or grid generation: decompose the underlying domain into single elements of finite size
  
  - **weak form**: the PDE has no longer to be fulfilled in each point of the domain, but only w.r.t. some scalar product with a certain class of functions:

    \[ < -\Delta u, v > = < f, v > \quad \forall v. \]

  
  - **finite dimensional ansatz space**: in the weak form, replace the continuous solution by suitable finite-dimensional approximations:

    \[ u \approx \sum_k u_k \varphi_k. \]
7. Finite Element Methods 2

- basic steps (continued):
  - finite dimensional test space: in the weak form, replace the class of test functions $\nu$ by a suitable finite set of basis functions:
    \[ \langle -\Delta \sum_k u_k \varphi_k, \psi_l \rangle = \langle f, \psi_l \rangle \quad \forall l. \]
  - system of linear equations: construct the corresponding system of linear equations by evaluating $\langle \Delta \varphi_k, \psi_l \rangle$ and $\langle f, \psi_l \rangle$ (one equation for each degree of freedom)
  - solution of the linear system: use appropriate solvers in order to obtain the finite element approximation to the PDE’s solution
8. FEM: Subdivision, Grid Generation

- subdivide the problem’s domain into *finite elements*:
  - from statics/mechanics/civil eng.: decompose complex objects into standard parts whose behaviour is easier to describe, and derive the object’s behaviour from that
  - in 3D, we get a finite element mesh consisting of
    - **elements**: 3D atoms (cubes, tetrahedra, . . .)
    - **faces**: 2D surface structures of elements (squares, triangles, . . .)
    - **edges**: 1D boundary structures of elements
    - **grid points / nodes**: where the unknowns are defined
9. **FEM: Ansatz Space**

- associated to each grid point is an *ansatz function* $\varphi_k$

- finite support: nonzero only in neighbouring elements
- all ansatz functions together span the finite-dimensional *ansatz space*, of which they form a basis
- in this ansatz space $V_h$, look for an approximation to the PDE’s solution
10. FEM: Weak Form of the PDE/Test Space

- Let $L$ denote the differential operator (e.g. Laplacian)
- Instead of $Lu = f$ on $\Omega$, consider

$$\int_{\Omega} Lu \cdot \psi_l d\Omega = \int_{\Omega} f \cdot \psi_l d\Omega \forall \psi_l$$

for some set of test functions $\psi_l$

- Method of weighted residuals or Galerkin approach
- A second finite-dimensional linear space spanned by these test functions - the test space $W_h$
- Identical test and ansatz spaces: Ritz-Galerkin approach
- Different test and ansatz spaces: Petrov-Galerkin approach
- Due to linearity: require weak form for basis functions only
- Introducing a bilinear form $a(.,.)$ and a linear form $b(.)$:

$$a(u, \psi_l) = b(\psi_l) \forall \psi_l \in W_h$$
11. FEM: Discrete Approximation and Equations

- In the above equation, replace the (exact/continuous) solution \( u \) by a discrete approximation in \( V_h \):

\[
u_{\text{approx}} = \sum_k u_k \varphi_k
\]

- in the weak form:

\[
a(u_{\text{approx}}, \psi_l) = \sum_k u_k \cdot a(\varphi_k, \psi_l) = b(\psi_l) \quad \forall \psi_l \in W_h.
\]

- neither the \( a(\varphi_k, \psi_l) \) nor the \( b(\psi_l) \) depend on the respective approximation to \( u \), but only on the problem!

- compute them once for all at the beginning: leads, as expected, to a system of linear equations \( A_h u_h = b_h \) with the so-called stiffness matrix \( A_h \)
12. FEM: Solving the Resulting SLE

• properties of $A_h$ and of our system of linear equations:
  
  – esp. in the Ritz-Galerkin case $V_h = W_h$, $A_h$ is often SPD
  
  – dream situation: $A_h$ is diagonal, i.e. all ansatz/test functions are (bi-) orthogonal (rare or hard to achieve, resp.!)  
  
  – due to the ansatz/test functions’ finite support, $A_h$ is typically sparse:

\[
a_{i,j} = a(\varphi_i, \varphi_j) = \int_{\Omega} L\varphi_i \cdot \varphi_j \, d\Omega = 0
\]

for non-overlapping supports!

(due to this and due to $A_h$’s size in practical computations, we need iterative solvers)

• strategy hence:

  – choose ansatz/test spaces with good approximation properties
  
  – construct bases of these which result in “nice” matrices and, thus, in “fast-to-solve” SLE (cf. hierarchical bases)
References