Study Guide: Solving differential equations with finite elements

Hans Petter Langtangen[1,2]

[1]Center for Biomedical Computing, Simula Research Laboratory
[2]Department of Informatics, University of Oslo

Nov 4, 2013

Contents

1 Differential equation models

Our aim is to extend the ideas for approximating f by u, or solving

$$u = f$$

to real differential equations like[[[

$$-u'' + bu = f, \quad u(0) = 1,\ u'(L) = D$$

Three methods are addressed:

1. least squares
2. Galerkin/projection
3. collocation (interpolation)

Method 2 will be totally dominating!

1.1 Abstract differential equation

$$\mathcal{L}(u) = 0, \quad x \in \Omega \tag{1}$$

Examples (1D problems):

$$\mathcal{L}(u) = \frac{d^2u}{dx^2} - f(x), \tag{2}$$

$$\mathcal{L}(u) = \frac{d}{dx}\left(\alpha(x)\frac{du}{dx}\right) + f(x), \tag{3}$$

$$\mathcal{L}(u) = \frac{d}{dx}\left(\alpha(u)\frac{du}{dx}\right) - au + f(x), \tag{4}$$

$$\mathcal{L}(u) = \frac{d}{dx}\left(\alpha(u)\frac{du}{dx}\right) + f(u,x) \tag{5}$$

1.2 Abstract boundary conditions

$$\mathcal{B}_0(u) = 0,\ x = 0, \quad \mathcal{B}_1(u) = 0,\ x = L \tag{6}$$

Examples:

$$\mathcal{B}_i(u) = u - g, \qquad \text{Dirichlet condition} \tag{7}$$

$$\mathcal{B}_i(u) = -\alpha\frac{du}{dx} - g, \qquad \text{Neumann condition} \tag{8}$$

$$\mathcal{B}_i(u) = -\alpha\frac{du}{dx} - h(u - g), \qquad \text{Robin condition} \tag{9}$$

1.3 Reminder about notation

- $u_{\mathrm{e}}(x)$ is the symbol for the *exact* solution of $\mathcal{L}(u_{\mathrm{e}}) = 0$
- $u(x)$ denotes an *approximate* solution
- We seek $u \in V$
- $V = \mathrm{span}\{\psi_0(x), \ldots, \psi_N(x)\}$, V has basis $\{\psi_i\}_{i\in\mathcal{I}_V}$
- $\mathcal{I}_V = \{0, \ldots, N\}$ is an index set
- $u(x) = \sum_{j\in\mathcal{I}_V} c_j\psi_j(x)$
- Inner product: $(u, v) = \int_\Omega uv\,\mathrm{d}x$
- Norm: $||u|| = \sqrt{(u,u)}$

1.4 New topics

Much is similar to approximating a function (solving $u = f$), but two new topics are needed:

- Variational formulation of the differential equation problem (including integration by parts)
- Handling of boundary conditions

1.5 Residual-minimizing principles

- When solving $u = f$ we knew the error $e = f - u$ and could use principles for minimizing the error
- When solving $\mathcal{L}(u_{\rm e}) = 0$ we do not know $u_{\rm e}$ and cannot work with the error $e = u_{\rm e} - u$
- We only have the *error in the equation*: the residual R

Inserting $u = \sum_j c_j\psi_j$ in $\mathcal{L} = 0$ gives a residual

$$R = \mathcal{L}(u) = \mathcal{L}(\sum_j c_j\psi_j) \neq 0 \tag{10}$$

Goal: minimize R wrt $\{c_i\}_{i\in\mathcal{I}_V}$ (and hope it makes a small e too)

$$R = R(c_0, \ldots, c_N; x)$$

1.6 The least squares method

Idea: minimize

$$E = ||R||^2 = (R, R) = \int_\Omega R^2 dx \tag{11}$$

Minimization wrt $\{c_i\}_{i\in\mathcal{I}_V}$ implies

$$\frac{\partial E}{\partial c_i} = \int_\Omega 2R\frac{\partial R}{\partial c_i}dx = 0 \quad \Leftrightarrow \quad (R, \frac{\partial R}{\partial c_i}) = 0, \quad i \in \mathcal{I}_V \tag{12}$$

$N + 1$ equations for $N + 1$ unknowns $\{c_i\}_{i\in\mathcal{I}_V}$

1.7 The Galerkin method

Idea: make R orthogonal to V,

$$(R, v) = 0, \quad \forall v \in V \tag{13}$$

This implies

$$(R, \psi_i) = 0, \quad i \in \mathcal{I}_V \tag{14}$$

$N + 1$ equations for $N + 1$ unknowns $\{c_i\}_{i\in\mathcal{I}_V}$

1.8 The Method of Weighted Residuals

Generalization of the Galerkin method: demand R orthogonal to some space W, possibly $W \neq V$:

$$(R, v) = 0, \quad \forall v \in W \tag{15}$$

If $\{w_0, \ldots, w_N\}$ is a basis for W:

$$(R, w_i) = 0, \quad i \in \mathcal{I}_V \tag{16}$$

- $N + 1$ equations for $N + 1$ unknowns $\{c_i\}_{i\in\mathcal{I}_V}$
- Weighted residual with $w_i = \partial R/\partial c_i$ gives least squares

1.9 Terminology: test and trial Functions

- ψ_j used in $\sum_j c_j\psi_j$ is called *trial function*
- ψ_i or w_i used as weight in Galerkin's method is called *test function*

1.10 The collocation method

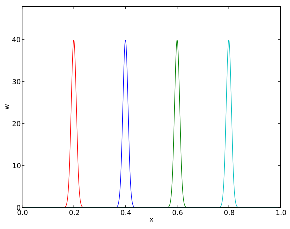
Idea: demand $R = 0$ at $N + 1$ points

$$R(x_i; c_0, \ldots, c_N) = 0, \quad i \in \mathcal{I}_V \tag{17}$$

Note: The collocation method is a weighted residual method with delta functions as weights

$$0 = \int_\Omega R(x; c_0, \ldots, c_N)\delta(x - x_i)\,\mathrm{d}x = R(x_i; c_0, \ldots, c_N)$$

$$\text{property of } \delta(x): \quad \int_\Omega f(x)\delta(x - x_i)dx = f(x_i), \quad x_i \in \Omega \tag{18}$$



2 Examples on using the principles

Goal.
Exemplify the least squares, Galerkin, and collocation methods in a simple 1D problem with global basis functions.

2.1 The first model problem

$$-u''(x) = f(x), \quad x \in \Omega = [0, L], \quad u(0) = 0,\ u(L) = 0 \tag{19}$$

Basis functions:

$$\psi_i(x) = \sin\left((i+1)\pi\frac{x}{L}\right), \quad i \in \mathcal{I}_V \tag{20}$$

The residual:

$$\begin{aligned} R(x; c_0, \ldots, c_N) &= u''(x) + f(x), \\ &= \frac{d^2}{dx^2}\left(\sum_{j\in\mathcal{I}_V} c_j\psi_j(x)\right) + f(x), \\ &= -\sum_{j\in\mathcal{I}_V} c_j\psi_j''(x) + f(x) \end{aligned} \tag{21}$$

2.2 Boundary conditions

Since $u(0) = u(L) = 0$ we must ensure that all $\psi_i(0) = \psi_i(L) = 0$. Then

$$u(0) = \sum_j c_j\psi_j(0) = 0, \quad u(L) = \sum_j c_j\psi_j(L)$$

- u known: Dirichlet boundary condition
- u' known: Neumann boundary condition
- Must have $\psi_i = 0$ where Dirichlet conditions apply

2.3 The least squares method; principle

$$(R, \frac{\partial R}{\partial c_i}) = 0, \quad i \in \mathcal{I}_V$$

$$\frac{\partial R}{\partial c_i} = \frac{\partial}{\partial c_i}\left(\sum_{j\in\mathcal{I}_V} c_j\psi_j''(x) + f(x)\right) = \psi_i''(x) \tag{22}$$

Because:

$$\frac{\partial}{\partial c_i}\left(c_0\psi_0'' + c_1\psi_1'' + \cdots + c_{i-1}\psi_{i-1}'' + c_i\psi_i'' + c_{i+1}\psi_{i+1}'' + \cdots + c_N\psi_N''\right) = \psi_i''$$

2.4 The least squares method; equation system

$$(\sum_j c_j\psi_j'' + f, \psi_i'') = 0, \quad i \in \mathcal{I}_V \tag{23}$$

Rearrangement:

$$\sum_{j\in\mathcal{I}_V} (\psi_i'', \psi_j'')c_j = -(f, \psi_i''), \quad i \in \mathcal{I}_V \tag{24}$$

This is a linear system

$$\sum_{j\in\mathcal{I}_V} A_{i,j}c_j = b_i, \quad i \in \mathcal{I}_V$$

with

$$\begin{aligned}
A_{i,j} &= (\psi_i'', \psi_j'') \\
&= \pi^4(i+1)^2(j+1)^2L^{-4}\int_0^L \sin\left((i+1)\pi\frac{x}{L}\right)\sin\left((j+1)\pi\frac{x}{L}\right)\,dx \\
&= \begin{cases} \frac{1}{2}L^{-3}\pi^4(i+1)^4 & i = j \\ 0, & i \neq j \end{cases}
\end{aligned} \tag{25}$$

$$b_i = -(f, \psi_i'') = (i+1)^2\pi^2L^{-2}\int_0^L f(x)\sin\left((i+1)\pi\frac{x}{L}\right)\,dx \tag{26}$$

2.5 Orthogonality of the basis functions gives diagonal matrix

Useful property:

$$\int_0^L \sin\left((i+1)\pi\frac{x}{L}\right)\sin\left((j+1)\pi\frac{x}{L}\right)\,dx = \delta_{ij}, \qquad \delta_{ij} = \left\{ \begin{array}{ll} \frac{1}{2}L & i=j \\ 0, & i\neq j \end{array}\right. \tag{27}$$

$\Rightarrow$ $(\psi_i'', \psi_j'') = \delta_{ij}$, i.e., diagonal $A_{i,j}$, and we can easily solve for c_i:

$$c_i = \frac{2L}{\pi^2(i+1)^2}\int_0^L f(x)\sin\left((i+1)\pi\frac{x}{L}\right)\,dx \tag{28}$$

2.6 Least squares method; solution

Let's sympy do the work ($f(x) = 2$):

```
from sympy import *
import sys

i, j = symbols('i j', integer=True)
x, L = symbols('x L')
f = 2
a = 2*L/(pi**2*(i+1)**2)
c_i = a*integrate(f*sin((i+1)*pi*x/L), (x, 0, L))
c_i = simplify(c_i)
print c_i
```

$$c_i = 4\frac{L^2\left((-1)^i+1\right)}{\pi^3\left(i^3+3i^2+3i+1\right)}, \quad u(x) = \sum_{k=0}^{N/2}\frac{8L^2}{\pi^3(2k+1)^3}\sin\left((2k+1)\pi\frac{x}{L}\right). \tag{29}$$

Fast decay: $c_2 = c_0/27$, $c_4 = c_0/125$ - only one term might be good enough:

$$u(x) \approx \frac{8L^2}{\pi^3}\sin\left(\pi\frac{x}{L}\right).$$

2.7 The Galerkin method; principle

$R = u'' + f$:

$$(u'' + f, v) = 0, \quad \forall v \in V,$$

or

$$(u'', v) = -(f, v), \quad \forall v \in V \tag{30}$$

This is a *variational formulation* of the differential equation problem.
$\forall v \in V$ means for all basis functions:

$$(\sum_{j\in\mathcal{I}_V} c_j\psi_j'', \psi_i) = -(f, \psi_i), \quad i \in \mathcal{I}_V \tag{31}$$

2.8 The Galerkin method; solution

Since $\psi_i'' \propto \psi_i$, Galerkin's method gives the same linear system and the same solution as the least squares method (in this particular example).

2.9 The collocation method

$R=0$ (i.e.,the differential equation) must be satisfied at $N+1$ points:

$$-\sum_{j\in\mathcal{I}_V} c_j\psi_j''(x_i) = f(x_i),\quad i\in\mathcal{I}_V \tag{32}$$

This is a linear system $\sum_j A_{i,j} = b_i$ with entries

$$A_{i,j} = -\psi_j''(x_i) = (j+1)^2\pi^2L^{-2}\sin\left((j+1)\pi\frac{x_i}{L}\right),\quad b_i = 2$$

Choose: $N=0$, $x_0 = L/2$

$$c_0 = 2L^2/\pi^2$$

2.10 Comparison of the methods

- Exact solution: $u(x) = x(L-x)$
- Galerkin or least squares ($N=0$): $u(x) = 8L^2\pi^{-3}\sin(\pi x/L)$
- Collocation method ($N=0$): $u(x) = 2L^2\pi^{-2}\sin(\pi x/L)$.
- Max error in Galerkin/least sq.: $-0.008L^2$
- Max error in collocation: $0.047L^2$

3 Useful techniques

3.1 Integration by parts

Second-order derivatives will hereafter be integrated by parts

$$\begin{aligned}\int_0^L u''(x)v(x)dx &= -\int_0^L u'(x)v'(x)dx + [vu']_0^L\\ &= -\int_0^L u'(x)v'(x)dx + u'(L)v(L) - u'(0)v(0)\end{aligned} \tag{33}$$

Motivation:

- Lowers the order of derivatives
- Gives more symmetric forms (incl. matrices)
- Enables easy handling of Neumann boundary conditions
- Finite element basis functions φ_i have discontinuous derivatives (at cell boundaries) and are not suited for terms with φ_i''

3.2 Boundary function; principles

- What about nonzero Dirichlet conditions? Say $u(L) = D$
- We always require $\psi_i(L) = 0$ (i.e., $\psi_i = 0$ where Dirichlet conditions applies)
- Problem: $u(L) = \sum_j c_j\psi_j(L) = \sum_j c_j \cdot 0 = 0 \neq D$ - always
- Solution: $u(x) = B(x) + \sum_j c_j\psi_j(x)$
- $B(x)$: user-constructed boundary function that fulfills the Dirichlet conditions
- If $u(L) = D$, $B(L) = D$
- No restrictions of how $B(x)$ varies in the interior of Ω

3.3 Boundary function; example (1)

Dirichlet conditions: $u(0) = C$ and $u(L) = D$. Choose for example

$$B(x) = \frac{1}{L}(C(L-x) + Dx): \qquad B(0) = C,\ B(L) = D$$

$$u(x) = B(x) + \sum_{j\in\mathcal{I}_V} c_j\psi_j(x), \tag{34}$$

$$u(0) = B(0) = C, \quad u(L) = B(L) = D$$

3.4 Boundary function; example (2)

Dirichlet condition: $u(L) = D$. Choose for example

$$B(x) = D: \qquad B(L) = D$$

$$u(x) = B(x) + \sum_{j\in\mathcal{I}_V} c_j\psi_j(x), \tag{35}$$

$$u(L) = B(L) = D$$

3.5 Impact of the boundary function on the space where we seek the solution

- $\{\psi_i\}_{i\in\mathcal{I}_V}$ is a basis for V
- $\sum_{j\in\mathcal{I}_V} c_j\psi_j(x) \in V$
- But $u \notin V$!
- Reason: say $u(0) = C$ and $u \in V$ (any $v \in V$ has $v(0) = C$, then $2u \notin V$ because $2u(0) = 2C$
- When $u(x) = B(x) + \sum_{j\in\mathcal{I}_V} c_j\psi_j(x)$, $B \neq 0$, $B \notin V$ (in general) and $u \notin V$, but $(u - B) \in V$ since $\sum_j c_j\psi_j \in V$

3.6 Abstract notation for variational formulations

The finite element literature (and much FEniCS documentation) applies an abstract notation for the variational formulation:

Find $(u - B) \in V$ such that

$$a(u, v) = L(v) \quad \forall v \in V$$

3.7 Example on abstract notation

$$-u'' = f, \quad u'(0) = C,\ u(L) = D, \quad u = D + \sum_j c_j\psi_j$$

Variational formulation:

$$\int_\Omega u'v'dx = \int_\Omega fvdx \quad - v(0)C\text{or} \quad (u', v') = (f, v) - v(0)C \quad \forall v \in V$$

Abstract formulation: finn $(u - B) \in V$ such that

$$a(u, v) = L(v) \quad \forall v \in V$$

We identify

$$a(u, v) = (u', v'), \quad L(v) = (f, v) - v(0)C$$

3.8 Bilinear and linear forms

- $a(u, v)$ is a *bilinear form*
- $L(v)$ is a *linear form*

Linear form means

$$L(\alpha_1 v_1 + \alpha_2 v_2) = \alpha_1 L(v_1) + \alpha_2 L(v_2),$$

Bilinear form means

$$\begin{aligned} a(\alpha_1 u_1 + \alpha_2 u_2, v) &= \alpha_1 a(u_1, v) + \alpha_2 a(u_2, v), \\ a(u, \alpha_1 v_1 + \alpha_2 v_2) &= \alpha_1 a(u, v_1) + \alpha_2 a(u, v_2) \end{aligned}$$

In nonlinear problems: Find $(u - B) \in V$ such that $F(u; v) = 0\ \forall v \in V$

3.9 The linear system associated with abstract form

$$a(u, v) = L(v) \quad \forall v \in V \quad \Leftrightarrow \quad a(u, \psi_i) = L(\psi_i) \quad i \in \mathcal{I}_V$$

We can now derive the corresponding linear system once and for all:

$$a(\sum_{j\in\mathcal{I}_V} c_j\psi_j, \psi_i)c_j = L(\psi_i) \quad i \in \mathcal{I}_V$$

Because of linearity,

$$\sum_{j\in\mathcal{I}_V} \underbrace{a(\psi_j, \psi_i)}_{A_{i,j}} c_j = \underbrace{L(\psi_i)}_{b_i} \quad i \in \mathcal{I}_V$$

Given $a(u,v)$ and $L(v)$ in a problem, we can immediately generate the linear system:

$$A_{i,j} = a(\psi_j, \psi_i), \quad b_i = L(\psi_i)$$

3.10 Equivalence with minimization problem

If $a(u,v) = a(v,u)$,

$$a(u,v) = L(v) \quad \forall v \in V,$$

is equivalent to minimizing the functional

$$F(v) = \frac{1}{2} a(v,v) - L(v)$$

over all functions $v \in V$. That is,

$$F(u) \leq F(v) \quad \forall v \in V\,.$$

- Much used in the early days of finite elements
- Still much used in structural analysis and elasticity
- Not as general as Galerkin's method (since $a(u,v) = a(v,u)$)

4 Examples on variational formulations

Goal.
Derive variational formulations for many prototype differential equations in 1D that include

- variable coefficients
- mixed Dirichlet and Neumann conditions
- nonlinear coefficients

4.1 Variable coefficient; problem

$$-\frac{d}{dx}\left(\alpha(x)\frac{du}{dx}\right) = f(x), \quad x \in \Omega = [0,L],\ u(0) = C,\ u(L) = D \tag{36}$$

- Variable coefficient $\alpha(x)$
- *Nonzero* Dirichlet conditions at $x = 0$ *and* $x = L$
- Must have $\psi_i(0) = \psi_i(L) = 0$
- $V = \text{span}\{\psi_0, \ldots, \psi_N\}$
- $v \in V$: $v(0) = v(L) = 0$

$$u(x) = B(x) + \sum_{j \in \mathcal{I}_V} c_j \psi_i(x)$$

$$B(x) = C + \frac{1}{L}(D - C)x$$

4.2 Variable coefficient; variational formulation (1)

$$R = -\frac{d}{dx}\left(a\frac{du}{dx}\right) - f$$

Galerkin's method:

$$(R, v) = 0, \quad \forall v \in V,$$

or with integrals:

$$\int_\Omega \left(\frac{d}{dx}\left(\alpha\frac{du}{dx}\right) - f\right) v \,\mathrm{d}x = 0, \quad \forall v \in V\,.$$

4.3 Variable coefficient; variational formulation (2)

Integration by parts:

$$-\int_\Omega \frac{d}{dx}\left(\alpha(x)\frac{du}{dx}\right) v \,\mathrm{d}x = \int_\Omega \alpha(x)\frac{du}{dx}\frac{dv}{dx}\,\mathrm{d}x - \left[\alpha\frac{du}{dx}v\right]_0^L\,.$$

Boundary terms vanish since $v(0) = v(L) = 0$

Variational formulation.
Find $(u - B) \in V$ such that

$$\int_\Omega \alpha(x)\frac{du}{dx}\frac{dv}{dx}dx = \int_\Omega f(x)v dx, \quad \forall v \in V,$$

Compact notation:

$$\underbrace{(\alpha u', v')}_{a(u,v)} = \underbrace{(f, v)}_{L(v)}, \quad \forall v \in V$$

4.4 Variable coefficient; linear system (the easy way)

With

$$a(u, v) = (\alpha u', v), \quad L(v) = (f, v)$$

we can just use the formula for the linear system:

$$A_{i,j} = a(\psi_j, \psi_i) = (\alpha\psi_j', \psi_i') = \int_\Omega \alpha\psi_j'\psi_i' \,\mathrm{d}x = \int_\Omega \psi_i'\alpha\psi_j' \,\mathrm{d}x = a(\psi_i, \psi_j) = A_{j,i}$$

$$b_i = (f, \psi_i) = \int_\Omega f\psi_i \,\mathrm{d}x$$

4.5 Variable coefficient; linear system (full derivation)

$v = \psi_i$ and $u = B + \sum_j c_j\psi_j$:

$$(\alpha B' + \alpha \sum_{j\in\mathcal{I}_V} c_j\psi_j', \psi_i') = (f,\psi_i), \quad i\in\mathcal{I}_V\,.$$

Reorder to form linear system:

$$\sum_{j\in\mathcal{I}_V} (\alpha\psi_j', \psi_i')c_j = (f,\psi_i) + (a(D-C)L^{-1}, \psi_i'), \quad i\in\mathcal{I}_V\,.$$

This is $\sum_j A_{i,j}c_j = b_i$ with

$$\begin{aligned} A_{i,j} &= (a\psi_j', \psi_i') = \int_\Omega \alpha(x)\psi_j'(x)\psi_i'(x)\,\mathrm{d}x \\ b_i &= (f,\psi_i) + (a(D-C)L^{-1}, \psi_i') = \int_\Omega \left(f(x)\psi_i(x) + \alpha(x)\frac{D-C}{L}\psi_i'(x)\right)\mathrm{d}x \end{aligned}$$

4.6 First-order derivative in the equation and boundary condition; problem

$$-u''(x) + bu'(x) = f(x), \quad x\in\Omega = [0,L],\ u(0) = C,\ u'(L) = E \tag{37}$$

New features:

- first-order derivative u' in the equation
- boundary condition with u': $u'(L) = E$

Initial steps:

- Must force $\psi_i(0) = 0$ because of Dirichlet condition at $x = 0$
- Boundary function: $B(x) = C(L-x)$ or just $B(x) = C$
- No requirements on $\psi_i(L)$ (no Dirichlet condition at $x = L$)

4.7 First-order derivative in the equation and boundary condition; details

$$u = C + \sum_{j\in\mathcal{I}_V} c_j\psi_i(x)$$

Galerkin's method: multiply by v, integrate over Ω, integrate by parts.

$$(-u'' + bu' - f, v) = 0, \quad \forall v\in V$$

$$(u', v') + (bu', v) = (f, v) + [u'v]_0^L, \quad \forall v\in V$$

Now, $[u'v]_0^L = u'(L)v(L) = Ev(L)$ because $v(0) = 0$ and $u'(L) = E$:

$$(u'v') + (bu', v) = (f, v) + Ev(L), \quad \forall v\in V$$

4.8 First-order derivative in the equation and boundary condition; observations

$$(u'v') + (bu', v) = (f, v) + Ev(L), \quad \forall v \in V,$$

Important:

- The boundary term can be used to implement Neumann conditions
- Forgetting the boundary term implies the condition $u' = 0$ (!)
- Such conditions are called *natural boundary conditions*

4.9 First-order derivative in the equation and boundary condition; abstract notation

Abstract notation:

$$a(u, v) = L(v) \quad \forall v \in V$$

Here:

$$\begin{aligned} a(u, v) &= (u', v') + (bu', v) \\ L(v) &= (f, v) + Ev(L) \end{aligned}$$

4.10 First-order derivative in the equation and boundary condition; linear system

Insert $u = C + \sum_j c_j \psi_j$ and $v = \psi_i$:

$$\sum_{j\in\mathcal{I}_V} \underbrace{((\psi_j', \psi_i') + (b\psi_j', \psi_i))}_{A_{i,j}} c_j = \underbrace{(f, \psi_i) + E\psi_i(L)}_{b_i}$$

Observation: $A_{i,j}$ is not symmetric because of the term

$$(b\psi_j', \psi_i) = \int_\Omega b\psi_j'\psi_i dx \neq \int_\Omega b\psi_i'\psi_j dx = (\psi_i', b\psi_j)$$

4.11 Terminology: natural and essential boundary conditions

$$(u', v') + (bu', v) = (f, v) + u'(L)v(L) - u'(0)v(0)$$

- Note: forgetting the boundary terms implies $u'(L) = u'(0) = 0$ (unless prescribe a Dirichlet condition)
- Conditions on u' are simply inserted in the variational form and called *natural conditions*
- Conditions on u at $x = 0$ requires modifying V (through $\psi_i(0) = 0$) and are known as *essential conditions*

Lesson learned.
It is easy to forget the boundary term when integrating by parts. That mistake may prescribe a condition on u'!

4.12 Nonlinear coefficient; problem

Problem:

$$-(\alpha(u)u')' = f(u), \quad x \in [0,L],\ u(0)=0,\ u'(L)=E \tag{38}$$

- V: basis $\{\psi_i\}_{i\in\mathcal{I}_V}$ with $\psi_i(0)=0$ because of $u(0)=0$
- How does the nonlinear coefficients $\alpha(u)$ and $f(u)$ impact the variational formulation?
- (Not much!)

4.13 Nonlinear coefficient; variational formulation

Galerkin: multiply by v, integrate, integrate by parts

$$\int_0^L \alpha(u)\frac{du}{dx}\frac{dv}{dx}\,\mathrm{d}x = \int_0^L f(u)v\,\mathrm{d}x + [\alpha(u)vu']_0^L \quad \forall v\in V$$

- $\alpha(u(0))v(0)u'(0)=0$ since $v(0)$
- $\alpha(u(L))v(L)u'(L)=\alpha(u(L))v(L)E$ since $u'(L)=E$

$$\int_0^L \alpha(u)\frac{du}{dx}\frac{dv}{dx}v\,\mathrm{d}x = \int_0^L f(u)v\,\mathrm{d}x + \alpha(u(L))v(L)E \quad \forall v\in V$$

or

$$(\alpha(u)u', v') = (f(u), v) + \alpha(u(L))v(L)E \quad \forall v\in V$$

4.14 Nonlinear coefficient; where does the nonlinearity cause challenges?

- Abstract notation: no $a(u,v)$ and $L(v)$ because a and L are nonlinear
- Instead: $F(u;v)=0\ \forall v\in V$
- What about forming a linear system? We get a *nonlinear* system of algebraic equations
- Must use methods like Picard iteration or Newton's method to solve nonlinear algebraic equations
- But: the variational formulation was not much affected by nonlinearities

4.15 Computing with Dirichlet and Neumann conditions; problem

$$-u''(x) = f(x), \quad x\in\Omega=[0,1], \quad u'(0)=C,\ u(1)=D$$

- Use a *global* polynomial basis $\psi_i \sim x^i$ on $[0,1]$
- Because of $u(1)=D$: $\psi_i(1)=0$
- Basis: $\psi_i(x)=(1-x)^{i+1}$, $i\in\mathcal{I}_V$
- $B(x)=Dx$

4.16 Computing with Dirichlet and Neumann conditions; details

$$A_{i,j} = (\psi'_j, \psi'_i) = \int_0^1 \psi'_i(x)\psi'_j(x)dx = \int_0^1 (i+1)(j+1)(1-x)^{i+j}dx,$$

Choose $f(x) = 2$:

$$\begin{aligned} b_i &= (2, \psi_i) - (D, \psi'_i) - C\psi_i(0) \\ &= \int_0^1 \left(2(1-x)^{i+1} - D(i+1)(1-x)^i\right) dx - C\psi_i(0) \end{aligned}$$

Can easily do the integrals with `sympy`. $N = 1$:

$$\left(\begin{array}{cc} 1 & 1 \\ 1 & 4/3 \end{array}\right) \left(\begin{array}{c} c_0 \\ c_1 \end{array}\right) = \left(\begin{array}{c} -C + D + 1 \\ 2/3 - C + D \end{array}\right)$$

$$c_0 = -C + D + 2, \quad c_1 = -1,$$

$$u(x) = 1 - x^2 + D + C(x-1) \quad \text{(exact solution)}$$

4.17 When the numerical method is exact

Assume that apart from boundary conditions, u_e lies in the same space V as where we seek u:

$$u = B + F, \quad F \in V a(B+F, v) \quad = L(v) \quad \forall v \in V u_\text{e} = B + E, \quad E \in V a(B+E, v) \quad = L(v) \quad \forall v \in V$$

Subtract: $a(F - E, v) = 0 \;\Rightarrow\; E = F$ and $u = u_\text{e}$

5 Computing with finite elements

Tasks:

- Address the model problem $-u''(x) = 2$, $u(0) = u(L) = 0$
- Uniform finite element mesh with P1 elements
- Show all finite element computations in detail

5.1 Variational formulation, finite element mesh, and basis

$$-u''(x) = 2, \quad x \in (0, L),\ u(0) = u(L) = 0,$$

Variational formulation:

$$(u', v') = (2, v) \quad \forall v \in V$$

Since $u(0) = 0$ and $u(L) = 0$, we must force

$$v(0) = v(L) = 0, \quad \psi_i(0) = \psi_i(L) = 0$$

Use finite element basis, but exclude φ_0 and φ_{N_n} since these are not 0 on the boundary:

$$\psi_i = \varphi_{i+1}, \quad i = 0, \ldots, N = N_n - 2$$

Introduce index mapping $\nu(j)$: $\psi_i = \varphi_{\nu(i)}$

$$u = \sum_{j\in\mathcal{I}_V} c_j\varphi_{\nu(i)},\quad i=0,\ldots,N,\quad \nu(j)=j+1$$

Irregular numbering: more complicated $\nu(j)$ table

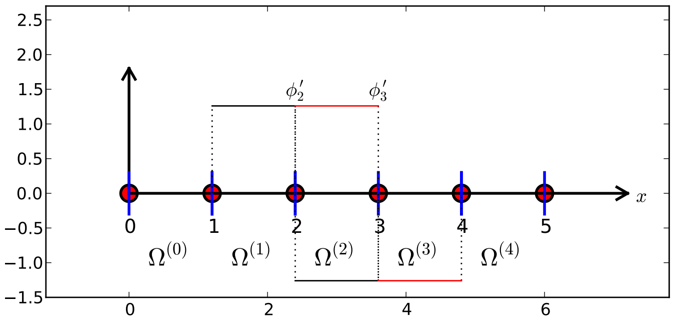
5.2 Computation in the global physical domain; formulas

$$A_{i,j} = \int_0^L \varphi'_{i+1}(x)\varphi'_{j+1}(x)dx,\quad b_i = \int_0^L 2\varphi_{i+1}(x)dx$$

Many will prefer to change indices to obtain a $\varphi'_i\varphi'_j$ product: $i+1\rightarrow i$, $j+1\rightarrow j$

$$A_{i-1,j-1} = \int_0^L \varphi'_i(x)\varphi'_j(x)\,\mathrm{d}x,\quad b_{i-1} = \int_0^L 2\varphi_i(x)\,\mathrm{d}x$$

5.3 Computation in the global physical domain; details



$$\varphi_i = \pm h^{-1}$$

$$A_{i-1,i-1} = h^{-2}2h = 2h^{-1},\quad A_{i-1,i-2} = h^{-1}(-h^{-1})h = -h^{-1},\quad A_{i-1,i} = A_{i-1,i-2}$$

$$b_{i-1} = 2(\frac{1}{2}h+\frac{1}{2}h) = 2h$$

5.4 Computation in the global physical domain; linear system

$$\frac{1}{h}\left(\begin{array}{ccccccccc}
2 & -1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0\\
-1 & 2 & -1 & \ddots & & & & & \vdots\\
0 & -1 & 2 & -1 & \ddots & & & & \vdots\\
\vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots\\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots\\
\vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots\\
\vdots & & & & \ddots & \ddots & \ddots & \ddots & 0\\
\vdots & & & & & \ddots & \ddots & \ddots & -1\\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & -1 & 2
\end{array}\right)
\left(\begin{array}{c} c_0\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ c_N \end{array}\right)
=
\left(\begin{array}{c} 2h\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ 2h \end{array}\right) \tag{39}$$

5.5 Comparison with a finite difference discretization

- Recall: $c_i = u(x_{i+1}) \equiv u_{i+1}$
- Write out a general equation at node $i-1$, expressed by u_i

$$-\frac{1}{h}u_{i-1} + \frac{2}{h}u_i - \frac{1}{h}u_{i+1} = 2h \tag{40}$$

The standard finite difference method for $-u'' = 2$ is

$$-\frac{1}{h^2}u_{i-1} + \frac{2}{h^2}u_i - \frac{1}{h^2}u_{i+1} = 2$$

The finite element method and the finite difference method are identical *in this example.*

(Remains to study the equations involving boundary values)

5.6 Cellwise computations; formulas

- Repeat the previous example, but apply the cellwise algorithm
- Work with one cell at a time
- Transform physical cell to reference cell $X \in [-1, 1]$

$$A^{(e)}_{i-1,j-1} = \int_{\Omega^{(e)}} \varphi_i'(x)\varphi_j'(x)\,\mathrm{d}x = \int_{-1}^{1} \frac{d}{dx}\tilde{\varphi}_r(X)\frac{d}{dx}\tilde{\varphi}_s(X)\frac{h}{2}\,\mathrm{d}X,$$

$$\tilde{\varphi}_0(X) = \frac{1}{2}(1-X), \quad \tilde{\varphi}_1(X) = \frac{1}{2}(1+X)$$

$$\frac{d\tilde{\varphi}_0}{dX} = -\frac{1}{2}, \quad \frac{d\tilde{\varphi}_1}{dX} = \frac{1}{2}$$

From the chain rule

$$\frac{d\tilde{\varphi}_r}{dx} = \frac{d\tilde{\varphi}_r}{dX}\frac{dX}{dx} = \frac{2}{h}\frac{d\tilde{\varphi}_r}{dX}$$

5.7 Cellwise computations; details

$$A^{(e)}_{i-1,j-1} = \int_{\Omega^{(e)}} \varphi_i'(x)\varphi_j'(x)\,\mathrm{d}x = \int_{-1}^{1} \frac{2}{h}\frac{d\tilde{\varphi}_r}{dX}\frac{2}{h}\frac{d\tilde{\varphi}_s}{dX}\frac{h}{2}\,\mathrm{d}X = \tilde{A}^{(e)}_{r,s}$$

$$b^{(e)}_{i-1} = \int_{\Omega^{(e)}} 2\varphi_i(x)\,\mathrm{d}x = \int_{-1}^{1} 2\tilde{\varphi}_r(X)\frac{h}{2}\,\mathrm{d}X = \tilde{b}^{(e)}_r, \quad i = q(e,r),\ r = 0,1$$

Must run through all $r, s = 0, 1$ and $r = 0, 1$ and compute each entry in the element matrix and vector:

$$\tilde{A}^{(e)} = \frac{1}{h}\left(\begin{array}{rr} 1 & -1 \\ -1 & 1 \end{array}\right), \quad \tilde{b}^{(e)} = h\left(\begin{array}{c} 1 \\ 1 \end{array}\right). \tag{41}$$

Example:

$$\tilde{A}^{(e)}_{0,1} = \int_{-1}^{1} \frac{2}{h}\frac{d\tilde{\varphi}_0}{dX}\frac{2}{h}\frac{d\tilde{\varphi}_1}{dX}\frac{h}{2}\,\mathrm{d}X = \frac{2}{h}(-\frac{1}{2})\frac{2}{h}\frac{1}{2}\frac{h}{2}\int_{-1}^{1}\mathrm{d}X = -\frac{1}{h}$$

5.8 Cellwise computations; details of boundary cells

- The boundary cells involve only one unknown
- $\Omega^{(0)}$: left node value known, only a contribution from right node
- $\Omega^{(N_e)}$: right node value known, only a contribution from left node

For $e = 0$ and $= N_e$:

$$\tilde{A}^{(e)} = \frac{1}{h}\left(\begin{array}{c} 1 \end{array}\right), \quad \tilde{b}^{(e)} = h\left(\begin{array}{c} 1 \end{array}\right)$$

Only one degree of freedom ("node") in these cells ($r = 0$)

5.9 Cellwise computations; assembly

4 P1 elements:

```
vertices = [0, 0.5, 1, 1.5, 2]
cells = [[0, 1], [1, 2], [2, 3], [3, 4]]
dof_map = [[0], [0, 1], [1, 2], [2]]       # only 1 dof in elm 0, 3
```

Python code for the assembly algorithm:

```
# Ae[e][r,s]: element matrix, be[e][r]: element vector
# A[i,j]: coefficient matrix, b[i]: right-hand side

for e in range(len(Ae)):
    for r in range(Ae[e].shape[0]):
        for s in range(Ae[e].shape[1]):
            A[dof_map[e,r],dof_map[e,s]] += Ae[e][i,j]
        b[dof_map[e,r]] += be[e][i,j]
```

Result: same linear system as arose from computations in the physical domain

5.10 General construction of a boundary function

- Now we address nonzero Dirichlet conditions
- $B(x)$ is not always easy to construct (extend to the interior of Ω), especially not in 2D and 3D
- With finite element φ_i, $B(x)$ can be constructed in a completely general way
- I_b: set of indices with nodes where u is known
- U_i: Dirichlet value of u at node i, $i \in I_b$

$$B(x) = \sum_{j\in I_b} U_j\varphi_j(x) \tag{42}$$

Suppose we have a Dirichlet condition $u(x_k) = U_k$, $k \in I_b$:

$$u(x_k) = \sum_{j\in I_b} U_j \underbrace{\varphi_j(x)}_{\neq 0 hbox only for j=k} + \sum_{j\in\mathcal{I}_V} c_j \underbrace{\varphi_{\nu(j)}(x_k)}_{=0,\ k\notin\mathcal{I}_V} = U_k$$

5.11 Example with two Dirichlet values; variational formulation

$$-u'' = 2, \quad u(0) = C,\ u(L) = D$$

$$\int_0^L u'v' \,\mathrm{d}x = \int_0^L 2v \,\mathrm{d}x \quad \forall v \in V$$

$$(u', v') = (2, v) \quad \forall v \in V$$

5.12 Example with two Dirichlet values; boundary function

$$B(x) = \sum_{j\in I_b} U_j \varphi_j(x) \tag{43}$$

Here $I_b = \{0, N_n\}$, $U_0 = C$, $U_{N_n} = D$,

$$\psi_i = \varphi_{\nu(i)}, \nu(i) = i+1, \quad 0, \ldots, N = N_n - 2$$

$$u(x) = C\varphi_0(x) + D\varphi_{N_n}(x) + \sum_{j\in\mathcal{I}_V} c_j \varphi_{\nu(j)} \tag{44}$$

5.13 Example with two Dirichlet values; details

Insert $u = B + \sum_j c_j \varphi_{\nu(j)}$ in variational formulation:

$$A_{i,j} = \int_0^L \psi_i'(x)\psi_j'(x) \,\mathrm{d}x, \quad b_i = \int_0^L (f(x) - B'(x))\psi_i(x) \,\mathrm{d}x$$

$$u(x) = \underbrace{C\cdot\varphi_0 + D\varphi_{N_n}}_{B(x)} + \sum_{j\in\mathcal{I}_V} c_j \varphi_{j+1} \quad = C\cdot\varphi_0 + D\varphi_{N_n} + c_0\varphi_1 + c_1\varphi_2 + \cdots + c_N\varphi_{N_n-1}\,.$$

$$A_{i-1,j-1} = \int_0^L \varphi_i'(x)\varphi_j'(x) \,\mathrm{d}x, \quad b_{i-1} = \int_0^L (f(x) - C\varphi_0'(x) - D\varphi_{N_n}'(x))\varphi_i(x) \,\mathrm{d}x,$$

for $i, j = 1, \ldots, N+1 = N_n - 1$.

New boundary terms from $-\int B'\varphi_i \,\mathrm{d}x$: $C/2$ from $e = 0$ and $-D/2$ from $e = N_e$

5.14 Example with two Dirichlet values; cellwise computations

- Element matrices as in the previous example (with $u = 0$ on the boundary)
- New element vector in the first and last cell

From the last cell:

$$\tilde b_0^{(e)} = \int_{-1}^{1} \left(f - D\frac{2}{h}\frac{d\tilde\varphi_1}{dX}\right)\tilde\varphi_0 \frac{h}{2} \,\mathrm{d}X = (\frac{h}{2}(2 - D\frac{2}{h}\frac{1}{2})\int_{-1}^{1} \tilde\varphi_0 \,\mathrm{d}X = h - D/2, \quad e = N_e = N_n - 1$$

From the first cell:

$$\tilde b_0^{(0)} = \int_{-1}^{1} \left(f - C\frac{2}{h}\frac{d\tilde\varphi_0}{dX}\right)\tilde\varphi_1 \frac{h}{2} \,\mathrm{d}X = (\frac{h}{2}(2 + C\frac{2}{h}\frac{1}{2})\int_{-1}^{1} \tilde\varphi_1 \,\mathrm{d}X = h + C/2\,.$$

5.15 Modification of the linear system; ideas

- Method 1: incorporate Dirichlet values through a $B(x)$ function and demand $\psi_i = 0$ where Dirichlet values apply
- Method 2: drop $B(x)$, drop demands to ψ_i, just assemble as if there were no Dirichlet conditions, and modify the linear system instead

Method 2: always $\psi_i = \varphi_i$ and

$$u(x) = \sum_{j\in\mathcal{I}_V} c_j\varphi_j(x),\quad \mathcal{I}_V = \{0,\ldots,N=N_n\} \tag{45}$$

u is treated as unknown at $x = 0$ and $x = L$ when computing entires in the linear system

5.16 Modification of the linear system; original system

$$-u'' = 2,\quad u(0) = 0,\ u(L) = D$$

$$\frac{1}{h}\begin{pmatrix}
1 & -1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0\\
-1 & 2 & -1 & \ddots & & & & & \vdots\\
0 & -1 & 2 & -1 & \ddots & & & & \vdots\\
\vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots\\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots\\
\vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots\\
\vdots & & & & \ddots & \ddots & \ddots & \ddots & 0\\
\vdots & & & & & \ddots & \ddots & \ddots & -1\\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & -1 & 1
\end{pmatrix}
\begin{pmatrix} c_0\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ c_N\end{pmatrix}
=
\begin{pmatrix} h\\ 2h\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ 2h\\ h\end{pmatrix} \tag{46}$$

5.17 Modification of the linear system; row replacement

- Dirichlet condition $u(x_k) = U_k$ means $c_k = U_k$ (since $c_k = u(x_k)$)
- Replace first row by $c_0 = 0$
- Replace last row by $c_N = D$

$$\frac{1}{h}\begin{pmatrix}
1 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0\\
-1 & 2 & -1 & \ddots & & & & & \vdots\\
0 & -1 & 2 & -1 & \ddots & & & & \vdots\\
\vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots\\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots\\
\vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots\\
\vdots & & & & \ddots & \ddots & \ddots & \ddots & 0\\
\vdots & & & & & \ddots & \ddots & \ddots & -1\\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix} c_0\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ c_N\end{pmatrix}
=
\begin{pmatrix} 0\\ 2h\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ \vdots\\ 2h\\ D\end{pmatrix} \tag{47}$$

5.18 Modification of the linear system; element matrix/vector

In cell 0 we know u for local node (degree of freedom) $r = 0$. Replace the first cell equation by $\tilde{c}_0 = 0$:

$$\tilde{A}^{(0)} = A = \frac{1}{h}\left(\begin{array}{cc} 1 & 0 \\ -1 & 1 \end{array}\right), \quad \tilde{b}^{(0)} = \left(\begin{array}{c} 0 \\ h \end{array}\right) \tag{48}$$

In cell N_e we know u for local node $r = 1$. Replace the last equation in the cell system by $\tilde{c}_1 = D$:

$$\tilde{A}^{(N_e)} = A = \frac{1}{h}\left(\begin{array}{cc} 1 & -1 \\ 0 & 1 \end{array}\right), \quad \tilde{b}^{(N_e)} = \left(\begin{array}{c} h \\ D \end{array}\right) \tag{49}$$

5.19 Symmetric modification of the linear system; algorithm

- The modification above destroys symmetry of the matrix: $A_{0,1} \neq A_{1,0}$)
- Symmetry is often important in 2D and 3D (faster computations)
- A more complex modification preserves symmetry

Algorithm for incorporating $c_i = U_i$:

1. Subtract column i times U_i from the right-hand side
2. Zero out column and row no i
3. Place 1 on the diagonal
4. Set $b_i = U_i$

5.20 Symmetric modification of the linear system; example

$$\frac{1}{h}\left(\begin{array}{cccccccccc}
1 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\
0 & 2 & -1 & \ddots & & & & & \vdots \\
0 & -1 & 2 & -1 & \ddots & & & & \vdots \\
\vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots \\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\
\vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots \\
\vdots & & & & \ddots & \ddots & \ddots & \ddots & 0 \\
\vdots & & & & & \ddots & \ddots & \ddots & 0 \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0 & 1
\end{array}\right)
\left(\begin{array}{c} c_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ c_N \end{array}\right)
=
\left(\begin{array}{c} 0 \\ 2h \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 2h + D/h \\ D \end{array}\right) \tag{50}$$

5.21 Symmetric modification of the linear system; element level

Symmetric modification applied to $\tilde{A}^{(N_e)}$:

$$\tilde{A}^{(N_e)} = A = \frac{1}{h}\left(\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right), \quad \tilde{b}^{(N-1)} = \left(\begin{array}{c} h + D/h \\ D \end{array}\right) \tag{51}$$

5.22 Boundary conditions: specified derivative

Focus.
How can we incorporate $u'(0) = C$ with finite elements?

$$-u'' = f, \quad u'(0) = C,\ u(L) = D$$

5.23 The variational formulation

Galerkin's method:

$$\int_0^L (u''(x) + f(x))\psi_i(x)dx = 0, \quad i \in \mathcal{I}_V$$

Integration of $u''\varphi_i$ by parts:

$$\int_0^L u'(x)'\psi_i'(x)\,\mathrm{d}x - (u'(L)\psi_i(L) - u'(0)\psi_i(0)) = \int_0^L f(x)\psi_i(x)\,\mathrm{d}x, \quad i \in \mathcal{I}_V\,.$$

- $u'(L)\varphi_i(L) = 0$ since $\psi_i(L) = 0$
- $u'(0)\varphi_i(0) = C\varphi_i(0)$ since $u'(0) = C$

5.24 Boundary function and exclusion of Dirichlet degrees of freedom

- $B(x) = D\varphi_{N_n}(x)$
- $\psi_i = \varphi_i$, $i \in \mathcal{I}_V = \{= 0, \ldots, N = N_n - 1\}$

$$\int_0^L u'(x)\varphi_i'(x)dx + C\varphi_i(0) = \int_0^L f(x)\varphi_i(x)dx, \quad i \in \mathcal{I}_V$$

5.25 What if we use all φ_i and insert the Dirichlet condition in the linear system?

- Now $\psi_i = \varphi_i$, $i = 0, \ldots, N = N_n$
- $\varphi_N(L) \neq 0$, so $u'(L)\varphi_N(L) \neq 0$
- However, the term $u'(L)\varphi_N(L)$ in b_N will be erased when we insert the Dirichlet value in $b_N = D$

We can forget about the term $u'(L)\varphi_i(L)$!

Result.
Boundary terms $u'\varphi_i$ at points where Dirichlet values apply can always be forgotten.

5.26 Linear system

With boundary function:

$$u(x) = B(x) + \sum_{j=0}^{N} c_j\varphi_j(x), \quad B(x) = D\varphi_{N_n}(x),$$

$$\sum_{j=0}^{N=N_n-1} \left(\int_0^L \varphi_i'(x)\varphi_j'(x)dx\right) c_j = \int_0^L (f(x)\varphi_i(x) - D\varphi_N'(x)\varphi_i(x))\,dx - C\varphi_i(0), \tag{52}$$

for $i = 0, \ldots, N = N_n - 1$.

Without boundary function (assemble entries for $i, j = 0, \ldots, N = N_n$ and then modify the last equation to $c_N = D$):

$$u(x) = \sum_{j=0}^{N=N_n} c_j\varphi_j(x)$$

5.27 How the Neumann condition impacts the element matrix and vector

The extra term $C\varphi_0(0)$ affects only the element vector from the first element since $\varphi_0 = 0$ on all other elements.

$$\tilde{A}^{(0)} = A = \frac{1}{h}\left(\begin{array}{cc} 1 & 1 \\ -1 & 1 \end{array}\right), \quad \tilde{b}^{(0)} = \left(\begin{array}{c} h - C \\ h \end{array}\right) \tag{53}$$

6 The finite element algorithm

The differential equation problem defines the integrals in the variational formulation.

Request these functions from the user:

```
integrand_lhs(phi, r, s, x)
boundary_lhs(phi, r, s, x)
integrand_rhs(phi, r, x)
boundary_rhs(phi, r, x)
```

Must also have a mesh with `vertices`, `cells`, and `dof_map`

6.1 Python pseudo code; the element matrix and vector

```
<Declare global matrix and rhs: A, b>

# Loop over all cells
for e in range(len(cells)):

    # Compute element matrix and vector
    n = len(dof_map[e])  # no of dofs in this element
    h = vertices[cells[e][1]] - vertices[cells[e][1]]
    <Declare element matrix and vector: A_e, b_e>

    # Integrate over the reference cell
    points, weights = <numerical integration rule>
    for X, w in zip(points, weights):
        phi = <basis functions and derivatives at X>
        detJ = h/2
        x = <affine mapping from X>
        for r in range(n):
            for s in range(n):
                A_e[r,s] += integrand_lhs(phi, r, s, x)*detJ*w
            b_e[r] += integrand_rhs(phi, r, x)*detJ*w

    # Add boundary terms
    for r in range(n):
        for s in range(n):
            A_e[r,s] += boundary_lhs(phi, r, s, x)*detJ*w
        b_e[r] += boundary_rhs(phi, r, x)*detJ*w
```

6.2 Python pseudo code; boundary conditions and assembly

```
for e in range(len(cells)):
    ...

    # Incorporate essential boundary conditions
    for r in range(n):
        global_dof = dof_map[e][r]
        if global_dof in essbc_dofs:
            # dof r is subject to an essential condition
            value = essbc_docs[global_dof]
            # Symmetric modification
            b_e -= value*A_e[:,r]
            A_e[r,:] = 0
            A_e[:,r] = 0
            A_e[r,r] = 1
            b_e[r] = value

    # Assemble
    for r in range(n):
        for s in range(n):
            A[dof_map[e][r], dof_map[e][r]] += A_e[r,s]
        b[dof_map[e][r] += b_e[r]

<solve linear system>
```

7 Variational formulations in 2D and 3D

How to do integration by parts is the major difference when moving to 2D and 3D.

7.1 Integration by parts

$$\nabla\cdot(a(\boldsymbol{x})\nabla u) = \frac{\partial}{\partial x}\left(a(x,y)\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(a(x,y)\frac{\partial u}{\partial y}\right)$$

Integration by parts.

$$-\int_\Omega \nabla\cdot(a(\boldsymbol{x})\nabla u)v\,\mathrm{d}x = \int_\Omega a(\boldsymbol{x})\nabla u\cdot\nabla v\,\mathrm{d}x - \int_{\partial\Omega} a\frac{\partial u}{\partial n}v\,\mathrm{d}s, \qquad (54)$$

- $\int_\Omega ()\,\mathrm{d}x$: area (2D) or volume (3D) integral

- $\int_{\partial\Omega}()\,\mathrm{d}s$: line(2D) or surface (3D) integral

- $\partial\Omega_N$: Neumann conditions $-a\frac{\partial u}{\partial n} = g$
- $\partial\Omega_D$: Dirichlet conditions $u = u_0$
- $v \in V$ must vanish on $\partial\Omega_D$

7.2 Example on integration by parts; problem

$$\boldsymbol{v}\cdot\nabla u + \alpha u = \nabla\cdot(a\nabla u) + f, \quad \boldsymbol{x}\in\Omega \tag{55}$$

$$u = u_0, \quad \boldsymbol{x}\in\partial\Omega_D \tag{56}$$

$$-a\frac{\partial u}{\partial n} = g, \quad \boldsymbol{x}\in\partial\Omega_N \tag{57}$$

- Known: a, α, f, u_0, and g.
- Second-order PDE: must have *exactly one boundary condition at each point of the boundary*

$$u(\boldsymbol{x}) = B(\boldsymbol{x}) + \sum_{j\in\mathcal{I}_V} c_j\psi_j(\boldsymbol{x}), \quad B(\boldsymbol{x}) = u_0(\boldsymbol{x})$$

7.3 Example on integration by parts; details (1)

Galerkin's method: multiply by $v\in V$ and integrate over Ω,

$$\int_\Omega(\boldsymbol{v}\cdot\nabla u + \alpha u)v\,\mathrm{d}x = \int_\Omega \nabla\cdot(a\nabla u)\,\mathrm{d}x + \int_\Omega fv\,\mathrm{d}x$$

Integrate second-order term by parts:

$$\int_\Omega \nabla\cdot(a\nabla u)\,v\,\mathrm{d}x = -\int_\Omega a\nabla u\cdot\nabla v\,\mathrm{d}x + \int_{\partial\Omega} a\frac{\partial u}{\partial n}v\,\mathrm{d}s,$$

Result:

$$\int_\Omega(\boldsymbol{v}\cdot\nabla u + \alpha u)v\,\mathrm{d}x = -\int_\Omega a\nabla u\cdot\nabla v\,\mathrm{d}x + \int_{\partial\Omega} a\frac{\partial u}{\partial n}v\,\mathrm{d}s + \int_\Omega fv\,\mathrm{d}x$$

7.4 Example on integration by parts; details (2)

Note: $v\neq 0$ only on $\partial\Omega_N$:

$$\int_{\partial\Omega} a\frac{\partial u}{\partial n}v\,\mathrm{d}s = \int_{\partial\Omega_N} a\frac{\partial u}{\partial n}v\,\mathrm{d}s$$

Insert flux condition $a\frac{\partial u}{\partial n} = -g$ on $\partial\Omega_N$:

$$\int_{\partial\Omega_N} a\frac{\partial u}{\partial n}v\,\mathrm{d}s = -\int_{\partial\Omega_N} gv\,\mathrm{d}s$$

The final variational form:

$$\int_\Omega(\boldsymbol{v}\cdot\nabla u + \alpha u)v\,\mathrm{d}x = -\int_\Omega a\nabla u\cdot\nabla v\,\mathrm{d}x - \int_{\partial\Omega} gv\,\mathrm{d}s + \int_\Omega fv\,\mathrm{d}x$$

With inner product notation:

$$(\boldsymbol{v}\cdot\nabla u, v) + (\alpha u, v) = -(a\nabla u, \nabla v) - (g, v)_N + (f, v)$$

$(g, v)_N$: line or surface integral over $\partial\Omega_N$.

7.5 Example on integration by parts; linear system

$$u = u_0 + \sum_{j\in\mathcal{I}_V} c_j\psi_j$$

$$A_{i,j} = (\boldsymbol{v}\cdot\nabla\varphi_j, \varphi_i) + (\alpha\varphi_j, \varphi_i) + (a\nabla\varphi_j, \nabla\varphi_i)$$

$$b_i = (g, \varphi_i)_N + (f, \varphi_i) - (\boldsymbol{v}\cdot\nabla u_0, \varphi_i) + (\alpha u_0, \varphi_i) + (a\nabla u_0, \nabla\varphi_i),$$

7.6 Transformation to a reference cell in 2D and 3D (1)

We want to compute an integral in the physical domain by integrating over the reference cell.

$$\int_{\Omega^{(e)}} a(\boldsymbol{x})\nabla\varphi_i\cdot\nabla\varphi_j\,\mathrm{d}x \tag{58}$$

Mapping from reference to physical coordinates:

$$\boldsymbol{x}(\boldsymbol{X}),$$

with Jacobian J,

$$J_{i,j} = \frac{\partial x_j}{\partial X_i}$$

- Step 1: $\mathrm{d}x \rightarrow \det J\,\mathrm{d}X$.
- Step 2: express $\nabla\varphi_i$ by an expression with $\tilde{\varphi}_r$ $(i = q(e, r))$
- We want $\nabla_{\boldsymbol{x}}\tilde{\varphi}_r(\boldsymbol{X})$ (derivatives wrt $\boldsymbol{x}$)
- What we readily have: $\nabla_{\boldsymbol{X}}\tilde{\varphi}_r(\boldsymbol{X})$ (derivative wrt $\boldsymbol{X}$)
- Need to transform $\nabla_{\boldsymbol{X}}\tilde{\varphi}_r(\boldsymbol{X})$ to $\nabla_{\boldsymbol{x}}\tilde{\varphi}_r(\boldsymbol{X})$

7.7 Transformation to a reference cell in 2D and 3D (2)

Can derive

$$\begin{aligned}\nabla_{\boldsymbol{X}}\tilde{\varphi}_r &= J\cdot\nabla_{\boldsymbol{x}}\varphi_i\\ \nabla_{\boldsymbol{x}}\varphi_i &= J^{-1}\cdot\nabla_{\boldsymbol{X}}\tilde{\varphi}_r\end{aligned}$$

Integral transformation from physical to reference coordinates:

$$\int_{\Omega}^{(e)} a(\boldsymbol{x})\nabla_{\boldsymbol{x}}\varphi_i\cdot\nabla_{\boldsymbol{x}}\varphi_j\,\mathrm{d}x \int_{\tilde{\Omega}^r} a(\boldsymbol{x}(\boldsymbol{X}))(J^{-1}\cdot\nabla_{\boldsymbol{X}}\tilde{\varphi}_r)\cdot(J^{-1}\cdot\nabla\tilde{\varphi}_s)\det J\,\mathrm{d}X \tag{59}$$

8 Time-dependent problems

- So far: used the finite element framework for discretizing in space
- What about $u_t = u_{xx} + f$?
- Use finite differences in time
- Solve a recursive set of spatial problems by the finite element method

8.1 Example: diffusion problem

$$\frac{\partial u}{\partial t} = \alpha\nabla^2 u + f(\boldsymbol{x}, t), \quad \boldsymbol{x}\in\Omega, t\in(0,T] \tag{60}$$

$$u(\boldsymbol{x}, 0) = I(\boldsymbol{x}), \quad \boldsymbol{x}\in\Omega \tag{61}$$

$$\frac{\partial u}{\partial n} = 0, \quad \boldsymbol{x}\in\partial\Omega,\ t\in(0,T]\,. \tag{62}$$

8.2 A Forward Euler scheme; ideas

$$[D_t^+ u = \alpha\nabla^2 u + f(\boldsymbol{x}, t)]^n, \quad n = 1, 2, \ldots, N_t - 1 \tag{63}$$

$$u^{n+1} = u^n + \Delta t\left(\alpha\nabla^2 u^n + f(\boldsymbol{x}, t_n)\right)\,. \tag{64}$$

- $u^n = \sum_j c_j^n \psi_j$
- Compute u^0 from I
- Compute u^{n+1} from u^n

8.3 A Forward Euler scheme; stages in the discretization

- $u_{\mathrm{e}}(\boldsymbol{x}, t)$: exact solution of the space-and time-continuous problem
- $u_{\mathrm{e}}^n(\boldsymbol{x})$: exact solution of time-discrete problem (after applying a finite difference scheme in time)
- $u^n = \sum_{j\in\mathcal{I}_V} c_j^n\psi_j$: solution at the time- and space-discrete problem (after applying a Galerkin method in space)

$$\frac{\partial u_{\mathrm{e}}}{\partial t} = \alpha\nabla^2 u_{\mathrm{e}} + f(\boldsymbol{x}, t) \tag{65}$$

$$u_{\mathrm{e}}^{n+1} = u_{\mathrm{e}}^n + \Delta t\left(\alpha\nabla^2 u_{\mathrm{e}}^n + f(\boldsymbol{x}, t_n)\right) \tag{66}$$

8.4 A Forward Euler scheme; weighted residual (or Galerkin) principle

$$u_{\mathrm{e}}^n \approx u^n = \sum_{j=0}^{N_s} c_j^n \psi_j(\boldsymbol{x}) \tag{67}$$

$$u_{\mathrm{e}}^{n+1} \approx u^{n+1} = \sum_{j=0}^{N_s} c_j^{n+1} \psi_j(\boldsymbol{x}) \tag{68}$$

$$R = u^{n+1} - u^n - \Delta t \left(\alpha\nabla^2 u^n + f(\boldsymbol{x}, t_n)\right)$$

The weighted residual principle,

$$\int_\Omega R w_i \,\mathrm{d}x = 0, \quad i = 0, \ldots, N_s,$$

results in

$$\int_\Omega \left[u^{n+1} - u^n - \Delta t \left(\alpha\nabla^2 u^n + f(\boldsymbol{x}, t_n)\right)\right] w_i \,\mathrm{d}x = 0, \quad i = 0, \ldots, N$$

Galerkin: $w_i = \psi_i$

8.5 A Forward Euler scheme; integration by parts

Isolating the unknown u^{n+1} on the left-hand side:

$$\int_\Omega u^{n+1}\psi_i \,\mathrm{d}x = \int_\Omega \left[u^n - \Delta t \left(\alpha\nabla^2 u^n + f(\boldsymbol{x}, t_n)\right)\right] \psi_i \,\mathrm{d}x$$

Integration by parts:

$$\int_\Omega \alpha\nabla^2 u^n \psi_i \,\mathrm{d}x = -\int_\Omega \alpha\nabla u^n \cdot \nabla\psi_i \,\mathrm{d}x + \underbrace{\int_{\partial\Omega} \alpha\frac{\partial u^n}{\partial n}\psi \,\mathrm{d}x}_{=0 \text{ because } \partial u^n/\partial n = 0}$$

$$\boxed{\int_\Omega u^{n+1}\psi_i \,\mathrm{d}x = \int_\Omega u^n\psi_i \,\mathrm{d}x - \Delta t \int_\Omega \alpha\nabla u^n \cdot \nabla\psi_i \,\mathrm{d}x + \Delta t \int_\Omega f^n\psi_i \,\mathrm{d}x \qquad (69)}$$

8.6 New notation for the solution at the most recent time levels

- u and `u`: the spatial unknown function to be computed
- u_1 and `u_1`: the spatial function at the previous time level $t - \Delta t$
- u_2 and `u_2`: the spatial function at $t - 2\Delta t$

$$\int_\Omega u\psi_i \,\mathrm{d}x = \int_\Omega u_1\psi_i \,\mathrm{d}x - \Delta t \int_\Omega \alpha\nabla u_1 \cdot \nabla\psi_i \,\mathrm{d}x + \Delta t \int_\Omega f^n\psi_i \,\mathrm{d}x \tag{70}$$

or

$$(u, \psi_i) = (u_1, \psi_i) - \Delta t(\alpha\nabla u_1, \nabla\psi_i) + (f^n, \psi_i) \tag{71}$$

8.7 Deriving the linear systems

$$u = \sum_{j=0}^{N} c_j \psi_j(\boldsymbol{x}), \quad u_1 = \sum_{j=0}^{N} c_{1,j}\psi_j(\boldsymbol{x})$$

Insert these in

$$(u, \psi_i) = (u_1, \psi_i) - \Delta t(\alpha \nabla u_1, \nabla \psi_i) + (f^n, \psi_i) \tag{72}$$

and order terms as matrix-vector products:

$$\sum_{j=0}^{N} \underbrace{(\psi_i, \psi_j)}_{M_{i,j}} c_j = \sum_{j=0}^{N} \underbrace{(\psi_i, \psi_j) c_{1,j}}_{M_{i,j}} - \Delta t \alpha \sum_{j=0}^{N} \underbrace{(\nabla \psi_i, \nabla \psi_j)}_{K_{i,j}} c_{1,j} + (f^n, \psi_i), \quad i = 0, \ldots, N \tag{73}$$

8.8 Structure of the linear systems

$$Mc = Mc_1 - \alpha \Delta t K c_1 + f \tag{74}$$

$$\begin{aligned}
M &= \{M_{i,j}\}, \quad M_{i,j} = (\psi_i, \psi_j), \quad i, j \in \mathcal{I}_V, \\
K &= \{K_{i,j}\}, \quad K_{i,j} = (\nabla \psi_i, \nabla \psi_j), \quad i, j \in \mathcal{I}_V, \\
f &= \{(f(\boldsymbol{x}, t_n), \psi_i)\}_{i \in \mathcal{I}_V}, \\
c &= \{c_i\}_{i \in \mathcal{I}_V}, \\
c_1 &= \{c_{1,i}\}_{i \in \mathcal{I}_V}.
\end{aligned}$$

8.9 Computational algorithm

1. Compute M and K.
2. Initialize u^0 by either interpolation or projection
3. For $n = 1, 2, \ldots, N_t$:
   (a) compute $b = Mc_1 - \alpha \Delta t K c_1 + f$
   (b) solve $Mc = b$
   (c) set $c_1 = c$

Initial condition:

- Interpolation: $c_{1,j} = I(\boldsymbol{x}_j)$ (finite elements)
- Projection: solve $\sum_j M_{i,j} c_{1,j} = (I, \psi_i)$, $i \in \mathcal{I}_V$.

Index