Numerical Methods for Partial Differential Equations TA Summary

NPDE

Jonas Bachmann, Paul Fischill, Samuel Russo and Nico Graf

0 Preface

Theorem 0.3.1.19 Cauchy Schwarz Inequality
if a is symmetric positive semi-definite bilinear form, then

$$|a(u,v)| \leq a(u,u)^{\frac{1}{2}} a(v,v)^{\frac{1}{2}} \tag{1}$$

Norms

- **suprenum norm**: $\|\mathbf{u}\|_\infty = \|\mathbf{u}\|_{L^\infty(\Omega)} := \sup_{\mathbf{x}\in\Omega} \|u(x)\|$
- **L^2 norm**: $\|\mathbf{u}\|_2 = \|\mathbf{u}\|_{L^2(\Omega)} := \left(\int_\Omega \|\mathbf{u}(\mathbf{x})\|^2 dx\right)^{\frac{1}{2}}$

Theorem 0.3.2.31 Transformation rule for Integration
given to domains $\Omega, \hat{\Omega}$ and a continuous, differentiable mapping $\mathbf{\Phi} : \hat{\Omega} \to \Omega$

$$\int_\Omega f(\mathbf{x})\mathbf{dx} = \int_{\hat{\Omega}} f(\mathbf{\Phi}(\hat{\mathbf{x}}))|\det \mathrm{D}\mathbf{\Phi}(\hat{\mathbf{x}})\mathbf{d\hat{x}} \tag{2}$$

1 Second-Order Scalar Elliptic Boundary Value Problems

1.2 Quadratic Minimization Problems

Linear forms
Let V be a Vector Space over $\mathbb{R}$, $l : V \to \mathbb{R}$ is a *linear form / linear functional* $\iff$

$$l(\alpha u + \beta v) = \alpha l(u) + \beta l(v) \qquad \forall u, v \in V, \forall \alpha, \beta \in \mathbb{R} \tag{3}$$

Bilinear forms
Let V be a Vector Space over $\mathbb{R}$, $a : V \times V \to \mathbb{R}$ is a *bilinear form* $\iff$

$$\begin{aligned} &a(\alpha u_1 + u_2, \beta v_1 + v_2) \\ &\quad = \alpha\beta a(u_1, v_1) + \alpha a(u_1, v_2) + \beta a(u_2, v_1) + a(u_2, v_2) \end{aligned} \qquad \forall u_1, u_2, v_1, v_2 \in V, \forall \alpha, \beta \in \mathbb{R} \tag{4}$$

Positive definite bilinear form
A bilinear form $a : V \times V \to \mathbb{R}$ is *positive definite* if

$$u \in V \setminus \{\mathbf{0}\} \iff a(u, u) > 0 \tag{5}$$

It is *positive semi-definite* if

$$a(u, u) \geq 0 \qquad u \in V \tag{6}$$

Quadratic functional
A *quadratic functional* $J : V \to \mathbb{R}$ is

$$J(u) := \frac{1}{2} a(u, u) - l(u) + c \qquad u \in V \tag{7}$$

where $a : V \times V \to \mathbb{R}$ is symmetric bilinear form, $l : V \to \mathbb{R}$ is linear form and $c \in \mathbb{R}$ **Continuity of linear form**
A linear form $l : V \to \mathbb{R}$ is *continuous / bounded* on V, if

$$\exists C > 0 \qquad |l(v)| \leq C\|v\| \qquad \forall v \in V \tag{8}$$

1.3 Sobolev Spaces

Given a Quadratic minimization problem, i.e. a quadratic function for which we search a minimizer. Then we first need to define the space of functions in which we want to look for the solution. There is the following guidelines: *Choose the largest space such that the problem still makes sense.*
For example in Physics we generally want the solution (function that describes e.g. the shape of a string, as in the elastic string model) to be continuous. So it makes no sense to look for a minimizer with jumps.

To formulate this mathematically we need the Sobolev spaces. Because Sobolev spaces make sure that the bilinear form in the quadratic functional is well defined (i.e. finite for all functions in the space). Hence the

mathematical space in which we look for minimizers is determined by the given quadratic functional. So we can adapt the guideline ... *Choose the largest space such that the problem is well defined.*

If the quadratic minimization problem is well defined, we then get the following lemma for existence and uniqueness of minimizers.

Theorem 1.3.3.6 Existence of minimizers in Hilbert spaces
On a real Hilbert space V with norm $\|\cdot\|_a$ for any $\|\cdot\|_a$-bounded linear functional $l : V \to \mathbb{R}$ the quadratic minimization problem

$$u_* = \operatorname{argmin}_{v \in V} J(v) \qquad J(v) := \frac{1}{2}\|v\|_a^2 - l(v) \tag{9}$$

has a unique solution.

Note that here we use the norm as bilinear form with twice the same argument $a(u,u) = \|u\|_a^2$. The main point is that in (energy) minimization problems, the bilinear form of the quadratic minimization problem can be seen as the norm of some Sobolev space. This then leads to a solution if we check boundedness of the linear form.

For checking boundedness we can in many cases use the Cauchy-Schwarz and Poincaré-Friedrichs inequalities.

1.4 Linear Variational Problem

Linear variational problem
With V a vector (function) space, $\hat{V} \subset V$ an affine space, and $V_0 \subset V$ the associated subspace. The equation

$$u \in \hat{V} \qquad a(u,v) = l(v) \qquad \forall v \in V_0 \tag{10}$$

is called a (generalized) *linear variational problem*, if

- $a : V \times V_0 \to \mathbb{R}$ is bilinear form
- $l : V_0 \to \mathbb{R}$ is linear form

Knowing that a solution exists of course not enough. And solving a minimization problem over infinite dimensional spaces is not an easy task. So we reformulate the problems in a linear variational form which is then which is then already pretty close to what we will be able to solve numerically. Therefore we have the following equivalence

Theorem 1.4.1.8 Equivalence of quadratic minimization problem and linear variational problem
For a (generalized) quadratic functional $J(v) = \frac{1}{2}a(v,v) - l(v) + c$ on a vector space V and with a symmetric positive definite bilinear form $a : V \times V \to \mathbb{R}$ the following is equivalent:

1. The quadratic minimization problem for $J(v)$ has the unique minimizer $u_* \in \hat{V}$ over the affine subspace $\hat{V} = g + V_0, g \in V$
2. The linear variational problem

$$u \in \hat{V} \qquad a(u,v) = l(v) \qquad \forall v \in V_0 \tag{11}$$

has the unique solution $u_* \in \hat{V}$

Note that the test space V_0 and the trial space $\hat{V}$ can be different because in the trial space $\hat{V}$ we sometimes need to consider boundary condition (if of Dirichlet type). This ($\hat{V}$) is also the space where we want to find a solution. However in the test space V_0 we do not need to respect the boundary conditions so we set them to zero in this case (Dirichlet boundary condition).

1.5 Boundary Value Problems

Lemma 1.5.2.1 General product rule
for all $\mathbf{j} \in (C^1(\bar{\Omega}))^d, v \in C^1(\bar{\Omega})$ holds

$$\operatorname{div}(\mathbf{j}v) = v \operatorname{div} \mathbf{j} + \mathbf{j} \cdot \mathbf{grad} v \quad \text{in } \Omega \tag{12}$$

Lemma 1.5.2.4 Gauss' Theorem
let $\mathbf{n} : \partial\Omega \to \mathbb{R}^d$ denote the exterior unit normal vector field on $\partial\Omega$ and dS denote integration over a surface, we have

$$\int_\Omega \operatorname{div} \mathbf{j}(\mathbf{x}) \mathbf{dx} = \int_{\partial\Omega} \mathbf{j}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) dS(x) \quad \forall \mathbf{j} \in (C^1_{pw}(\bar{\Omega}))^d \tag{13}$$

Lemma 1.5.2.4 Green's first formula
for all vector fields $\mathbf{j} \in (C^1_{pw}(\bar{\Omega}))^d$ and functions $v \in C^1_{pw}(\bar{\Omega})$ holds

$$\int_\Omega \mathbf{j} \cdot \mathbf{grad} v \mathbf{dx} = -\int_\Omega \operatorname{div} \mathbf{j} v \mathbf{dx} + \int_{\partial\Omega} \mathbf{j} \cdot \mathbf{n} v \, dS \tag{14}$$

Lemma 1.5.3.4 Fundamental lemma of the calculus of variations
let $f \in L^2(\Omega)$ satisfy

$$\int_\Omega f(\mathbf{x}) v(\mathbf{x}) \mathbf{dx} = 0, \quad \forall v \in C_0^\infty(\Omega) \tag{15}$$

then $f \equiv 0$.

We have seen equivalence of minimization problem of a quadratic functional and linear variational problem. They are called the **weak form**, we can transform them with extra smoothness requirements to its **strong form**, i.e. into an elliptic BVP. Mainly with the help of the above lemmas.

1.7 Boundary Conditions

For 2nd-order elliptic BVPs we need boundary conditions to get a unique solution. To be more precise, we need **exactly one** of the following boundary conditions on every part of $\partial\Omega$

Fundamental boundary conditions for 2nd-order elliptic BVPs

1. **Dirichlet**: u is fixed: with $g : \partial\Omega \to \mathbb{R}$

$$u = g \quad \text{on } \partial\Omega$$

2. **Neumann**: the flux, $\mathbf{j} = -\kappa(x)\mathbf{grad}u$ through $\partial\Omega$ is fixed: with $h : \partial\Omega \to \mathbb{R}$

$$\mathbf{j} \cdot \mathbf{n} = -h \quad \text{on } \partial\Omega$$

3. **Radiation**: flux depends on u: with increasing function $\Psi : \mathbb{R} \to \mathbb{R}$

$$\mathbf{j} \cdot \mathbf{n} = \Psi(u) \quad \text{on } \partial\Omega$$

1.8 Second-Order Ellliptic Variational Problems

We have seen, how we can get from a minimization problem via a variational problem to a BVP. Now we want to move in the opposite direction, from a PDE and its boundary conditions we want to get to a variational problem. This can again be done using the lemmas from section 1.5 and consider the boundary conditions to choose a suitable (Sobolev) function space.

For Neumann problems there is a **compatibility condition**, if we choose test function $v \equiv 1$ we get the requirement

$$-\int_{\partial\Omega} h \, dS = \int_{\Omega} f \, \mathbf{dx}$$

for the existence of solutions. Additionally the solution of Neumann problems is unique only up to constants. To address this we can use the constrained function space

$$H^1_*(\Omega) := \{v \in H^1(\Omega) : \int_{\Omega} v \, \mathbf{dx} = 0\}$$

Theorem 1.8.0.20 (Second) Poincaré-Friedrichs inequality
if $\Omega \subset \mathbb{R}^d$ is bounded and connected, then

$$\exists C = C(\Omega) > 0 : \|u\|_0 \leq C \operatorname{diam}(\Omega) \|\mathbf{grad}u\|_0 \quad \forall u \in H^1_*(\Omega)/H^1_0(\Omega) \tag{16}$$

1.9 Essential and Natural boundary Conditions

Essential boundary conditions are boundary conditions which have been imposed directly on the trial space, i.e. Dirichlet BC. While Neumann BC are only enforced through the variational equation, so called natural boundary conditions.

- **Admissible Dirichlet Data**: Dirichlet boundary values need to be continuous.
- **Admissible Neumann Data**: h needs to be in $L^2(\Omega)$ (can be discontinuous)

Theorem 1.9.0.10 Multiplicative trace inequality

$$\exists C = C(\Omega) > 0 : \|u\|^2_{L^2(\partial\Omega)} \leq C\|u\|_{L^2(\Omega)} \cdot \|u\|_{H^1(\Omega)} \quad \forall u \in H^1(\Omega) \tag{17}$$

2 Finite Element Method

2.2 Galerkin Discretization

The idea is to replace an infinite function space V_0 by $V_{0,h} \subset V_0$

Theorem 2.2.1.5 Existence and uniqueness of solution of discrete variational problems
If the bilinear form $a : V_0 \times V_0 \to \mathbb{R}$ is symmetric and positive definite and the linear form $l : V_0 \to \mathbb{R}$ is continuous. Then the discrete variational Problem:

$$u_h \in V_{0,h} : a(u_h, v_h) = l(v_h), \quad \forall v_h \in V_{0,h} \tag{18}$$

has a unique *Galerkin* solution $u_h \in V_{0,h}$ satisfying the energy estimate

$$\|u\|_a \leq \sup_{v_h \in V_{0,h}} \frac{|l(v_h)|}{\|v_h\|_a} \tag{19}$$

Remember the definition of a basis: $\{b^1, \dots, b^N\} \subset V$ is a basis, if for every $v \in V$ there are unique coefficients μ_l such that $v = \sum_{l=1}^{N} \mu_l b^l$. And N agrees with the dimension of V. Now we can expand $u_h = \mu_1 b^1 + \cdots + \mu_N b^N$ and our goal is to find the coefficients μ_i.

Galerkin Discretization
Linear discrete variational problem (18) $\xrightarrow{\text{choosing basis } \mathfrak{B}_h}$ Linear system of equations $\mathbf{A}\vec{\mu} = \vec{\varphi}$

$$\textbf{Galerkin Matrix} : \quad \mathbf{A} = \left[a(b_h^k, b_h^j)\right]_{j,k=1}^{N} \in \mathbb{R}^{N,N} \tag{20}$$

$$\textbf{RHS vector} : \quad \vec{\varphi} = \left[l(b_h^j)\right]_{j=1}^{N} \in \mathbb{R}^N \tag{21}$$

$$\textbf{coefficient vector} : \quad \vec{\mu} = \left[\mu_1, \dots, \mu_N\right]^\top \in \mathbb{R}^N \tag{22}$$

Note that $\mathbf{A}_{j,k} = a(b_h^k, b_h^j) \neq a(b_h^j, b_h^k)$ in case of non-symmetric a. Of course the bilinear form a determines some properties of the Galerkin matrix. If a is symmetric and/or positive definite, the Galerkin matrix $\mathbf{A}$ will have the same properties.

The choice of $V_{0,h}$ alone determines the quality of the solution u_h. While mathematically the choice of basis $\mathfrak{B}_h$ does not matter, for solving the equation numerically, the choice is crucial as the basis determines how stable and efficiently the solution can be computed, as it determines for example the sparsity of $\mathbf{A}$.

2.3 Linear FEM in 1D

In FEM the goal is to approximate u by piecewise polynomial functions.

Mesh in one dimension
let $\Omega = [a, b]$, we equip it with $M + 1$ **nodes** resulting in the set of nodes:

$$\mathcal{V}(\mathcal{M}) = \{a = x_0 < x1 < \cdots < x_M = b\}$$

the nodes define intervals, which build up the mesh:

$$\mathcal{M} = \{]x_{j-1}, x_j[: 1 \leq j \leq M\}$$

the intervals $[x_{j-1}, x_j]$ are the **cells** of the mesh
we define local cell size $h_j = |x_j - x_{j-1}|$ and global mesh width $h_{\mathcal{M}} = \max_j h_j$

A simple space for continuous, $\mathcal{M}$-piecewise polynomial funcitons in $H_0^1(]a, b[)$:

$$V_{0,h} = S^0_{1,0}(\mathcal{M}) = \left\{ v \in C^0([a, b]) : v_{|[x_{i-1,i}]} \text{is linear}, i = 1, \ldots, M, v(a) = v(b) = 0 \right\} \tag{23}$$

$$\rightarrow N = \dim S^0_{1,0}(\mathcal{M}) = M - 1$$

The 0-superscript stands for global C^0 of the functions. The 1-subscript denotes local degree 1 polynomial and the 0-subscript denotes 0 on the boundary. The S stands for Scalar functions.

Common basis functions are the 1D tent functions:

$$b_h^j(x) = \begin{cases} (x - x_{j-1})/h_j & \text{if } x_{j-1} \leq x \leq x_j \\ (x_j - x)/h_{j+1} & \text{if } x_j \leq x \leq x_{j+1} \\ 0 & else \end{cases} \tag{24}$$

$$\rightarrow b_h^j(x_i) = \delta_{ij} \tag{25}$$

A basis satisfying condition (25) is called a **cardinal** basis. Another key property of tent functions is that their support just comprises two adjacent cells:

$$\text{supp}(b_h^j) = [x_{j-1}, x_{j+1}]$$

Where the support of a function $f : \Omega \rightarrow \mathbb{R}$ is defined as

$$\text{supp}(f) = \overline{\{x \in \Omega : f(x) \neq 0\}} \tag{26}$$

Polynomials are further nice, as they allow for easy computation of derivatives and integral, which often occur in (bi)linear forms a and l.

2.4 Linear FEM in 2D

Mesh in two dimension
meshes in 2D rely on **triangulations**. A **triangulation** $\mathcal{M}$ of Ω satisfies:

1. $\mathcal{M} = \{K_i\}$, K_i are open triangles
2. $i \neq j \rightarrow K_i \cap K_j = \varnothing$
3. $\bigcup_{i=1}^{M} \overline{K_i} = \overline{\Omega}$
4. $i \neq j \rightarrow \overline{K_i} \cap \overline{K_j}$ is either $\varnothing$, an edge from both triangles or a vertex from both

Again the vertices are called **nodes** and the triangles are the **cells**.

This definition does not allow for hanging nodes because of point 4. We can get a similar definition as (23).

$$V_{0,h} = S_1^0(\mathcal{M}) = \left\{ v \in C^0(\overline{\Omega}) \,:\, v_K(\mathbf{x}) = \alpha_K + \beta_K \cdot \mathbf{x}, \alpha_K \in \mathbb{R}, \beta_K \in \mathbb{R}^2, \mathbf{x} \in K \right\} \tag{27}$$

$$\rightarrow \dim S_1^0(\mathcal{M}) = \#\mathcal{V}(\mathcal{M}) \tag{28}$$

And $S_{1,0}^0(\mathcal{M})$ would additionally require functions to be zero on $\partial\Omega$, with

$$\dim S_{1,0}^0(\mathcal{M}) = \#\{x \in \mathcal{V}(\mathcal{M}) \,:\notin \partial\Omega\}\} \tag{29}$$

Similarly the 1D tent functions can be extended to 2D by requiring the cardinal property. This property is already enough as there is only one fixed plane through three points (i.e. the vertices of each triangle). Cardinal bases will produce sparse Galerkin matrices, as the support of the basis functions only cover the neighbouring triangles and can hence only interact with neighbouring basis functions.

Computation of Galerkin Matrix
Often bilinear forms incur integration over the whole domain. but we have seen, that the support of basis functions is only local. We can exploit this by performing only integration over the cells.

$$\mathbf{A}_{ij} = a(b_h^j, b_h^i) = \sum_{K \in \text{supp}(b_h^j) \cap \text{supp}(b_h^i)} a_{|K}(b_h^j, b_h^i) \tag{30}$$

where $a_{|K}$ is the local bilinear form over cell K.
Cell oriented assembly
To further take advantage of (30), cell oriented assembly can be performed. Go through all cells and compute $a_{|K}(b_h^j, b_j^i)$ of all basis functions, associated with cell K (element matrix) and add it to the entry of $\mathbf{A}$.

The same procedure can be applied to calculating the right hand side vector φ, just that only one basis function is involved as the rhs comes from a linear functional.

2.5 Building Blocks of General Finite Element Methods

First building block are meshes, see 2.3 and 2.4. Next we need to choose a space of functions.

Definition 2.5.2.2 Multivariate Polynomials
Space of d-variate degree (total) p polynomials:

$$\mathcal{P}_p(\mathbb{R}^d) = \left\{ \mathbf{x} \in \mathbb{R}^d \to \sum_{\alpha \in \mathbb{N}_0^d, |\alpha| \leq p} c_\alpha \mathbf{x}^\alpha, c_\alpha \in \mathbb{R} \right\} \tag{31}$$

with $\alpha = (\alpha_1, \dots, \alpha_d), \mathbf{x}^\alpha = x_1^{\alpha_1} \cdot \cdots \cdot x_d^{\alpha_d}$ and $|\alpha| = \alpha_1 + \cdots + \alpha_d$

as an example, $\mathcal{P}_2(\mathbb{R}^2) = \text{Span}\{1, x_1, x_2, x_1^2, x_2^2, x_1 x_2\}$

Theorem 2.5.2.5 Dimension of spaces of Polynomials

$$\dim \mathcal{P}_p(\mathbb{R}^d) = \binom{d+p}{p}, \quad p \in \mathbb{N}_0, q \in \mathbb{N} \tag{32}$$

in the limit of $p \to \infty$ this behaves like $O(p^d)$

Definition 2.5.2.7 Tensor Product Polynomials
Space of tensor product polynomials of degree p in each coordinate

$$\mathcal{Q}_p(\mathbb{R}^d) = \left\{ \mathbf{x} \in \mathbb{R}^d \to \sum_{l_1=0}^{p} \cdots \sum_{l_d=0}^{p} c_{l_1,\dots,l_d} x_1 \cdot \cdots \cdot x_d, c_{l_1,\dots,l_d} \in \mathbb{R} \right\} \tag{33}$$

$$= \text{Span}\left\{ \mathbf{x} \to p_1(x_1) \cdot \cdots \cdot p_d(x_d), p_i \in \mathcal{P}_p(\mathbb{R}) \right\} \tag{34}$$

as an example, $\mathcal{Q}_2(\mathbb{R}^2) = \text{Span}\{1, x_1, x_2, x_1 x_2, x_1^2, x_1^2 x_2, x_1^2 x_2^2, x_1 x_2^2, x_2^2\}$

Theorem 2.5.2.8 Dimension of spaces of tensor product Polynomials

$$\dim \mathcal{Q}_q(\mathbb{R}^d) = (p+1)^d \tag{35}$$

Finally we need **locally** supported basis functions. These basis $\mathfrak{B}_h = b_h^1, \cdots, b_h^N$ should satisfy the following constraints:

1. $\mathfrak{B}_h$ is a basis of V_h, hence $\dim\mathfrak{B}_h = \dim V_h$.
2. each b_h^i is associated with a single mesh geometry entity (cell/edge/face/vertex).
3. each b_h^i is only locally supported, i.e. only nonzero in adjacent cells.

2.6 Lagrangian Finite Element Methods

Remember eq. 2.3 and 2.4 as two examples of finite element spaces. Generally they are called Lagrangian FE spaces:

Definition 2.6.1.1 Simplical Lagrangian finite element spaces

$$S_p^0(\mathcal{M}) = \left\{ v \in C^0(\overline{\Omega}) : v_{|K} \in \mathcal{P}_p(K), \forall K \in \mathcal{M} \right\} \tag{36}$$

This space is well suited for triangular meshes, as the local dimension $\binom{d+p}{p} = \binom{2+p}{p}$ (in 2D) is the same as the amount of vertices in a triangle and its interpolation nodes. The local basis functions of S_1^0 are the barycentric coordinate functions. In S_2^0, the local basis functions are linear coombinations of barycentric coordinate functions:

$$\begin{aligned} b_K^1 &= (2\lambda_1 - 1)\lambda_1, \quad & b_K^4 &= 4\lambda_1\lambda_2, \\ b_K^2 &= (2\lambda_2 - 1)\lambda_2, \quad & b_K^5 &= 4\lambda_2\lambda_3, \\ b_K^3 &= (2\lambda_3 - 1)\lambda_3, \quad & b_K^6 &= 4\lambda_1\lambda_3, \end{aligned}$$

where the local basis functions 1-3 are associated with vertices and 4-6 with edges.

Analogously, the following space is well suited for quadrilaterals:

Definition 2.6.2.5 Tensor product Lagrangian finite element spaces

$$S_p^0(\mathcal{M}) = \left\{ v \in C^0(\overline{\Omega}) : v_{|K} \in \mathcal{Q}_p(K), \forall K \in \mathcal{M} \right\} \tag{37}$$

Note, the only difference is the local polynomial space, $\mathcal{Q}_p(K)$ instead of $\mathcal{P}_p(K)$. This space works well for quadrilaterals, as the dimension $(p+1)^d = (p+1)^2$ (in 2D) is again the same as the amount of vertices and its interpolation points.

Of course these spaces can be mixed, i.e. on mixed meshes where the definition 2.6 is used on triangles and 2.6 on quadrilaterals.

2.7 Implementation of Finite Element Methods

Remember the principle of cell orientated assembly. The goal is to rely mostly on local computations. To perform cell oriented assembly, a map from local to global indices is needed. In LehrFEM++ this is the job of the dofhandler (`lf::assemble::DofHandler` documentation). It provides the following main methods:

- `NumDofs()`, returns the total number of global basis functions, the dimension of the FE space.
- `NumLocalDofs(const lf::mesh::Entity &)`, returns the number of global basis functions covering any geometric entity.
- `GlobalDofIndices(const lf::mesh::Entity &)`, returns an array of indices of the global basis function covering the given entity.
- `NumInteriorDofs(const lf::mesh::Entity &)`, returns the number of global basis function, associated with the given entity.
- `InteriorGlobalDofIndices(const lf::mesh::Entity &)`, similar to `GlobalDofIndices`, but returns only the indices of the global basis functions, associated with the given entity.
- `Entity(gdof_idx_t dofn)`, returning the entity, associated with the global index `dofnum`.

Instead of dimension, in LehrFEM++ the concept of **co-dimension** is used. instead of going from a point with dimension 0 to a triangle with dimension 2, the co-dimension is the other way around. The highest dimension entity has co-dimension 0. This ensures, that cells are always of co-dimension 0.

To assemble the Galerkin matrix, `lf::assemble::AssembleMatrixLocally` (docs) can be used. To use it, we need element matrix providers (docs). These are construct, which provide the element matrix for given bilinear forms. Some common bilinear forms are already implemented.

- $\int_K \alpha(x)\mathbf{grad}\, u \cdot \mathbf{grad}\, v dx$ is implemented in `lf::fe::DiffusionElementMatrixProvider`
- $\int_K \gamma(x)\, u\, v\, dx$ is implemented in `lf::fe::MassElementMatrixProvider`
- $\int_e \gamma(x)\, u\, v\, dS$ is implemented in `lf::fe::MassEdgeMatrixProvider`. Note the integration over edge and not cell.
- $\int_K \alpha(x)\mathbf{grad}\, u \cdot \mathbf{grad}\, v dx + \int_K \gamma(x)\, u\, v dx$ combined is implemented in lf::uscalfe::ReactionDiffusionElementMatrixProv
- $\int_K f(x) v\, dx$ is implemented by lf::fe::ScalarLoadElementVectorProvider
- $\int_e f(x) v\, dS$ is implemented by lf::fe::ScalarLoadEdgeVectorProvider. Note again the integration over edge.

Note that the last two are actually element vector providers.

Lemma 2.7.5.5 Integration of powers of barycentric coordinate functions
For d-simplex (line in 1D, triangle in 2D, tetrahedron in 3D) with barycentric coordinate functions $\lambda_1, \dots, \lambda_{d+1}$

$$\int_K \lambda_1^{\alpha_1} \cdot \dots \cdot \lambda_{d+1}^{\alpha_{d+1}} dx = d!|K| \frac{\alpha_1! \cdot \dots \cdot \alpha_{d+1}!}{(\alpha_1 + \dots + \alpha_{d+1} + d)!}, \quad \alpha_i \in \mathbb{N} \tag{38}$$

Quadrature rule

$$\int_K f(x)dx \approx \sum_{l=1}^{P_K} w_l^K f(\zeta_l^K), \quad w_l^K \to \text{weights}, \zeta_l^K \to \text{(quadrature) nodes} \tag{39}$$

Order of a quadrature rule: a quad rule is of order q if

- for a simplex K is exact for all polynomials $f \in \mathcal{P}_{p-1}(\mathbb{R}^d)$
- for a tensor product element K is exact for all polynomials $f \in \mathcal{Q}_{p-1}(\mathbb{R}^d)$

Lemma 2.7.5.14 Affine transformation of triangles
For any triangle K, $|K| > 0$, there is a unique affine transformation $\Phi_K(\hat{x}) = F_K\hat{x} + \tau_K$, with $K = \Phi_K(\hat{K})$ and $\hat{K}$ the unit triangle.

This is nice as that allows us to transform th unit triangle to any triangle and perform easy integration. Additionally, Φ_K can be computed straight forward:

$$\text{let } K \text{ be a triangle with vertices } a^1, a^2, a^3 \to \Phi_K(\hat{x}) = \begin{bmatrix} a_1^2 - a_1^1 & a_1^3 - a_1^2 \\ a_2^2 - a_2^1 & a_2^3 - a_2^2 \end{bmatrix} \hat{x} + \begin{bmatrix} a_1^1 \\ a_2^1 \end{bmatrix} \tag{40}$$

Essential Boundary Conditions
Remember from 1.9, essential boundary conditions are Dirichlet boundary conditions, i.e. $u = g$ on $\partial\Omega$, and can be solved with the offset function trick. This trick can also be used in FEM. Suppose:

$$A = \begin{bmatrix} A_0 & A_{0\partial} \\ A_{0\partial}^\top & A_{\partial\partial} \end{bmatrix} \tag{41}$$

where A_0 is the Galerkin matrix for $\mathcal{S}^0_{p,0}(\mathcal{M})$, $(A_{0\partial})_{ij} = a(b_h^j, b_h^i)$, where b_h^j belongs to the boundary and b_h^i to the interior. Similarly, $A_{\partial\partial}$ consists only of entries calculated from basis functions of the boundary. Then we want to solve:

$$\begin{bmatrix} A_0 & A_{0\partial} \\ A_{0\partial}^\top & A_{\partial\partial} \end{bmatrix} \begin{bmatrix} \mu_0 \\ \mu_\partial \end{bmatrix} = \begin{bmatrix} \varphi \\ \varphi_\partial \end{bmatrix} \tag{42}$$

where μ_∂ are the coefficients of the basis expansion of g on the boundary. Hence its known. We only need to solve for μ_0 which results in

$$A_0\mu_0 = \varphi - A_{0\partial}\mu_\partial \tag{43}$$

This can be done in LehrFEM++ with lf::assemble::FixFlaggedSolutionComponents or lf::assemble::FixFlaggedSolutionCompAlt. Both modify, the matrix A and rhs vector b such that it corresponds to the above equation, but do it slightly different (see the docs).

2.8 Parametric Finite Element Methods

Definition 2.8.1.2 Pullback
Given domains $\Omega, \hat{\Omega} \subset \mathbb{R}^d$ and a bijective mapping $\Phi : \hat{\Omega} \to \Omega$, the *pullback* of a function $u : \Omega \to \mathbb{R}$ is a function on $\hat{\Omega}$ defined by $(\Phi^* u)(\hat{x}) = (u \circ \Phi)(\hat{x}) := u(\Phi(\hat{x})), \hat{x} \in \hat{\Omega}$

An example for this is the affine transformation of triangles (Lemma 2.7.5.14) above. Note that in the following we will use $\hat{x}$ for an element that "lives" in a reference Triangle (or Quadrilateral). That is the triangle with corners ((0,0), (1,0), (0,1)) and the square with edge length one rooted at 0.

Note that all bilinear forms and linear forms in this course consists of integrals. Hence we will sooner or later use quadrature to approximate the integrals. But in the literature quadrature rules are defined over the aforementioned reference elements. Hence we want to make a change of variables in the integrals of the linear and bilinear forms, such that we can apply these quadrature rules. That's where we need the pullback functions.

For the simple example of e mass matrix we get (by some multidimensional analysis)

$$\int_K b_{Ki}(x) b_{Kj}(x)\, dx = \int_{\hat{K}} (\Phi_K^* b_{Ki})(\hat{x})(\Phi_K^* b_{Kj})(\hat{x}) \sqrt{\det\left(D\Phi_K^T(\hat{x}) D\Phi_K(\hat{x})\right)}\, dx, \tag{44}$$

And for the diffusion matrix

$$\int_K \nabla b_{Ki}(x) \nabla b_{Kj}(x)\, dx = \int_{\hat{K}} \nabla_x(\Phi_K^* b_{Kj})(\hat{x}) \nabla_x(\Phi_K^* b_{Kj})(\hat{x}) \sqrt{\det\left(D\Phi_K^T(\hat{x}) D\Phi_K(\hat{x})\right)}\, dx. \tag{45}$$

Where $b_{Kj} : K \to \mathbb{R}$ is one of the basis functions. E.g. a barycentric function on K.

Both equations follow form the substitution rule of multivariate analysis. But just a note for the term $\det\left(D\Phi_K^T(\hat{x}) D\Phi_K(\hat{x})\right)$: It is only necessary if Ω and $\hat{\Omega}$ do not live in the same space, i.e. if for example $\Omega \subset \mathbb{R}^3$ describes a 2-d plane in a 3-d world and $\hat{\Omega} \subset \mathbb{R}^2$. Then we will have $D\Phi_K \in \mathbb{R}^{3x2}$ and hence $\det\left(\Phi_K\right)$ is not defined. But no worries all you need with respect to this monster is implemented in `lf::geometry::IntegrationElement`.

The next thing which needs clarification are the $\nabla_x(\Phi_K^\star b_{Kj})(\hat{x})$, this is because the computation of these is in this form not really clear because $\nabla_x(\Phi_K^\star b_{Kj})(\hat{X}) = \nabla_x b_{Kj}(x)$ by the definition of Φ_K. But $\nabla_x b_{Kj}(x)$ depends on the shape of K, hence it is not clear how these gradients will look like in the general case.

Therefore we use

Lemma 2.8.3.10 Transformation formula for gradients
For differentiable $u : K \to \mathbb{R}$ and any diffeomorphism $\Phi_K : \hat{K} \to K$ we have

$$(\nabla_{\hat{x}}(u \circ \Phi_K))(\hat{x}) = (D\Phi_K(\hat{x}))^T((\nabla_x u) \circ \Phi_K)(\hat{x}) = (D\Phi_K(\hat{x}))^T \nabla_x u(x) \tag{46}$$

Note that the brackets around the gradients are important because this implies

$$\nabla_x b_{Kj}(x) = (D\Phi_K(\hat{x}))((D\Phi_K(\hat{x}))^T(D\Phi_K(\hat{x})))^{-1}\nabla_{\hat{x}} b_{Kj}(\Phi_K(\hat{x})) \tag{47}$$
$$= (D\Phi_K(\hat{x}))((D\Phi_K(\hat{x}))^T(D\Phi_K(\hat{x})))^{-1}\nabla_{\hat{x}}\hat{b}(\hat{x}) \tag{48}$$

Here $\hat{b}(\hat{x})$ is the basis reference basis function on the reference shape $\hat{K}$. I.e. we can compute $\nabla_{\hat{x}}\hat{b}(\hat{x})$ easily by hand.

Note that in the script $(D\Phi_K(\hat{x}))((D\Phi_K(\hat{x}))^T(D\Phi_K(\hat{x})))^{-1} = (D\Phi_K(\hat{x}))^{-T}$ with some abuse of notation. And in case $D\Phi_K(\hat{x}) \in \mathbb{R}^{d\times d}$ i.e. it is a squared matrix, then we have that $D\Phi_K(\hat{x})$ is invertable and $(D\Phi_K(\hat{x}))^{-T}$ is accurate. So the long term only matters, when we have as above Ω and $\hat{\Omega}$ do not live in the same space, for example if $\Omega \subset \mathbb{R}^3$ describes a 2-d plane in a 3-d world and $\hat{\Omega} \subset \mathbb{R}^2$. But the same applies here you can in any case just use `lf::geometry::JacobianInverseGramian` which will return $(D\Phi_K(\hat{x}))^{-T}$ in any case.

Bilinear Transformation for Quadrilaterals
let $\{\mathbf{a^1}, \mathbf{a^2}, \mathbf{a^3}, \mathbf{a^4}\}$ be the ordered corners of a quadrilateral. Then

$$\Phi_K(\hat{x}) = (1-\hat{x}_1)(1-\hat{x}_2)\mathbf{a^1} + \hat{x}_1(1-\hat{x}_2)\mathbf{a^2} + (1-\hat{x}_1)\hat{x}_2\mathbf{a^3} + (1-\hat{x}_1)\hat{x}_2\mathbf{a^4} \tag{49}$$
$$= \begin{bmatrix}\alpha_1 + \beta_1\hat{x}_1 + \gamma_1\hat{x}_2 + \delta_1\hat{x}_1\hat{x}_2 \\ \alpha_2 + \beta_2\hat{x}_1 + \gamma_2\hat{x}_2 + \delta_2\hat{x}_1\hat{x}_2\end{bmatrix} \tag{50}$$

with

$$\begin{bmatrix}\alpha_1 \\ \alpha_2\end{bmatrix} = \mathbf{a^1}, \begin{bmatrix}\beta_1 \\ \beta_2\end{bmatrix} = \mathbf{a^2} - \mathbf{a^1}, \begin{bmatrix}\gamma_1 \\ \gamma_2\end{bmatrix} = \mathbf{a^4} - \mathbf{a^1}, \begin{bmatrix}\delta_1 \\ \delta_2\end{bmatrix} = \mathbf{a^4} - \mathbf{a^3} - \mathbf{a^2} + \mathbf{a^1}$$

3 FEM: Convergence and Accuracy

3.1 Abstract Galerkin Error Estimates

The main point here is Optimality of Galerkin solutions.

Theorem 3.1.3.7 Cea's Lemma
Under some assumptions that guarantee the existence of a unique solution we have

$$\|u - u_h\|_a = \inf_{v_h \in V_{0,h}} \|u - v_h\|_a \tag{51}$$

So the solution we get form FEM are the best with respect to the energy norm, in the chosen discrete subspace.

Next we want to discuss the types of refinement.

h-refinement
Replace the mesh $\mathcal{M}$ (underlying $V_{0,h}$) with a finer mesh $\mathcal{M}'$ (underlying larger discrete trial space $V'_{0,N'}$)

p-refinement
Replace $V_{0,h} := S^0_p(\mathcal{M}), p \in \mathbb{N}$, with $V'_{0,h} := S^0_{p+1}(\mathcal{M}) \implies V_{0,h} \subset V'_{0,h}$

So h-refinement refines the mesh (smaller and smaller triangles). And p-refinement chooses more powerful basis functions (start with linear then quadratic, ...). The h in h-refinement corresponds to the

Definition 3.2.1.4 Mesh width
Given a mesh $\mathcal{M} = \{K\}$, the *mesh width* $h_{\mathcal{M}}$ is defined as

$$h_{\mathcal{M}} := \max\{\mathrm{diam}K \,:\, K \in \mathcal{M}\} \tag{52}$$
$$\mathrm{diam}K := \max\{|p-q| \,:\, p,q \in K\} \tag{53}$$

3.2 Empirical (Asymptotic) Convergence of Lagrangian FEM

As in NCSE, there are basically two types of convergence, algebraic and exponential. We refer to the number of basis functions (dimension of the trial space) as N. And we study the behaviour of errors with $N \to \infty$.

Definition 3.2.2.1 Types of convergence

$\|u - u_N\| = \mathcal{O}(N^{-\alpha}), \alpha > 0$ is called *algebraic* convergence with rate α.

$\|u - u_N\| = \mathcal{O}(\exp(-\gamma N^{\delta}), \gamma, \delta > 0$ is called *exponential* convergence.

Note that in the case of h-refinement we get the relation between N and h is given by

Equation 3.3.5.16

$$N = \dim S^0_p(\mathcal{M}) \approx p^d h_{\mathcal{M}}^{-d} \implies \frac{h_{\mathcal{M}}}{p} \approx N^{-\frac{1}{d}} \tag{54}$$

Where p are the dimensions of the local basis functions i.e. for the linear basis function we have $p = 1$. and d is the dimension of the underlying space Ω.

For example in the case where we have $\Omega \subset \mathbb{R}^2$ and piecewise linear basis function, we get $h_{\mathcal{M}}^{-2} \approx N$

3.3 A Priori (Asymtotic) Finite Element Error Estimates

Linear interpolation error 1D
Using the linear interpolant I_1 we want to study the interpolation error $u - \mathrm{I}_1 u$. The following interpolation error estimates can be used for sufficiently smooth functions u:

$$\|u - \mathrm{I}_1 u\|_{L^\infty(]a,b[)} \leq \frac{1}{4} h_{\mathcal{M}}^2 \|u''\|_{L^\infty(]a,b[)} \tag{55}$$

$$\|u - \mathrm{I}_1 u\|_{L^2(]a,b[)} \leq h_{\mathcal{M}}^2 \|u''\|_{L^2(]a,b[)} \tag{56}$$

$$|u - \mathrm{I}_1 u|_{H^1(]a,b[)} \leq h_{\mathcal{M}} \|u''\|_{L^2(]a,b[)} \tag{57}$$

Linear interpolation error 2D
In 2D linear interpolation corresponds to tent functions. $\mathrm{I}_1 u = \sum_{p \in \mathcal{V}(\mathcal{M})} u(p) b^p$, where b^p is the tent function associated with point p. After some derivations one arrives at:

$$\|u - \mathrm{I}_1 u\|_{L^2(\Omega)} \leq \sqrt{\frac{3}{8}} h_{\mathcal{M}}^2 \left\| \left\| D^2 u \right\|_F \right\|_{L^2(\Omega)} \tag{58}$$

and

$$\|\mathbf{grad}(u - \mathrm{I}_1 u)\|_{L^2(\Omega)} \leq \sqrt{\frac{3}{32}} \rho_{\mathcal{M}} h_{\mathcal{M}} \left\| \left\| D^2 u \right\|_F \right\|_{L^2(\Omega)} \tag{59}$$

These bounds might seem complicated but D^2 is just the Hessian, hence $D^2 u = \begin{bmatrix} \frac{\partial^2 u}{\partial x_1^2} & \frac{\partial^2 u}{\partial x_1 \partial x_2} \\ \frac{\partial^2 u}{\partial x_1 \partial x_2} & \frac{\partial^2 u}{\partial x_2^2} \end{bmatrix}$.

The "second" derivative, as above. $\|\cdot\|_F$ is the Frobenius norm, $\|A\|_F = \left(\sum_{i,j} A_{i,j}^2 \right)^{\frac{1}{2}}$.

And $\rho_{\mathcal{M}}$ is the shape regularity measure of the mesh $\mathcal{M}$, defined as $\rho_{\mathcal{M}} = \max_{K \in \mathcal{M}} \frac{h_{\mathcal{M}}^2}{|K|}$ for a triangular mesh.

To get rid of this cumbersome notation, we can introduce more Sobolov spaces.

Definition 3.3.3.1 Higher order Sobolov spaces/norms
The m-th order Sobolev norm is defined as

$$\|u\|_{H^m(\Omega)}^2 = \sum_{k=0}^{m} \sum_{\alpha \in \mathbb{N}^d, |\alpha| = k} \int_\Omega |D^\alpha u|^2 dx, \text{ where } \quad D^\alpha u = \frac{\partial^{|\alpha|} u}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \tag{60}$$

Hence we can define the m-th Sobolov space as

$$H^m(\Omega) = \left\{ v : \Omega \to \mathbb{R} : \|v\|_{H^m(\Omega)} < \infty \right\} \tag{61}$$

Definition 3.3.3.3 Higher order Sobolov semi-norms
The m-th order Sobolev semi-norm is defined as

$$|u|^2_{H^m(\Omega)} = \sum_{\alpha \in \mathbb{N}^d, |\alpha|=m} \int_\Omega |D^\alpha u|^2 dx \tag{62}$$

Remember the multidex α defined in 2.5. Using this new notation, we can rewrite the error bounds from 3.3 as

$$\|u - \mathrm{I}_1 u\|_{L^2(\Omega)} \leq \sqrt{\frac{3}{8}} h^2_{\mathcal{M}} |u|_{H^2(\Omega)} \tag{63}$$

and

$$\|\mathbf{grad}(u - \mathrm{I}_1 u)\|_{L^2(\Omega)} \leq \sqrt{\frac{3}{32}} \rho_{\mathcal{M}} h_{\mathcal{M}} |u|_{H^2(\Omega)} \tag{64}$$

Now the question is, whether these bounds are sharp. After some investigation, the bound might not be very smooth, but a sharper bound can be made for Lagrangian finite elements:

Theorem 3.3.5.6. Best approximation error estimates for Lagrangian finite elements
given a triangular mesh $\mathcal{M}$

$$\inf_{v_h \in S^0_p(\mathcal{M})} \|u - v_h\|_{H^1(\Omega} \leq C \cdot \left(\frac{h_{\mathcal{M}}}{p}\right)^{min\{p,k-1\}} \|u\|_{H^k(\Omega)} \quad \forall u \in H^k(\Omega) \tag{65}$$

We might not know the constant C and/or $\|u\|_{H^k(\Omega)}$ but we know p and $h_{\mathcal{M}}$ as they are imposed by the choice of function space and mesh. Remember the concept of refinement 3.1, we can adjust these values. And from Eq. 3.2 we know $\frac{h_{\mathcal{M}}}{p} \in \mathcal{O}\left(N^{-\frac{1}{d}}\right)$ in case of simplicial meshes. Hence the error displays **algebraic** convergence with rate $\frac{min\{p,k-1\}}{d}$. What still remains a question, is k, the smoothness of the solution u.

3.4 Elliptic regularity

Theorem 3.4.0.2 Smooth elliptic lifting theorem
For domains Ω with smooth boundaries $\partial\Omega$, i.e. no corners and sufficiently smooth σ, if

$$u \in H^1_0(\Omega) \quad \text{and} \quad -\operatorname{div}(\sigma \mathbf{grad}(u)) \in H^k(\Omega) \tag{66}$$

or

$$u \in H^1(\Omega),\ -\operatorname{div}(\sigma \mathbf{grad}(u)) \in H^k(\Omega) \quad \text{and} \quad \mathbf{grad}(u) \cdot \mathbf{n} \text{ on } \partial\Omega \tag{67}$$

holds, then $u \in H^{k+2}(\Omega)$ and

$$\|u\|_{H^{k+2}(\Omega)} \leq C \cdot \|\operatorname{div}(\sigma \mathbf{grad}(u))\|_{H^k(\Omega)} \tag{68}$$

This tells us, that when solving $-\text{div}(\sigma \mathbf{grad}(u)) = f$ and the source term f is in $H^k(\Omega)$, the solution u will be in $H^{k+2}(\Omega)$ (of course under the right assumptions).

The theorem requires smooth domains, but our meshes will have corners, so what can be done there? As long as the domain and all cells are convex, the above still holds. That is encapsulated in the following:

$$\text{if } \Omega \text{ convex}, u \in H_0^1(\Omega), \Delta u \in L^2(\Omega) \rightarrow u \in H^2(\Omega) \tag{69}$$

3.5 Variational Crimes

What are variational crimes? Instead of solving $u_h \in V_{0,h} : a(u_h, v_h) = l(v_h), \forall v_h \in V_{0,h}$ solving perturbed variational problem $\tilde{u}_h \in V_{0,h} : a_h(\tilde{u}_h, v_h) = l_h(v_h), \forall v_h \in V_{0,h}$ with modified bi-, linear forms a_h, l_h. With computers, the use of quadrature and approximation of boundaries result in such a crime. As Hiptmair likes to say, "we are all sinners".

But what are acceptable "crimes"? Crimes which do not affect the type and rate of convergence.

So how to not temper with the convergence?

if $\|u - u_h\|_1 \in \mathcal{O}(h_{\mathcal{M}}^p)$ then use quadrature rule of order at least $2p - 1$

if $V_{0,h} = \mathcal{S}_p^0(\mathcal{M})$ then use boundary fitting with polynomials of degree p

3.6 FEM: Duality Techniques for Error Estimation

Theorem 3.6.1.7. Duality estimate for linear functional output
given a functional $F : V_0 \rightarrow \mathbb{R}$ the dual solution g_F solves:

$$g_F \in V_0 : a(g_F, v) = F(v) \quad \forall v \in V_0 \tag{70}$$

and

$$|F(u) - F(u_h)| \leq \|u - u_h\|_a \cdot \inf_{v_h \in V_{0,h}} \|g_F - v_h\|_a \tag{71}$$

Why is this useful? If g_F can be approximates well in $V_{0,h}$, then the output error $|F(u) - F(u_h)|$ can converge to 0 much faster, than $\|u - u_h\|_a$

5 Non-Linear Ellipctic Boundary Value Problems

5.1 Elastic String Model

We want to derive the general variational equation for an elastic string. For this, one approximates the string as n point masses affected by gravity connected with springs, whose energy behaves according to Hooke's law. Then, one takes the limit $n \to \infty$ to derive a continuous model. The total energy is then just given by the sum of elastic and gravitational energies - given positions $(\mu_0, \dots, \mu_n)$ and $x_i = a + hi$, assuming the spring constants are 1:

Total energy of discrete spring system

$$E(\mu) = \frac{1}{2} \sum_{i=0}^{n} \left(\sqrt{h^2 + (\mu_{i+1} - \mu_i)^2} \right)^2 + \sum_{i=1}^{n} m_i \mu_i g \tag{72}$$

Then, the equilibrium position for this model can be found by minimizing this expression over μ. A continuous model is derived by replacing the discrete positions μ_i by a function $u(x_i)$, and the mass by a mass density. Then, performing some manipulations, one obtains

Total energy for the continuous string model

$$J_s(u) = \int_a^b \frac{1}{2} \frac{b-a}{L} \sigma(x) \left(\sqrt{1 + |u'(x)|^2} - \frac{L}{b-a} \right)^2 + \int_a^b g\rho(x)u(x)\mathrm{d}x \tag{73}$$

In a similiar fashion, a membrane model can be derived by assuming a two-dimensional grid of springs containing points masses and taking the limit $n \to \infty$ springs. The energy then becomes

Total energy for the membrane model

$$J_M(u) = \int_\Omega \frac{1}{2L} \sigma(x) \left(\left(\sqrt{1 + \left| \frac{\partial u}{\partial x_1}(x) \right|^2} - \frac{L}{b-a} \right)^2 + \left(\sqrt{1 + \left| \frac{\partial u}{\partial x_2} \right|^2} - \frac{L}{b-a} \right)^2 \right) + g\rho(x)u(x)\mathrm{d}x \tag{74}$$

In the limit of a taut membrane, i.e. for $L \ll b - a$, these equations just reduce to the problem of minimizing the familiar functionals seen in earlier chapters:

Elastic string taut membrane limit

$$J_s(u) = \frac{1}{2}\int_a^b \hat{\sigma}(x)|u'(x)|^2 + g\rho(x)u(x)\mathrm{d}x \tag{75}$$

Taut membrane limit

$$J_M(u) = \frac{1}{2}\int_\Omega \hat{\sigma}(x)\|\mathrm{grad}\, u(x)\|^2 + g\rho(x)u(x)\mathrm{d}x \tag{76}$$

5.2 Calculus of Variations

The difference between the equations seen so far, which hold in the limit of a very stretched string / membrane and the general equations given above, is that they are a **quadratic** minimization problem, while the ones given above are **nonlinear** minimization problems. The theory used so far mapped quadratic minimization problems to **linear** variational problems, which were then discretized. The new equations, however, yield nonlinear variational equations. Therefore, more general variational problems need to be derived.

The idea employed is that, for a minimizer of $J(u)$, every perturbation $J(u+v)$ would be larger than $J(u)$. This means that $f(t) = J(u+tv)$ has a minimum at $t = 0$ for every function v:

Theorem 5.2.1.5 Characterization of global minimizers Assume u is a global minimizer of $J(u)$

$$u = \mathrm{argmin}_{u\in V_0} J(u) \tag{77}$$

Then, if $\varphi_v(t) = J(u_* + tv)$ is differentiable in $t = 0$,

$$\frac{\mathrm{d}\varphi_v}{\varphi t}(0) = 0 \quad \forall v \in V_0 \tag{78}$$

This means that nonlinear variational equations can be derived by computing this derivative for an arbitrary v. As an example, for the elastic string model introduced in the last subchapter, this yields

Variational equations for elastic string model

$$\int_a^b \frac{\sigma(x)}{c}\left(\sqrt{1+|u_*'(x)|^2}-c\right)\frac{u_*'(x)v'(x)}{\sqrt{1+|u_*'(x)|^2}} + g\rho(x)v(x)\,\mathrm{d}x = 0 \quad \forall v \in H_0^1(]a,b[) \tag{79}$$

This can be formulated more generally as a **general variational equation**:

General variational equation A general, nonlinear variational equation reads

$$u \in \hat{V} : \quad a(u; v) = 0 \quad \forall v \in V_0 \tag{80}$$

Where a is **linear in the second argument** v and $V_0, \hat{V}$ are function spaces.

5.3 Nonlinear Boundary value problems

Similarly to the linear case, nonlinear PDEs can be derived from the variational equations by "stripping" of the derivatives of v by partial integration and employing the fundamental lemma of calculus of variations. As an example, for the string model, this yields for $u(a) = u_a$, $u(b) = u_b$

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\sigma(x)}{c}\left(\sqrt{1+|u_*'(x)|^2}-c\right)\frac{u_*'(x)}{\sqrt{1+|u_*'(x)|^2}}\right) = g\rho(x) \quad \text{in }]a,b[\tag{81}$$

5.4 Galerkin Discretization of Non-Linear BVPs

The idea of Galerkin discretization for non-linear variational equations is exactly the same as for linear equations, but they yield nonlinear systems of equations instead of linear systems of equations: One restricts u and v to a finite function space $u_h = \hat{V}_h$ and $v_h \in V_{0,h}$, and expands the functions in some basis of the space. This, then, leads to nonlinear equations for the basis expansion coefficients. These equations could be solved directly by employing some fixed-point iteration seen in NumCSE.

Galerkin discretization of variational problem Given a variational problem $a(u; v) = 0 \forall v \in V_{0,h}$, the Galerkin discretization reads

$$(F(\mu))_i = a\left(u_{0,h} + \sum_{j=1}^{N} \mu_j b_h^j; b_h^i\right), \quad i = 1, \ldots, N \tag{82}$$

Where b_h^j are fixed basis functions, μ_i are the basis function coefficients and $u_{0,h}$ contains Dirichlet boundary conditions.

Another option is to already linearize the continuous problem, and then discretize it to derive linear systems of equations. This is done by employing Newton's method in function space: The conventional Newtons iteration is given as

$$\xi^{(k+1)} = \xi^{(k)} - \mathrm{D}F(\xi^{(k)})^{-1} F(\xi^{(k)}) \tag{83}$$

Replacing the vector ξ with a function u and the derivative by a function derivative now gives

Functional Newton iteration

$$w \in V_0 \quad a(u^{(k)}; v) + \mathrm{D}_u a(u^{(k)}; v)w = 0 \quad \forall v \in V_0 \tag{84}$$

$$u^{(k+1)} = u^{(k)} + w \tag{85}$$

Here, the directional derivative is defined as

$$\mathrm{D}_u a(u^{(k)}; v)w = \lim_{t \to 0} \frac{a(u + tw; v) - a(u; v)}{t}, \quad u^{(k)} \in \hat{V}, \quad v, w \in V_0 \tag{86}$$

Now, the advantage of this equation for w is that the functional derivative is linear, i.e. $(v, w) \mapsto D_u a(u^{(k)}; v)w$ is a **bilinear form**. Now, one can employ Galerkin discretization for the linear problem in w, exactly like it was done in Chapter 2 and 3. The final equations then read

Nonlinear Newton equations for variational problems

$$w_h \in V_{0,h}^{(k)} : \mathrm{D}_u a(u_h^{(k-1)}; v_h)w_h = -a(u_h^{(k-1)}; v_h) \quad \forall v_h \in V_{0,h}^{(k)} \tag{87}$$

$$u_h^{(k)} = P_h^{(k)}(u_h^{(k-1)} + w_h) \tag{88}$$

Here, different function spaces can be used for each iterations, so a projector $P_h^{(k)}$ needs to be used to project the solution from $V_h^{(k-1)}$ to $V_h^{(k)}$. In all of this equations, the previous iterate $u_h^{(k-1)}$ is kept fixed, a linear system like derived in Chapter 2 is solved to obtain the intermediate w_h, and then a new iterate $u^{(k)}$ is obtained.

9 Second-Order Linear Evolution Problems

9.2 Parabolic Initial-Boundary Value Problems

Heat Equation
In local form the heat equation is given by

$$\frac{\partial}{\partial t}(\rho u) - \operatorname{div}(\kappa(x)\mathbf{grad}\, u) = f \quad \text{in } \tilde{\Omega} = \Omega\times]0,T[\tag{89}$$

where u is the temperature, ρ the heat capacity, κ the heat conductivity and f a (time dependent) heat source/sink. Without the time derivative, this looks very similar to standard PDE we know how to transform into a nice variational problem.

To solve it we still need boundary conditions. Besides the boundary conditions of the spatial domain, which is now required for the whole time, one also needs initial conditions over the whole domain at time 0.

$$u(x,t) = g(x,t) \quad \text{for } (x,t) \in \partial\Omega\times]0,T[\tag{90}$$
$$u(x,0) = u_0(x) \quad \text{for all } x \in \Omega \tag{91}$$

Testing with time independent test functions v and assuming ρ to be time independent as well, we get to

$$\int_\Omega \rho(x)\dot{u}v\,dx + \int_\Omega \kappa(x)\mathbf{grad}\, u \cdot \mathbf{grad}\, v\,dx = \int_\Omega f(x,t)v\,dx \quad \forall v \in H_0^1(\Omega) \tag{92}$$
$$u(x,0) = u_0(x) \in H_0^1(\Omega) \tag{93}$$

with the shorthand notation

$$m(\dot{u},v) = \int_\Omega \rho(x)\dot{u}v\,dx \tag{94}$$
$$a(u,v) = \int_\Omega \kappa(x)\mathbf{grad}\, u \cdot \mathbf{grad}\, v\,dx \tag{95}$$
$$l(v) = \int_\Omega f(x,t)v\,dx \tag{96}$$

and the realisation, that $m(\dot{u},v) = \frac{d}{dt}m(u,v)$ as only u depends on time (note, that its also important, that the domain Ω stays constant) we can rewrite 92 as

$$\frac{d}{dt}m(u,v) + a(u,v) = l(v) \tag{97}$$

which looks like something we know how to solve from NumCSE.

Stability

Lemma 9.2.3.8. Decay of solutions of parabolic evolutions
if $f \equiv 0$, the solution $u(t)$ of 92 satisfies

$$\|u(t)\|_m \leq e^{-\gamma t}\|u_0\|_m, \quad \|u(t)\|_a \leq e^{-\gamma t}\|u_0\|_a \quad \forall t \in]0, T[\tag{98}$$

where $\gamma = \text{diam}(\Omega)^{-2}$

Note that lemma also tells us, that if f is time independent, the solution $u(t)$ converges exponentially (in time) to the stationary solution (the solution of 92 without the $m(\cdot,\cdot)part$.

Method of Lines
Now lets look into how we can solve 97. Lets apply the Galerkin discretization. As u is now also time dependent, let the expansion coefficients of u also be time dependent.

$$u_h(t) = \sum_{i=1}^{N} \mu_i(t) b_h^i \tag{99}$$

Combining this with 97, we get

$$\mathbf{M}\left\{\frac{d}{dt}\vec{\mu}(t)\right\} + \mathbf{A}\vec{\mu}(t) = \vec{\varphi}(t) \tag{100}$$

$$\vec{\mu}(0) = \vec{\mu}_0 \tag{101}$$

This is now an ODE with respect to time and can be solved by time stepping, learned in NumCSE.

Recall ODE's
An ODE is given as

$$\dot{\mathbf{u}} = \mathbf{f}(t, \mathbf{u}) \tag{102}$$

and is called linear, if $\mathbf{f}(t, \mathbf{u}) = \mathbf{A}(t)\mathbf{u}$. With the ODE there is an evaluation operator associated, defined as $\Phi^{t_0,t}u_0 = u(t)$. There are some methods to approximate the evaluation operator with a discrete evaluation operator Ψ.

- explicit Euler: $\Psi^{t,t+\tau}\mathbf{u} = \mathbf{u} + \tau\mathbf{f}(t, \mathbf{u})$
- implicit Euler:$\Psi^{t,t+\tau}\mathbf{u} = \mathbf{w}, \mathbf{w} = \mathbf{u} + \tau\mathbf{f}(t + \tau, \mathbf{w})$
- implicit midpoint: $\Psi^{t,t+\tau}\mathbf{u} = \mathbf{w}, \mathbf{w} = \mathbf{u} + \tau\mathbf{f}(t + \frac{1}{2}\tau, \frac{1}{2}(\mathbf{w} + \mathbf{u}))$

Hence we can calculate the time evolution by the sequence

$$\mathbf{u}^{(0)} = \mathbf{u}_0, \quad \mathbf{u}^{(j)} = \Psi^{t_{j-1},t_j}\mathbf{u}^{(j-1)}, \quad j = 1, \ldots, M \tag{103}$$

As Ψ is the discrete approximation, the question about the error is immediate. One usually considers

- the error at final time: $\epsilon_M = \|\mathbf{u}^{(M)} - \mathbf{u}(T)\|$

- maxium error in the sequence: $\epsilon_\infty = \max_j \|\mathbf{u}^{(j)} - \mathbf{u}(t_j)\|$

Theorem 9.2.6.14. Convergence of single-step methods
given the above sequence of solutions, obtained by a single step method of order $q \in \mathbb{N}$, then

$$\epsilon_\infty = \max_j \|\mathbf{u}^{(j)} - \mathbf{u}(t_j)\| \leq C\tau^q \tag{104}$$

with $\tau = \max_j |t_j - t_{j-1}|$

Runge-Kutta Single-Step Methods

Definition 7.3.3.1. General Runge-Kutta single-step method
For coefficients $b_i, a_{i,j} \in \mathbb{R}, c_i = \sum_{j=1}^s a_{i,j}$, the discrete evolution operator $\Psi^{s,t}$ of an **s-stage Runge-Kutta single step method** (RK-SSM) for the ODE $\dot{\mathbf{u}} = \mathbf{f}(t, \mathbf{u})$ is defined by

$$\mathbf{k}_i = \mathbf{f}(t + c_i\tau, \mathbf{u} + \tau \sum_{j=1}^s a_{i,j}\mathbf{k}_j), \quad i = 1, \dots, s, \quad \Psi^{t,t+\tau}\mathbf{u} = \mathbf{u} + \tau \sum_{j=1}^s b_j\mathbf{k}_j \tag{105}$$

with $\mathbf{k}_j$ the increments.

The RK-SSM methods can be written down in compact form (the butcher scheme) as

$$\begin{array}{c|c} \mathbf{c} & \mathfrak{A} \\ \hline & \mathbf{b} \end{array} \tag{106}$$

where $\mathbf{c}$ is a vector containing the coefficients c_i, $\mathbf{b}$ the coefficients b_i and $\mathfrak{A}$ a matrix containing the coefficients $a_{i,j}$.

So continuing from 100 with different time steppings, we get

- explicit Euler:
$$\vec{\mu}^{(j)} = \vec{\mu}^{(j-1)} + \tau_j \mathbf{M}^{-1} \left(\vec{\varphi}(t_{j-1}) - \mathbf{A}\vec{\mu}^{(j-1)}\right) \tag{107}$$
- implicit Euler:
$$\vec{\mu}^{(j)} = \left(\tau_j \mathbf{A} + \mathbf{M}\right)^{-1} \left(\mathbf{M}\vec{\mu}^{(j-1)} + \tau_j \vec{\varphi}(t_{j-1})\right) \tag{108}$$
- implicit midpoint (Crank-Nicolson)
$$\vec{\mu}^{(j)} = \left(\mathbf{M} + \frac{1}{2}\mathbf{A}\right)^{-1} \tau_j \left(\left(\mathbf{M} - \frac{1}{2}\mathbf{A}\right)\vec{\mu}^{(j-1)} + \frac{1}{2}\left(\vec{\varphi}(t_j) + \vec{\varphi}(t_{j-1})\right)\right) \tag{109}$$

Which all involve solving a linear system of equations each time step. However note, that the matrices to invert, stay constant with respect to time, so we can calculate the decomposition only once to save a lot of time.

Using a general RK-SSM method as the time step, we get the following system of equations

$$\mathbf{M}\vec{\kappa}_i + \sum_{m=1}^{s} \tau a_{i,m}\mathbf{A}\vec{\kappa}_m = \vec{\varphi}(t_j + c_i\tau) - \mathbf{A}\vec{\mu}^{(j)} \tag{110}$$

$$\vec{\mu}^{(j+1)} = \vec{\mu}^{(j)} + \tau \sum_{m=1}^{s} b_m\vec{\kappa}_m \tag{111}$$

with the Kronecker product, this can be rewritten as

$$\left(\mathbf{I}_s \otimes \mathbf{M} + \tau\mathfrak{A} \otimes \mathbf{A}\right)\begin{bmatrix}\vec{\kappa}_1 \\ \vdots \\ \vec{\kappa}_s\end{bmatrix} = \begin{bmatrix}\vec{\varphi}(t_j + c_1\tau) - \mathbf{A}\vec{\mu}^{(j)} \\ \vdots \\ \vec{\varphi}(t_j + c_s\tau) - \mathbf{A}\vec{\mu}^{(j)}\end{bmatrix} \tag{112}$$

which can be used to solve for the increments $\vec{\kappa}_i$.

Remember stiff initial value problems:

Stiff IVP
An initial value problem is called stiff, if stability imposes much tighter timestep constraints on explicit single step methods than the accuracy requirements

To study the stiffness of the method of lines, we first diagonalize it. For 100 let $\vec{\psi}_1, \dots, \vec{\psi}_N$ denote the N linearly independent generalized eigenvectors satisfying

$$\mathbf{A}\vec{\psi}_i = \lambda_i\mathbf{M}\vec{\psi}_i, \quad \vec{\psi}_j^\top\mathbf{M}\vec{\psi}_i = \delta_{ij} \tag{113}$$

with positive eigenvalues λ_i. With $\mathbf{T} = \left[\vec{\psi}_1, \dots, \vec{\psi}_N\right]$ and $\mathbf{D} = \operatorname{diag}(\lambda_1, \dots, \lambda_N)$, this can be rewritten as

$$\mathbf{AT} = \mathbf{MTD}, \quad \mathbf{T}^\top\mathbf{MT} = \mathbf{I} \tag{114}$$

The eigenvectors with positive eigenvalues is guaranteed, as $\mathbf{A}, \mathbf{M}$ are (semi)positive definit. Thus with a change of basis to the eigenvector basis, one can diagonalize 100.

$$\vec{\mu}(t) = \sum_k \eta_k(t)\vec{\psi}_k \iff \vec{\mu}(t) = \mathbf{T}\vec{\eta}(t) \iff \mathbf{T}^\top\mathbf{M}\vec{\mu}(t) = \vec{\eta}(t) \tag{115}$$

$$\to \mathbf{MT}\frac{d}{dt}\vec{\eta}(t) + \mathbf{MTD}\vec{\eta}(t) = \vec{\varphi}(t) \tag{116}$$

$$\to \frac{d}{dt}\vec{\eta}(t) + \mathbf{D}\vec{\eta}(t) = \mathbf{T}^\top\vec{\varphi}(t) \tag{117}$$

As $\mathbf{D}$ is diagonal, this amount in N decoupled scalar ODE's. And on the we can perform our analysis more easily. In NumCSE you have seen, that the Euler schemes and also Crank-Nicolson can be rewritten as a RK-SSM for appropriate coefficients, so we can study the stability of the general RK-SSM for the scalar case. For $\dot{u} = -\lambda u$, with the butcher scheme 106 we obtain $\Psi_\lambda^{t,t+\tau}u = S(-\lambda\tau)u$, with the stability function

$$S(z) = 1 + z\mathbf{b}^\top(I - z\mathfrak{A})^{-1}\mathbf{1} = \frac{\det(\mathbf{I} - z\mathfrak{A} + z\mathbf{b}\mathbf{1}^\top)}{\det(\mathbf{I} - z\mathfrak{A})} \tag{118}$$

Unconditional stability of single step methods A necessary condition for unconditional stability of a single step method, is that the discrete evolution operator Ψ^t_λ applied to the scalar ODE $\dot{u} = -\lambda u$ satisfies

$$\lambda > 0 \to \lim_{j\to\infty} (\Psi^\tau_\lambda)^j u_0 = 0, \quad \forall u_0, \forall \tau > 0 \tag{119}$$

Definition 9.2.7.46. L-stability
A RK-SSM satisfying the above condition, is called L-stable if its stability function satisfies

$$|S(z)| < 1, \forall z < 0 \text{ and } S(-\infty) = \lim_{z\to-\infty} S(z) = 0 \tag{120}$$

plugging $-\infty$ int S we obtain $S(-\infty) = 1 - \mathbf{b}^\top \mathfrak{A}^{-1}\mathbf{1}$, which is equal to zero if $\mathbf{b}$ is equal to the last row of $\mathfrak{A}$.

"Meta-theorem" 9.2.8.5. Convergence of solutions of fully discrete parabolic evolution problems
Assume that

- the solution of the parabolic IBVP is "sufficiently smooth"
- its spatial Galerkin finite element discretization relies on degree p Lagrangian finite elements on uniformly shape-regular families of meshes
- timestepping is based on a L-stable single step method of order q with uniform timestep $\tau > 0$

Then we can expect an asymptotic behaviour of the total discretization error according to

$$\left(\tau \sum_{j=1}^{M} \left| u(\tau j) - u_h^{(j)} \right|^2_{H^1(\Omega)} \right)^{\frac{1}{2}} \leq C \left(h^p_{\mathcal{M}} + \tau^q \right) \tag{121}$$

Hence the total error is the spatial error plus the temporal error

9.3 Models for Vibrating Membrane

Wave Equation
In local form, the (linear) wave equation is given by

$$\rho(x)\frac{\partial^2}{\partial t^2}u - \operatorname{div}(\sigma(x)\mathbf{grad}\ u) = f \quad \text{in } \tilde{\Omega} \tag{122}$$

Note the similarity to the heat equation 89. As this makes the wave equation, a second order ODE $\ddot{\mathbf{u}} = \mathbf{f}(\mathbf{u})$, two initial conditions are needed. Additional to the initial conditions 90 & 91,

$$\frac{\partial}{\partial t}u(x,0) = v_0(x) \quad \text{for all } x \in \Omega \tag{123}$$

is also needed.

To be able to use the same time steppings as the ones introduced in the previous section, the wave function can be converted into a first order ODE:

$$\dot{u} = v \tag{124}$$

$$\rho\dot{v} = \operatorname{div}(\sigma(x)\mathbf{grad}\ u) \tag{125}$$

Remember from Analysis that the particular wave equation $\frac{\partial^2}{\partial t^2}u - c^2\frac{\partial^2}{\partial x^2}u = 0$ in 1D results in the d'Alembert solution:

$$u(x,t) = \frac{1}{2}\left(u_0(x+ct) + u_0(x-ct)\right) + \frac{1}{2c}\int_{x-ct}^{x+ct} v_0(s)ds \tag{126}$$

with u_0 and v_0 the initial conditions. Hence there is again the concept of domain of dependence and domain of influence. This will be important later. Furthermore, in the absence of a source term, as in the simple case above, the solution will stay undamped. This corresponds to *conservation of total energy*.

We can formulate the variational problem:

$$m(\ddot{u}, v) + a(u, v) = 0 \quad \forall v \in V_0 \tag{127}$$

Theorem 9.3.2.16. Energy conservation in wave propagation
If u solves eq. 127, then its energy is conserved, in the sense that

$$t \to \frac{1}{2}m(\frac{\partial}{\partial t}u, \frac{\partial}{\partial t}u) + \frac{1}{2}a(u,u) \equiv \text{const} \tag{128}$$

where $\frac{1}{2}m(\frac{\partial}{\partial t}u, \frac{\partial}{\partial t}u)$ is can be understood as the 'kinetic' energy and $\frac{1}{2}a(u,u)$ as the 'potential' energy.

Method of Lines
The method of lines gives rise to

$$\mathbf{M}\left\{\frac{d^2}{dt^2}\vec{\mu}(t)\right\} + \mathbf{A}\vec{\mu}(t) = \vec{\varphi}(t) \tag{129}$$

$$\vec{\mu}(0) = \vec{\mu}_0, \frac{d}{dt}\vec{\mu}(0) = \vec{v}_0 \tag{130}$$

Using $\vec{v} = \dot{\vec{\mu}}$, we can rewrite it to be a first order ODE.

$$\frac{d}{dt}\vec{\mu} = \vec{v} \tag{131}$$

$$\mathbf{M}\frac{d}{dt}\vec{v} = \vec{\varphi}(t) - \mathbf{A}\vec{\mu} \tag{132}$$

$$\vec{\mu}(0) = \vec{\mu}_0, \vec{v}(0) = \vec{v}_0 \tag{133}$$

Remember, in the case of $\vec{\varphi} \equiv 0$, there is conservation of energy:

$$E_h(t) = \frac{1}{2}\frac{d}{dt}\vec{\mu}^\top \mathbf{M}\frac{d}{dt}\vec{\mu} + \frac{1}{2}\vec{\mu}^\top \mathbf{A}\vec{\mu} \equiv \text{const} \tag{134}$$

So we would like, that the time stepping preserves this. Such time stepping schemes called *structure preserving*. One such timestepping scheme is the **Crank Nicolson** one:

$$\mathbf{M}\frac{\vec{\mu}^{(j+1)} - 2\vec{\mu}^{(j)} + \vec{\mu}^{(j-1)}}{\tau^2} = -\frac{1}{2}\mathbf{A}\left(\vec{\mu}^{(j-1)} + \vec{\mu}^{(j+1)}\right) + \frac{1}{2}\left(\vec{\varphi}\left(t_j - \frac{1}{2}\tau\right) + \vec{\varphi}\left(t_j + \frac{1}{2}\tau\right)\right) \tag{135}$$

Another one would be **Strömer scheme**:

$$\mathbf{M}\frac{\vec{\mu}^{(j+1)} - 2\vec{\mu}^{(j)} + \vec{\mu}^{(j-1)}}{\tau^2} = -\mathbf{A}\vec{\mu}^{(j)} + \vec{\varphi}\left(t_j\right) \tag{136}$$

For both these *second order* time stepping schemes, to get $\vec{\mu}^{(1)}$, $\vec{\mu}^{(-1)}$ is needed in the equation. Now the question is, where do we get this from? It can be obtained with a special initial step, using a symmetric (first order) difference quotient:

$$\frac{d}{dt}\vec{\mu}(0) = \vec{v}_0 \to \frac{\mu^{(1} - \mu^{(-1)}}{2\tau} = \vec{v}_0 \tag{137}$$

And finally there is the **Leapfrog** timestepping. Using the auxiliary variable $\vec{v}^{(j+1/2)} = \frac{\vec{\mu}^{(j+1)} - \vec{\mu}^{(j)}}{\tau}$ and inserting this into the Strömer scheme results in

$$\mathbf{M}\frac{\vec{v}^{(j+1)} - \vec{v}^{(j)}}{\tau} = -\mathbf{A}\vec{\mu}^{(j)} + \vec{\varphi}\left(t_j\right) \tag{138}$$

$$\frac{\vec{\mu}^{(j+1)} - \vec{\mu}^{(j)}}{\tau} = \vec{v}^{(j+1/2)} \tag{139}$$

with the inital step $\vec{v}^{(-1/2)} + \vec{v}^{(1/2)} = 2\vec{v}_0$.

10 Convection-Diffusion Problems

10.1 Heat conduction in a Fluid

Consider a flowing fluid. Then there is the key quantity, the *flow field* $\mathbf{v} : \Omega \subset \mathbb{R}^d \to \mathbb{R}^d$, where d is the dimension we consider. The flow field can be understand as $\mathbf{v}(x) =$ fluid velocity at point $x \in \Omega$.

Given a flow field $\mathbf{v}$, we can consider the autonomous initial value problem

$$\dot{\mathbf{y}} = \mathbf{v}(\mathbf{y}), \quad \mathbf{y}_0 = \mathbf{x}_0 \tag{140}$$

The solution $t \to \mathbf{y}(t)$ describes how a particle moves, carried by the fluid, also called *streamline*.

As the domain Ω is usually bounded, we cannot have fluid leaving the domain. This means the fluid velocity must be zero in the normal direction of the domain boundary. Hence

$$\mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = 0, \quad \forall \mathbf{x} \in \partial\Omega \tag{141}$$

Fourier's law in a moving fluid

$$\mathbf{j}(\mathbf{x}) = -\kappa \mathbf{grad}\, u(\mathbf{x}) + \mathbf{v}(\mathbf{x})\rho u(\mathbf{x}) \tag{142}$$

with κ the heat conductivity and ρ the volumetric heat capacity

We already know the first part, called diffusive heat flux, from the heat equation. The second part is called convective heat flux. With this new flux, the standart PDE becomes

$$-\mathrm{div}(\kappa \mathbf{grad}\, u) + \mathrm{div}(\rho \mathbf{v}(\mathbf{x}) u) = f \quad \text{in } \Omega \tag{143}$$

Incompressible Fluids
A fluid is called incompressible if its associated flow map (evaluation operator) Φ^t is volume preserving, i.e. $\left|\Phi^t(V)\right| = \left|\Phi^0(V)\right| = |V|$ for all control volumes V. This means that $\frac{\mathrm{d}}{\mathrm{d}t}\left|\Phi^t(V)\right| = 0$. Going through with some derivation (can be found in the lecture document) a equivalent statement to volume preservation is found

$$\mathrm{div}\, \mathbf{v} \equiv 0 \quad \text{in } \Omega \tag{144}$$

Hence the fluid is incompressible, if its flow velocity is divergence free. In the derivation an important theorem is used

Theorem 10.1.3.7. Differentiation formula for determinants
Let $\mathbf{S}$ be a smooth matrix-valued function. If $\mathbf{S}$ is regular then

$$\frac{\mathrm{d}}{\mathrm{d}t}\det(\mathbf{S}(t)) = \det(\mathbf{S}(t))\mathrm{tr}\Big(\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t}\mathbf{S}^{-1}(t)\Big) \tag{145}$$

with det the determinant and tr the trace of a matrix.

In case of incompressibility, equation 143 can be simplified, using the general product rule 12 and div $\mathbf{v} = 0$

$$-\text{div}(\kappa \mathbf{grad}\, u) + \mathbf{v}(\mathbf{x}) \cdot \mathbf{grad} \rho u = f \quad \text{in } \Omega \tag{146}$$

Of course we can also look at the time dependent heat flow, similar to the heat equation 92

$$\frac{\partial}{\partial t}(\rho u) - \text{div}(\kappa \mathbf{grad}\, u) + \text{div}(\rho \mathbf{v}(\mathbf{x}, \mathbf{t}) u) = f \quad \text{in } \Omega \tag{147}$$

We will see lateron how this can be solved.

10.2 Stationary Convection-Diffusion Problems

Here we will focus on the convection diffusion equation 143 with constant κ, ρ, incompressible flow $\mathbf{v}$ and zero dirichlet boundary conditions. Hence

$$-\kappa \Delta u + \rho \mathbf{v}(\mathbf{x}) \cdot \mathbf{grad} u = f \quad \text{in } \Omega, \qquad u = 0 \quad \text{on } \partial\Omega \tag{148}$$

Nondimentionalizing the problem results in

$$-\epsilon \Delta u + \mathbf{v}(\mathbf{x}) \cdot \mathbf{grad} u = f \quad \text{in } \Omega, \qquad u = 0 \quad \text{on } \partial\Omega \tag{149}$$

with $\|\mathbf{v}\|_{L^\infty(\Omega)} = 1$. This results in the following variational form

$$\epsilon \int_\Omega \mathbf{grad}\, u \cdot \mathbf{grad}\, v dx + \int_\Omega (\mathbf{v} \cdot \mathbf{grad}\, u) v dx = \int_\Omega f(x) v dx \tag{150}$$

with the lefthandside the bilinear form $a(u, v)$. However, a is not symmetric. This also means, it does not induce an energy norm. However it is still positive definite (see lecture document).

Singular perturbation
A boundary value problem depending on a parameter ϵ is called singularly perturbed, if the limit problem for $\epsilon \to \epsilon_0$ is not compatible with the boundary conditions.

For $\epsilon = 0$ the above PDE is singular perturbed. It cannot satisfy dirichlet boundary conditions on the outflow part of the boundary. $\Gamma_{\text{out}} = \{\mathbf{x} \in \partial\Omega : \mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) > 0\}$, similarly $\Gamma_{\text{in}} = \{\mathbf{x} \in \partial\Omega : \mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) < 0\}$

Upwinding
When trying to solve eq. 151 with the Galekrin approach, when ϵ is very close to 0, one can observe huge oscillations in the solution, which is not correct. It comes from the fact, that the Galerkin matrix becomes close to singular. So our goal is to get a robust method, which can solve eq. 151 no matter the ϵ.

Consider again eq. 151 but in $d = 1$ and with zero boundary conditions

$$\epsilon \int_0^1 \frac{\partial u}{\partial x} \frac{\partial v}{\partial x} dx + \int_0^1 \frac{\partial u}{\partial x} v dx = \int_0^1 f(x) v dx \tag{151}$$

To calculate the Galerkin matrix for an equidistant mesh with M cells, we use the global composite trapezoidal rule for the convective term

$$\int_0^1 \psi(x) dx = h \sum_{j=0}^{M} \psi(jh) \tag{152}$$

Hence the convective term of the bilinear form will be approximated by

$$\int_0^1 \frac{\partial u_h}{\partial x} v_h dx \approx h \sum_{j=1}^{M-1} \frac{\partial u_h}{\partial x}(jh) v_h(jh) \tag{153}$$

But $\frac{\partial u_h}{\partial x}(jh)$ is not valid, as its discontinuous at the nodes for $u_h \in S^0_{1,0}$. However, convection transports the information in the direction of $\mathbf{v}$ (= 1 in our case). Hence use

$$\frac{\partial u_h}{\partial x}(jh) = \lim_{\delta \to 0} \frac{\partial u_h}{\partial x}(jh - \delta \mathbf{v}) = \frac{\partial u_h}{\partial x}_{|]x_{j-1}, x_j[} \tag{154}$$

And generalized in more dimensions

$$\mathbf{v}(\mathbf{p}) \cdot \mathbf{grad}\, u_h(\mathbf{p}) = \lim_{\delta \to 0} \mathbf{v}(\mathbf{p}) \cdot \mathbf{grad}\, u_h(\mathbf{p} - \delta \mathbf{v}(\mathbf{p})) \tag{155}$$

Streamline diffusion

A totally different idea to fix the problem of $\epsilon \to 0$ is to add some h dependent diffusion. I.e. replace $\epsilon \leftarrow \epsilon + c(h)$ with $c(h) > 0$. However, there is smearing in the internal layers. But as the solution is smooth along the direction of $\mathbf{v}$, so adding diffusion along the velocity should not do any harm.

The method of *Anisotropic diffusion* is born. On cell K replace $\epsilon \leftarrow \epsilon \mathbf{I} + \delta_K \mathbf{v}_K \mathbf{v}_K^\top$ with $\mathbf{v}_K$ the local velocity, i.e. obtained by averaging and $\delta_K > 0$ some controlling parameter. Resuling in

$$\int_\Omega \left(\epsilon \mathbf{I} + \delta_K \mathbf{v}_K \mathbf{v}_K^\top\right) \mathbf{grad}\, u \cdot \mathbf{grad}\, v dx + \int_\Omega (\mathbf{v} \cdot \mathbf{grad}\, u) v dx = \int_\Omega f(x) v dx \tag{156}$$

However this affects the solution u, such that it will not be the same as the one from eq. 151. To get rid of this inconsistency, the anisotropic diffusion can be introduced via a residual term

$$\int_\Omega \epsilon \mathbf{grad}\, u \cdot \mathbf{grad}\, v dx + \int_\Omega (\mathbf{v} \cdot \mathbf{grad}\, u) v dx + \sum_{K \in \mathcal{M}} \delta_K \int_K (-\epsilon \Delta + \mathbf{v} \cdot \mathbf{grad}\, u - f) \mathbf{v} \cdot \mathbf{grad}\, v = \int_\Omega f(x) v dx \tag{157}$$

the added term will be zero for the exact solution (strong PDE) and the anisotropic diffusion is still here. The control parameter is usually chosen according to

$$\delta_K = \begin{cases} \epsilon^{-1} h_K^2 & \text{if } \frac{\|\mathbf{v}\|_{K,\infty} h_K}{2\epsilon} \leq 1 \\ h & \text{if } \frac{\|\mathbf{v}\|_{K,\infty} h_K}{2\epsilon} > 1 \end{cases} \tag{158}$$

With this, the $\mathcal{O}(h_{\mathcal{M}}^2)$ convergence of $\|u - u_h\|_{L^2(\Omega)}$ for h refinement is preserved, while upwind quadrature only achieves $\mathcal{O}(h_{\mathcal{M}})$ convergence.

10.3 Discretization of Time-Dependent (Transient) ConvectionDiffusion IBVPs

Now we will take a look at how time dependent convection diffusion can be modeled. Assuming the incompressibility condition and nondimensionalizing, eq. 147 becomes

$$\frac{\partial}{\partial t} u - \epsilon \Delta u + \mathbf{v}(\mathbf{x},t) \cdot \mathbf{grad}\, u = f \quad \text{in } \Omega \tag{159}$$

Up on inspecting the solution obtained with method of lines, one observes that without upwind quadrature, oscillations occur. However, with upwind damping is observed, which is wrong. Hence other methods of solving have to be explored. Of course the limit of $\epsilon \to 0$ again poses a problem. So lets first look at the pure transport problem

$$\frac{\partial}{\partial t} u + \mathbf{v}(\mathbf{x},t) \cdot \mathbf{grad}\, u = f \quad \text{in } \Omega \tag{160}$$

Its solution is given by the *Method of Characteristics*

$$u(\mathbf{x},t) = \begin{cases} u_0(\mathbf{x_0}) + \int_0^t f(\mathbf{y}(s),s)ds & \text{if } \mathbf{y}(s) \in \Omega \forall 0 < s < t \\ g(\mathbf{y}(s_0),s_0) + \int_{s_0}^t f(\mathbf{y}(s),s)ds & \text{if } \mathbf{y}(s_0) \in \partial\Omega, \mathbf{y}(s) \in \Omega \forall s_0 < s < t \end{cases} \tag{161}$$

where

$$\dot{\mathbf{y}}(t) = \mathbf{v}(\mathbf{y}(t),t) \tag{162}$$

u_0 the initial condition and g the dirichlet boundary conditions on the inflow boundary. Unfortunately, this only work for the pure transport problem. For $\epsilon > 0$ we need an other method

Splitting Methods
given a general ODE whose right hand side is the sum of two functions

$$\dot{\mathbf{y}} = \mathbf{g}(t,\mathbf{y}) + \mathbf{r}(t,\mathbf{y}) \tag{163}$$

The Strang splitting single step method provides a method to solve it

Strang splitting
compute $\mathbf{y}^{(j+1)}$ given $\mathbf{y}^{(j)}$ according to

$$\tilde{\mathbf{y}} = \mathbf{z}(t_j + \frac{1}{2}\tau), \qquad \text{where } \mathbf{z}(t) \text{ solves } \dot{\mathbf{z}} = \mathbf{g}(t,\mathbf{z}), \mathbf{z}(t_j) = \mathbf{y}^{(j)} \tag{164}$$

$$\hat{\mathbf{y}} = \mathbf{w}(t_{j+1}), \qquad \text{where } \mathbf{w}(t) \text{ solves } \dot{\mathbf{z}} = \mathbf{r}(t,\mathbf{w}), \mathbf{w}(t_j) = \tilde{\mathbf{y}} \tag{165}$$

$$\mathbf{y}^{(j+1)} = \mathbf{z}(t_{j+1}), \qquad \text{where } \mathbf{z}(t) \text{ solves } \dot{\mathbf{z}} = \mathbf{g}(t,\mathbf{z}), \mathbf{z}(t_j + \frac{1}{2}) = \hat{\mathbf{y}} \tag{166}$$

and $t_{j+1} = t_j + \tau$

Theorem 10.3.3.5. Order of Strang splitting single step method
Assuming exact or second order accuracy solution of the initial value problems of the sub-steps, the Strang splitting single step method is of second order

We can now apply this to eq. 159

$$\begin{array}{ccccc} \frac{\partial}{\partial t} u & = & \epsilon \Delta u & f - \mathbf{v} \cdot \mathbf{grad}\, u \\ \updownarrow & & \updownarrow & \updownarrow \\ \dot{\mathbf{y}} & = & \mathbf{g}(\mathbf{y}) & \mathbf{r}(\mathbf{y}) \end{array} \tag{167}$$

This amount to once solving pure diffusion

$$\frac{\partial}{\partial t} z - \epsilon \Delta z = 0 \tag{168}$$

and once pure transport

$$\frac{\partial}{\partial t} w + \mathbf{v} \cdot \mathbf{grad}\, u = f \tag{169}$$

To solve the pure transport problem, we have seen the method of characteristics eq. 161. However, it requires integration along streamlines. One idea is to solve it with the particle method.

1. Pick suitable interpolation nodes $\{\mathbf{p}_i\}$, the initial particle positions
2. Solve initial value problems
$$\dot{\mathbf{y}}(t) = \mathbf{v}(\mathbf{y},t) \quad , \quad \mathbf{y}(0) = \mathbf{p}_i \tag{170}$$
with suitable sigle step methods
3. Reconstruct the approximation. With the composite midpoint rule
$$u_h^{(j)}\left(\mathbf{p}_i^{(j)}\right) = u_0(\mathbf{p}_i) + \tau \sum_{l=1}^{j-1} f\left(\frac{1}{2}\left(\mathbf{p}_i^l + \mathbf{p}_i^{l-1}\right), \frac{1}{2}\left(t_l + t_{l-1}\right)\right) \tag{171}$$

But the interpolation nodes change over time and care needs to be taken, to add particles each step at the inflow boundary and remove one, which leave the domain. Because of the movement of the nodes and potential creation and deletion, each step we need to re-mesh, create a new triangular mesh with the advected nodes/particles.

Semi Lagrangian
An other method, which relies on a fixed mesh is the semi Lagrangian method.

Definition 10.3.4.2. Material derivative
given a velocity field $\mathbf{v}$, the material derivative of a function f is given by

$$\frac{Df}{D\mathbf{v}}(\mathbf{x}, t_0) = \lim_{\tau \to 0} \frac{f(\mathbf{x}, t_0) - f(\Phi^{t_0, -\tau}\mathbf{x}, t_0 - \tau)}{\tau} \tag{172}$$

By the chainrule we find

$$\frac{Df}{D\mathbf{v}}(\mathbf{x}, t) = \mathbf{grad}\, f(\mathbf{x}, t) \cdot \mathbf{v}(\mathbf{x}, t) + \frac{\partial}{\partial t} f(\mathbf{x}, t) \tag{173}$$

Hence the transient convection diffusion eq. 159 can be rewritten as

$$\frac{Du}{D\mathbf{v}} - \epsilon \Delta u = f \quad \text{in } \Omega \tag{174}$$

By using a backwards difference of the material derivative, we get a semi-discretization

$$\frac{u^{(j)}(\mathbf{x}) - u^{(j-1)}(\Phi^{t_j,t_j-\tau}\mathbf{x})}{\tau} - \epsilon\Delta u^{(j)} = f(\mathbf{x}, t_j) \quad \text{in } \Omega \tag{175}$$

with additional initialconditions for $t = t_j$. On this semi-discretization the standard Galerkin method can be applied.

$$\int_\Omega \frac{u^{(j)}(\mathbf{x}) - u^{(j-1)}(\Phi^{t_j,t_j-\tau}\mathbf{x})}{\tau} v d\mathbf{x} + \epsilon \int_\Omega \mathbf{grad}\, u^{(j)} \cdot \mathbf{grad}\, v d\mathbf{x} = \int_\Omega f(\mathbf{x}, t_j) v d\mathbf{x} \tag{176}$$

Unfortunately this cannot be implemented as is, because the function $u^{(j-1)}(\Phi^{t_j,t_j-\tau}\mathbf{x})$ is not smooth in $\mathcal{M}$ and is hence not a finite element function on $\mathcal{M}$. To get around this, simply replace it by its linear interpolant $I_1(u^{(j-1)} \circ \Phi^{t_j,t_j-\tau})$ and replace $\Phi^{t_j,t_j-\tau}\mathbf{x}$ by $\mathbf{x} - \tau\mathbf{v}(\mathbf{x},t_j)$ (explicit Euler).

$$\int_\Omega \frac{u^{(j)}(\mathbf{x}) - I_i(u^{(j-1)}(\cdot - \tau\mathbf{v}(\cdot,t_j)))(\mathbf{x})}{\tau} v d\mathbf{x} + \epsilon \int_\Omega \mathbf{grad}\, u^{(j)} \cdot \mathbf{grad}\, v d\mathbf{x} = \int_\Omega f(\mathbf{x}, t_j) v d\mathbf{x} \tag{177}$$

Which can be implemented now.

11 Numerical Methods for Conservation-Laws

11.2 Scalar Conservation Laws in 1D

The goal of this chapter is solve Cauchy problems which are of the form

$$\frac{\partial u}{\partial t}(x,t) + \frac{\partial}{\partial x}(f(u(x,t),x)) = s(u(x,t),x,t). \tag{178}$$

The flux $f : \mathbb{R} \times \Omega \to \mathbb{R}$ can be a general function, which can depend non-linearly on the solution u. Everything in this chapter will be one dimensional in space and time. So we have $\Omega \subseteq \mathbb{R}$.

Particle Model

One first example is the particle model, in the lecture we took cars as particles and wanted to model traffic speed as a function of the number of cars in the following way:

$$\dot{x}_i(t) = v_{max}(1 - \frac{\Delta_0}{\Delta x_i(t)}), \quad \Delta x_i(t) = x_{i+1}(t) - x_i(t). \tag{179}$$

Intuitively, this describes the speed of car i as a function of the maximum velocity the car can drive v_{max}, the distance to the car in front $\Delta x_i(t)$ and the minimal distance, i.e. car length Δ_0.

To get a differential equation we have to think about what quantities must preserved and how we can relate the changes of cars over some space interval to the speed of the cars in this interval. This can then be modelled as

$$\frac{\partial u}{\partial t}(x,t) = \frac{\partial}{\partial x} u(1-u), \tag{180}$$

where u describes the care density (average number of cars on the road in some infinitesimally small interval at position x and time t).

11.2.2 Characteristics

We consider the conservation law as above 178 with $s = 0$. Then a characteristic curve is defined as

Definition 11.2.2.3 Characteristic curve for one dimensional scalar conservation law
$\Gamma : [0, T] \to \mathbb{R} \times [0, T]$ with $\Gamma(\tau) := (\gamma(\tau), \tau)$, such that γ satisfies

$$\frac{d}{dt}\gamma(\tau) = f'(u(\gamma(\tau), \tau)) \tag{181}$$

for $0 \leq \tau \leq T$.

Generally, characteristic curves are lines along which information propagates. This means a $u(x, t)$ will only depend on the initial condition at x_0, $u_0(x_0)$ if there is a characteristic curve that starts in the point x_0 are travels to the point in space time (x, t). One property of Characteristic curves is the following:

Lemma 11.2.2.6 Classical solution and characteristic curves
Smooth solutions of 178 with $s \equiv 0$ are constant along characteristic curves.

For example in the case of linear advection (this is the ODE $\partial_t u + v\partial_x u = 0$) we can use this to solve the equation because $\gamma(\tau) = (x_0 + \tau v)$, which implies $u(x, t) = u_0(x - tv)$.

But this doesn't work if the solution is not smooth for example in the traffic flow model above, the solution has a jump after a certain time and hence this approach breaks down.

11.2.4 Jump conditions and Riemann Problem

The method of characteristics usually only works up to a certain point in time. To get the solution for times after that, we first note that the solution will usually have a discontinuity after the time where the method of characteristics breaks down. So we study how the solution behaves at these jumps (discontinuities).

The setting is as follows: We study the equation (178) still with $s \equiv 0$. Then we can derive that along jumps, the normal components must be continuous, which leads to the

Definition 11.2.4.2 Rankine Hugoniot (jump) condition

$$\dot{s}(u_l - u_r) = f(u_l) - f(u_r) \tag{182}$$

where $\dot{s} = \frac{d\gamma}{dt}$ is the time derivative of the discontinuity curve $\Gamma(t) = (\gamma(t), t) \in \mathbb{R} \times [0, T]$. And u_l is the solution value on the left of the jump and u_r is the solution value on the right of the jump.

Note that this is useful because it allows us to compute the jump if we know u_l and u_r.

The Riemann problem is given as

Definition 11.2.5.1 Riemann problem

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \tag{183}$$

and

$$u_0(x) = \begin{cases} u_l \in \mathbb{R} & \text{if } x < 0 \\ u_r \in \mathbb{R} & \text{if } x > 0 \end{cases} \tag{184}$$

Note that f can still be chosen to be any sufficiently smooth flux function.

Using the Rankine Hugoniot jump condition we then get the following solution for Riemann problems with a shock:

Lemma 11.2.5.4 Shock solution for Riemann problem
For any two states $u_r, u_l \in \mathbb{R}$ the piecewise constant function

$$u(x,t) := \begin{cases} u_l & \text{for } x < \dot{s}t \\ u_r & \text{for } x > \dot{s}t, \end{cases} \qquad \dot{s} := \frac{f(u_l) - f(u_r)}{u_l - u_r}, \qquad x \in \mathbb{R}, 0 < t < T \tag{185}$$

is a weak solution to the Riemann problem.

Note that the solution only holds if the equation implies a shock (jump). This for example the case if f is convex and $u_l > u_r$. Or if f is concave and $u_r > u_l$.

If the jump only exists in the beginning we have a different solution

Lemma 11.2.5.10 Rarefaction solution of Riemann problem

If $f \in C^2(\mathbb{R})$ is strictly $\begin{cases} \text{convex and } u_l < u_r \\ \text{concave and } u_r < u_l, \end{cases}$ then

$$u(x,t) := \begin{cases} u_l & \text{for } x < \min\{f'(u_l), f'(u_r)\} \cdot t \\ g(\frac{x}{t}) & \text{for } \min\{f'(u_l), f'(u_r)\} < \frac{x}{t} < \max\{f'(u_l), f'(u_r)\} \\ u_l & \text{for } x > \max\{f'(u_l), f'(u_r)\} \cdot t \end{cases} \tag{186}$$

$g := (f')^{-1}$, is a weak solution to the Riemann problem.

The question when to choose which of the two solution is answered by

Definition 11.2.6.1 Lax entropy condition

$u \hat{=}$ weak solution of the (178) piecewise classical solution in neigborhood of C^2-curve $\Gamma := (\gamma(\tau), \tau), 0 \leq \tau \leq T$, discontinuous across Γ.

u satisfies the *Lax entropy condition* in $(x_0, t_0) \in \Gamma \iff f'(u_l) > \frac{f(u_l)-f(u_r)}{u_l-u_r} > f'(u_r)$.

Now we have that if u satisfies the Lax entropy condition, then we have to pick the shock solution. Otherwise we pick the rarefaction solution.

11.2.7 Properties of Entropy Solutions

The essential properties here are that with the propagation speed $f'(u)$ we find the domain of dependence and the domain of influence, which is best illustrated by a picture and hence we encourage the reader to look at the lecture document and the illustrations below Theorem 11.2.7.3.

Moreover the second result is that the number of extremas of the solution is non-increasing in time.

11.3 Conservative Finite Volume (FV) Discretization

11.3.1 Finite-Difference Methods

Finite difference methods are probably the simplest methods for solving PDEs. We just replace the spacial derivatives by some finite difference quotient for example one of the following

Symmetric difference quotient

$$\frac{\partial f}{\partial x} \approx \frac{f(x_0 + h) - f(x_0 - h)}{2h} \tag{187}$$

Backward difference quotient

$$\frac{\partial f}{\partial x} \approx \frac{f(x_0) - f(x_0 - h)}{h} \tag{188}$$

Forward difference quotient

$$\frac{\partial f}{\partial x} \approx \frac{f(x_0 + h) - f(x_0)}{h} \tag{189}$$

Then we construct a solution by time stepping in the sense, that given $u(x, t_k)$ we compute $u(x, t_{k+1})$ by using some Runge Kutta integrator.

With the example of finite difference methods we observe that by the nature of the problems we are studying in this chapter the solutions have to be constructed under consideration of the flux direction. That is, in the spirit of characteristic curves we know that information propagates along curves in space time. And for example if this curve advances from left to right in space, then we need to use the forward difference quotient, because the backward difference quotient will not contain the information of the flow direction.

11.3.2 Spatially Semi-Discrete Conservation Form

The method we will use in this section of the course is the finite Volume Method, This means we build a mesh by taking intervals around the spacial points in which we approximate the solution u. And then we use the problem definition

$$\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x} f(u) \tag{190}$$

to derive in the simplest case

$$\frac{\partial u(x_i, t_k)}{\partial t} \approx \frac{\partial}{\partial t} \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t_k) dx \approx -\frac{1}{h} \big(F(u(x_j, t_k), u(x_{j+1}, t_k)) - F(u(x_{j-1}, t_k), u(x_j, t_k)) \big), \tag{191}$$

where $F(...)$ is a approximation of the flux function $f(u)$. Note that in the above example F only depends on two neighbouring nodes, but this can also be extended to use more nodes. As we can then plug the above equation given F in a Runge Kutta solver again, we reduced the problem to choosing appropriate Flux functions.

Definition 11.3.3.5 Consistency numerical flux function
A numerical function $F : \mathbb{R}^{m_l + m_r} \to \mathbb{R}$ is *consistent* with the flux $f : \mathbb{R} \to \mathbb{R}$ if

$$F(u, u, \dots, u) = f(u), \qquad \forall u \in \mathbb{R} \tag{192}$$

Then we have the result that consistent numerical flux functions will reconstruct the correct "discrete shock speed" when applied as explained above. As this is such an important property the finite volume method comes in very handy to solve these Cauchy problems.

11.3.4 Numerical Flux Functions

This section now treats how to find suitable flux functions. There are several options introduced starting with the simplest which basically corresponds to a average of the two inputs in $F(u, w)$. But then it turns out that this flux suffers from similar problems as the finite difference methods did.

One remedy for this is the *Lax-Friedrichs / Rusanov Flux* Flux which is useful but is flattens the edges of jumps. Which is due to it's constuction with additional diffusion.

As pointed out before, the direction in which the information flows is crucial, so an important idea to choose the right flux is to respect that. Moreover the flux has to reproduce physical solution in the sense as explained above when studying two possible solutions for the Riemann problem.

The final flux introduced, which solves these problems is the Godunov Flux

11.3.4.33 Godunov Flux

$$F_{GD}(v, w) = \begin{cases} \min_{v \le u \le w} f(u) & \text{, if } v < w \\ \max_{w \le u \le v} f(u) & \text{, if } v \ge w, \end{cases} \tag{193}$$

11.3.5 Monotone Schemes

In one of the above subsections it was mentioned that the number of extremas must not increase over time. This section shows that the two useful fluxes we derived in the previous chapter both have these property. This is established by

11.3.5.8 Monotonicity of Lax-Friedrichs / Rusanov numerical flux and Godunov flux
For any continuously differentiable flux function f the associated Lax-Friedrichs/Rusanov flux and Godunov flux are monotone.

and

11.3.5.13 Non-oscillatory monotone semi-discrete evolutions
If $\mu = \mu(t)$ solves the two point flux equation (191), with a monotone numerical flux and $\mu(0)$ has finitely many local extrema, then the number of local extrema of $\mu(t)$ cannot be larger than that of $\mu(0)$.

11.4 Timestepping for Finite-Volume Methods

As already introduced earlier, to solve the the equations, once we chose the numerical Flux, we use Runge Kutta numerical Integration.

This subsection then studies the some conditions that have to be considered, when applying these Runge Kutta methods. In particular, what constraints we have, when choosing the timestepsize τ.

11.4.2.5 Numerical domain of dependence
Consider explicit transition-invariant fully discrete evolution $\mu^{(k+1)} := \mathcal{H}(\mu^{(k)})$ on uniform spatio-temporal mesh $(x_j = hj, j \in \mathbb{Z}, t_k = k\tau, k \in \mathbb{N}_0)$ with

$$\exists m \in \mathbb{N}_0 : (\mathcal{H}(\mu))_j = \mathcal{H}(\mu_{j-m}, \dots, \mu_{j+m}), j \in \mathbb{Z}. \tag{194}$$

Then the *numerical domain of dependence* is given by

$$D_h^-(x_j, t_k) := \{(x_n, t_l) \in \mathbb{R} \times [0, t_k] : j - m(k-l) \leq n \leq j + m(k-l)\} \tag{195}$$

Note that the definition applied on the current problem with flux function $F(u_{j-m}, \dots, u_{j+m})$, we point out that $\mathcal{H}$ is the symbol for all the operations done in one timestep of Runge Kutta, and m corresponds to the number of neighbours we need to compute the numerical flux in one point.

The following kind of condition appears over and over again in numerical integration and gives a upper bound for the timestepsize τ that can be used to construct the numerical soution, such that the solution is stable.

11.4.2.11 Courant-Friedrichs-Lewy (CFL-) condition
An explicit translation-invariant local fully discrete evolution $\mu^(t+1) := \mathcal{H}(\mu)$ on uniform spacio-temporal mesh $(x_j = hj, j \in \mathbb{Z}, t_k = k\tau, k \in \mathbb{N})$ satisfies the CFL condition, if the convex hull of its numerical domain of dependence contains the maximal analytical domain of dependence

$$D^-(x_j, t_k) \subset convex(D_h^-(x_j, t_k)) \quad \forall j, k. \tag{196}$$

Applied on the problem we are studying right now this means

$$\frac{\tau}{h} \leq \frac{m}{\max\{|\dot{s}_{\min}|, |\dot{s}_{\max}|\}}. \tag{197}$$

11.4.4 Convergence of Fully Discrete FV Method

We essentially get at most order one convergence in the maximum and the L^1 norm. This can be seen by the following lemma

Order barrier for monotone numerical fluxes
Monotone numerical fluxes are at most first order consistent.

Alternative ways to see this is to do numerical experiments or Taylor expansion of for example the Godunov flux.