Fluid Dynamics

Chapter 7

Computational Fluid Dynamics
CFD

Ulrich H. Becker

Frankfurt University of Applied Sciences

Summer Term 2024

Basics

- **Computational Fluid Dynamics (CFD)** solves the general fluid flow equations from last chapter numerically in an **approximate way**.
- As said before, the general form of the equations is a system of **coupled, nonlinear partial differential equations**
- Consequently, the numerics involved is difficult and very much advanced.
- The functions we want to compute approximately are density ρ, pressure p, velocity **v** (a vector), and maybe temperature T and potentially many others like chemical species mass fractions in reactions.
- So to start with, just some of the basic ideas.

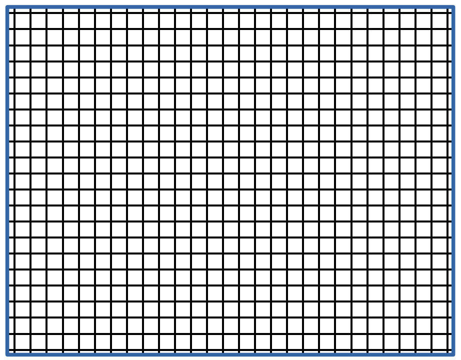
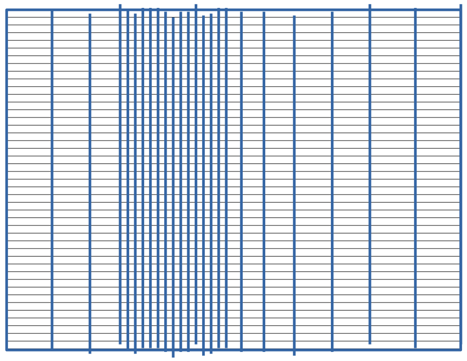
Discretization

- Solving a differential equation analytically gives a formula for the solution function that allows to compute the function value **exactly** at **every point** in space (and maybe time).
- For practical applications this may not be necessary. And way too much work, maybe even impossible.
- So the idea is to compute function values **approximately** at a **limited amount of points** in space (and maybe time).
- The only constraint is then that the approximation is good enough.

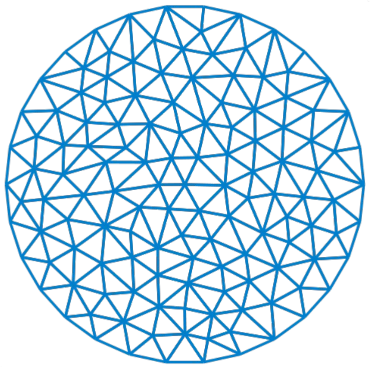
Discretization

- How to limited the amount of points? Discretization.
- Disect the domain of interest in simple shapes called „cells“ or „elements“
  - 2D: quadrangles, triangles

unstructured mesh

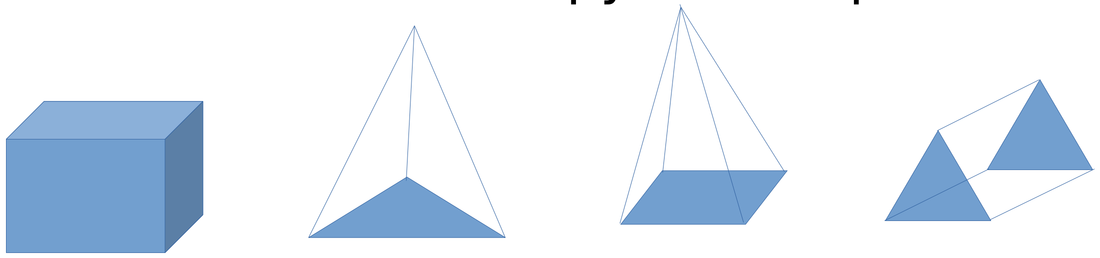



structured meshes



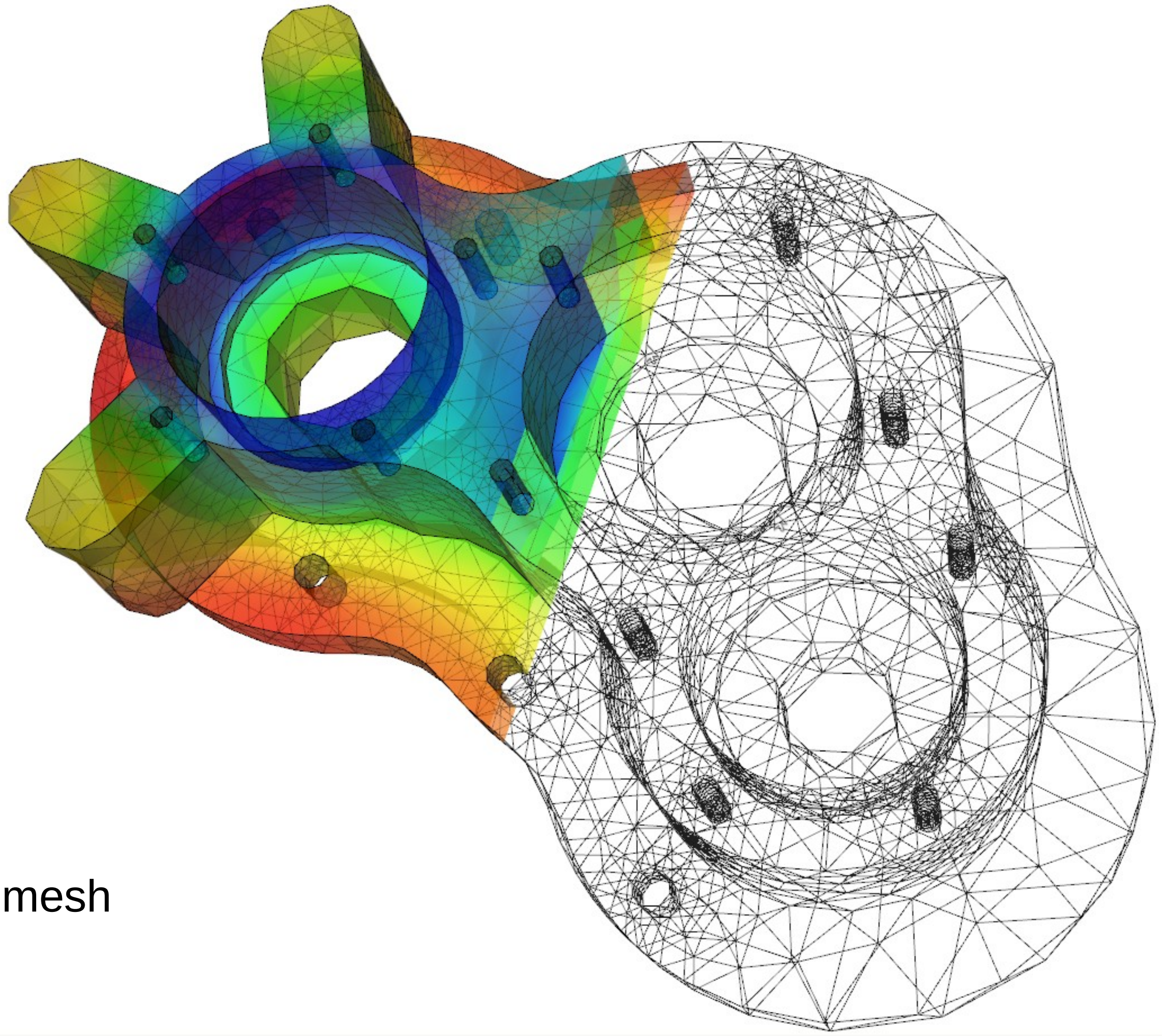
en.wikipedia.org/wiki/File:Finite_element_triangulation.svg

  - 3D: hexaeder, tetraeder, pyramids, prisms



Discretization

- 3D example for solid body



unstructured mesh

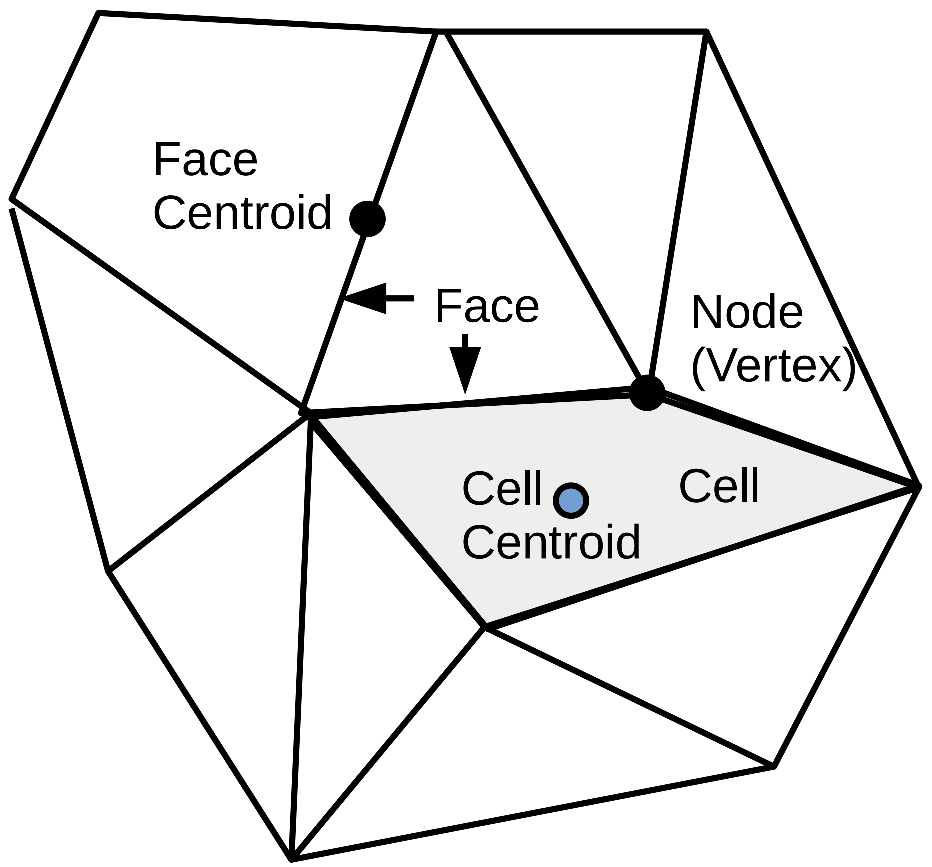
http://commons.wikimedia.org/wiki/File:Elmer-pump-heatequation.png

Discretization

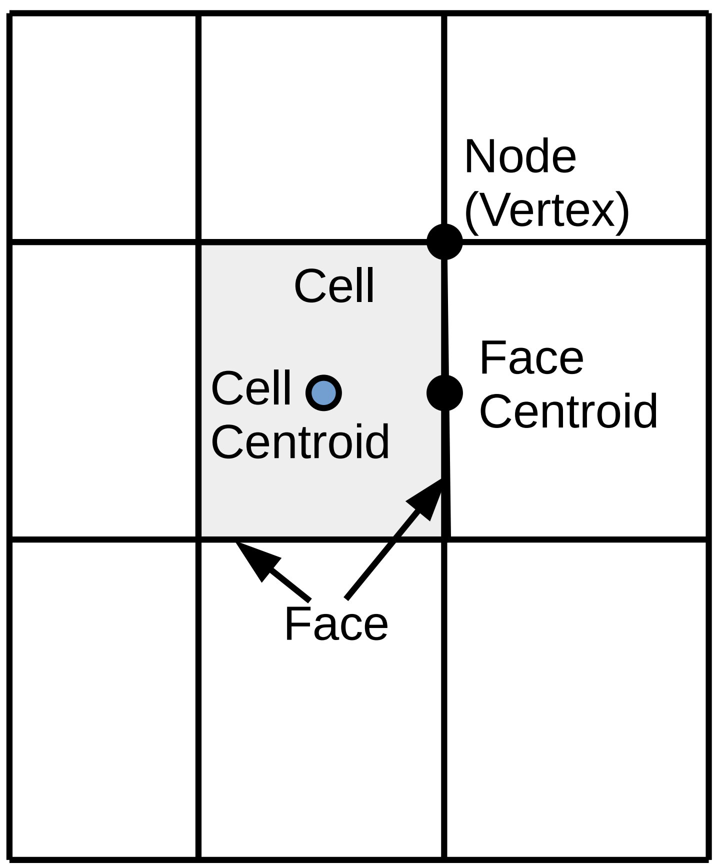
- In each cell identify points for approximating the solution functions.
  - e.g. corners, centroid, etc.
- Based on these points one can then approximate solution functions and their derivatives in different ways.
  - **Finite Differences**
    - that’s how it started …
  - **Finite Elements**
    - common in solid mechanics
  - **Finite Volumes**
    - common in CFD
  - and several others.

Discretization

- Finite Volumes: Evaluate functions at cell centroids.
  - Other points auxiliary only.



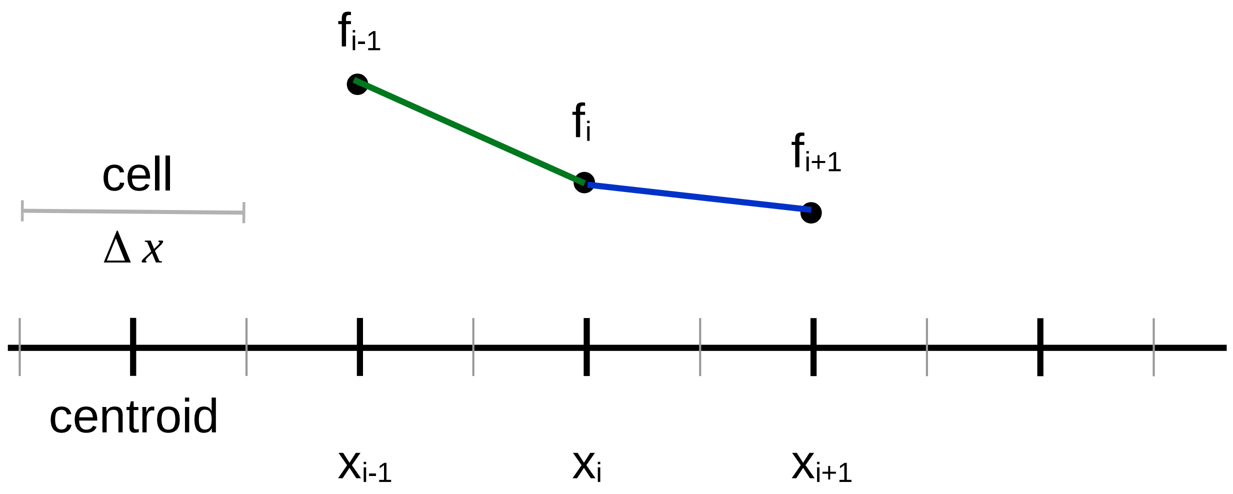
unstructured mesh



structured mesh

Discretization

- To be computed: Function values at centroids.
- But how to approximate derivatives?
- Expressed by function values. Here for 1D:



$$f'(x_i) = \frac{f_i - f_{i-1}}{\Delta x}$$ or $$f'(x_i) = \frac{f_{i+1} - f_i}{\Delta x}$$ or $$f'(x_i) = \frac{f_{i+1} - f_{i-1}}{2\Delta x}$$

to the left — to the right — average of both

Solution Procedure

- For **each cell** write down the DGL to solve in terms of these derivatives and function values
  - First, it is more complicated than that. But we leave that out for reasons of brevity.
  - Second, the above formulae are for 1st order derivatives in 1D. Formulae may be much more involved for 2D and 3D on structured meshes and are truely arduous for unstructured meshes. There is a lot to learn how to do this meaningfully.
- Thus get **one algebraic equation for each cell** that involves the function values $\mathbf{f_i}$ at the centroid and at the centroids of the neighboring cells, like $\mathbf{f_{i-1}}$ and $\mathbf{f_{i+1}}$ and so forth.
- Solve the large system of algebraic linear equations for these function values.
- **CFD is computationally very expensive!**

Solution Procedure

- Things left out in this unbelievably brief description:
  - DGLs nonlinear. How to get a linear system for f_i ?
  - DGLs form a system. Solve single DGLs one by one (segregated) or all together (coupled)?
  - How to handle boundary conditions?
  - Solving the linear systems efficiently is an art in itself (so called multigrid methods).
  - Setting up the linear algebraic systems from each cell involves an enormous amount of bookkeeping. Need efficient data structures and algorithmic workflows.
  - Performance usually requires parallel execution of the code on many CPUs, potentially across many physically different computers (clusters). Even more bookkeeping and potentially need for special and expensive hardware.
- **CFD is computationally very expensive!**

Time for examples

- Simple geometry 2D laminar internal flow
  - simple setup
  - what the results look like
  - ideal flow vs viscous flow
  - pathlines
  - Bernoulli equation (ideal and viscous)
- Simple geometry 2D external flow
  - comparison with experimental data
- Simple geometry 3D laminar internal flow
  - secondary flow pattern
- Complex geometry 3D internal flow
  - laminar flow
  - turbulent flow

Time for examples (continued)