Study Guide: Solving differential equations with finite elements

Hans Petter Langtangen[1,2]

Center for Biomedical Computing, Simula Research Laboratory[1]

Department of Informatics, University of Oslo[2]

Oct 27, 2013

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!

Abstract differential equation

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

Examples (1D problems):

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

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

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

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

Abstract boundary conditions

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

Examples:

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

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

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

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 $\{\psi_i\}_{i\in I}$
- $I = \{0, \ldots, N\}$ is an index set
- $u(x) = \sum_{j\in I} c_j\psi_j(x)$
- Inner product: $(u, v) = \int_\Omega uv\,\mathrm{d}x$
- Norm: $||u|| = \sqrt{(u, u)}$

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 $\{c_i\}_{i\in I}$ (and hope it makes a small e too)

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

The least squares method

Idea: minimize

$$E = ||R||^2 = (R, R) = \int_\Omega R^2 dx \qquad (11)$$

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

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

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

The Galerkin method

Idea: make R orthogonal to V,

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

This implies

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

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

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:

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

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

Terminology: test and Trial Functions

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

The collocation method

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

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

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

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

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

Examples on using the principles

Goal.

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

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:

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

The residual:

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

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

The least squares method; principle

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

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

Because:

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

The least squares method; equation system

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

Rearrangement:

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

This is a linear system

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

with

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

Orthogonality of the basis functions gives diagonal matrix

Useful property:

$$\int_0^L \sin\left((i+1)\pi\frac{x}{L}\right)\sin\left((j+1)\pi\frac{x}{L}\right)\,dx = \delta ij, \quad \Rightarrow (\psi_i'', \psi_j'') = \delta_{ij}, \quad \delta_{ij} = \tag{27}$$

With diagonal $A_{i,j}$ we can easily solve for c_i:

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

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
```

$$c_i = 4\frac{L^2\left((-1)^i + 1\right)}{\pi^3\left(i^3 + 3i^2 + 3i + 1\right)}$$

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

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

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

The Galerkin method; principle

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

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:

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

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).

The collocation method

$R = 0$ or the differential equation must be satisfied at $N+1$ points:

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

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

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

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

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

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: $0.047L^2$

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''

Boundary function; principles

- What about nonzero Dirichlet conditions?
- E.g. $u(L) = D$

Boundary function; example

$u(0) = C$ and $u(L) = D$. Choose

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

$$u(x) = L^{-1}(C(L-x) + Dx) + \sum_{j\in I} c_j\psi_j(x), \tag{34}$$

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

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

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

Example on abstract notation

Given a variational formulation for $-u'' = f$:

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

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

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

We identify

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

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 $F(u; v) = 0 \; \forall v \in V$

The linear system associated with abstract form

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

is equivalent to

$$a(u, \psi_i) = L(\psi_i) \quad i \in I$$

Insert $u = \sum_j c_j \psi_j$ and use linearity:

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

This is a linear system

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

with

$$\begin{aligned} A_{i,j} &= a(\psi_j, \psi_i) \\ b_i &= L(\psi_i) \end{aligned}$$

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

Examples on variational formulations

Goal.

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

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

Variable coefficient; problem

$$-\frac{d}{dx}\left(\alpha(x)\frac{du}{dx}\right) = f(x), \quad x \in \Omega = [0, L],\ u(0) = C,\ u(L) = D\,. \tag{35}$$

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

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

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

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:

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

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

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

Compact notation:

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

Variable coefficient; abstract notation

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

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

Variable coefficient; linear system

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

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

Reorder to form linear system:

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

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

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

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

$$-u''(x) + bu'(x) = f(x), \quad x \in \Omega = [0, L],\ u(0) = C,\ u'(L) = E \tag{36}$$

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: $B(x) = C(L - x)/L$
- No requirements on $\psi_i(L)$ (no Dirichlet condition at $x = L$)

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

$$u = \frac{C}{L}(L-x) + \sum_{j\in I} c_j\psi_i(x)$$

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

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

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

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

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

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

Important:

- $[u'v]_0^L = u'(L)v(L) = Ev(L)$ because $v(0) = 0$ and $u'(L) = E$
- 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*

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

Abstract notation:

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

where

$$a(u, v) = (u', v') + (bu', v), \quad L(v) = (f + C, v) + Ev(L)$$

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

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

$$\sum_{j\in I} \underbrace{((\psi_j', \psi_i') + (b\psi_j', \psi_i))}_{A_{i,j}} c_j = \underbrace{(f, \psi_i) + (bCL^{-1}, \psi_i') + E\psi_i(L)}_{b_i}$$

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

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

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'!

Nonlinear coefficient; problem

$$-(\alpha(u)u')' = f(u), \quad x \in [0, L],\ u(0) = 0,\ u'(L) = E\,. \qquad (37)$$

- V: basis $\{\psi_i\}_{i\in I}$ with $\psi_i(0) = 0$ because of $u(0) = 0$
- How does the nonlinear coefficient $\alpha(u)$ impact the variational formulation?

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

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

Nonlinear coefficient; where does the nonlinearity cause challenges?

- Abstract notation: no $a(u, v)$ and $L(v)$ because a and L are nonlinear
- Abstract notation: $F(u; v) = 0\ \forall v \in V$
- 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

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: $\psi_i(x) = (1-x)^{i+1}$, $i \in I$
- $B(x) = Dx$

Computing with Dirichlet and Neumann conditions; details

$$A_{i,j} = (\psi_j', \psi_i') = \int_0^1 \psi_i'(x)\psi_j'(x)dx = \int_0^1 (i+1)(j+1)(1-x)^{i+j}dx,$$

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

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

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

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

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

When the numerical method is exact

Let

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

Subtract: $a(F - E, v) = 0$ and $E = F$.
Apart from boundary conditions, u_{e} lies in the same as we seek u.
Then $u = u_{\mathrm{e}}$.
!split ========== 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
!split ======= Variational formulation, finite element mesh, and basis =======

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

Variational formulation:

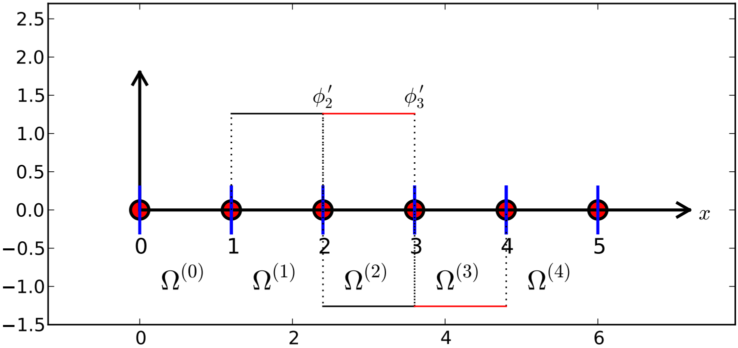
Computation in the global physical domain; formulas

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

$i+1 \rightarrow i$, $j+1 \rightarrow j$

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

Computation in the global physical domain; details



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

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

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

Computation in the global physical domain; linear system

$$\frac{1}{h}\begin{pmatrix}
2 & -1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\
-1 & 2 & -1 & \ddots & & & & & \vdots \\
0 & -1 & 2 & -1 & \ddots & & & & \vdots \\
\vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots \\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\
\vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots \\
\vdots & & & & \ddots & \ddots & \ddots & \ddots & 0 \\
\vdots & & & & & \ddots & \ddots & \ddots & -1 \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & -1 & 2
\end{pmatrix}
\begin{pmatrix} c_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ c_N \end{pmatrix}
=
\begin{pmatrix} 2h \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 2h \end{pmatrix} \tag{38}$$

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

$$-\frac{1}{h}u_{i-1} + \frac{2}{h}u_i - \frac{1}{h}u_{i+1} = 2h \qquad (39)$$

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)

Cellwise computations; formulas

- Repeat the previous example, but apply the cellwise algorithm
- Work with one cell at a time
- Thansform 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

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

Cellwise computations; details

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

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

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

$$\tilde{A}^{(e)} = \frac{1}{h}\left(\begin{array}{rr} 1 & -1 \\ -1 & 1 \end{array}\right), \quad \tilde{b}^{(e)} = h\left(\begin{array}{c} 1 \\ 1 \end{array}\right). \qquad (40)$$

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

$$\tilde{A}^{(e)} = \frac{1}{h}\left(\begin{array}{c} 1 \end{array}\right), \quad \tilde{b}^{(e)} = h\left(\begin{array}{c} 1 \end{array}\right), \quad e = 0,\ e = N_e$$

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

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]]
```

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

Boundary conditions: specified nonzero value

General construction of a boundary function

- $B(x)$ is not always easy to construct (extend to the interior of Ω), at least 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, $i \in I_b$

Example with two Dirichlet values; variational formulation

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

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

Insert $u = B + \sum_j c_j\psi_j$ in variational formulation:

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

Example with two Dirichlet values; details

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

$$u(x) = \underbrace{0 \cdot \varphi_0 + D\varphi_{N_n}}_{B(x)} + \sum_{j\in I} c_j\varphi_{j+1} \quad = D\varphi_{N_n} + c_0\varphi_1 + c_1\varphi_2 + \cdots +$$

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

for $i, j = 1, \ldots, N+1 = N_n - 1$.
New boundary term from B': $-D/2$

Example with two Dirichlet values; cellwise computations

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

$$\tilde{b}_0^{(e)} = \int_{-1}^{1} \left(f - D\frac{2}{h}\frac{\tilde{\varphi}_1}{dX} \right) \tilde{\varphi}_0 \frac{h}{2}\,\mathrm{d}X = (\frac{h}{2}(2 - D\frac{2}{h}\frac{1}{2}) \int_{-1}^{1} \tilde{\varphi}_0\,\mathrm{d}X = h - D/2,$$

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: $\psi_i = \varphi_i$

$$u(x) = \sum_{j\in I} c_j\varphi_j(x), \quad I = \{0,\ldots,N=N_n\} \tag{43}$$

u is treated as unknown at $x = 0$ and $x = L$

Modification of the linear system; linear system

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

$$\frac{1}{h}\begin{pmatrix} 1 & -1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ -1 & 2 & -1 & \ddots & & & & & \vdots \\ 0 & -1 & 2 & -1 & \ddots & & & & \vdots \\ \vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots \\ \vdots & & & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & & & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} h \\ 2h \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 2h \\ h \end{pmatrix} \tag{44}$$

Modification of the linear system; modifications

- Dirichlet condition $u(x_i) = U_i$ means $c_i = U_i$ (since $c_i = u(x_i)$)
- Replace first row by $c_0 = 0$
- Replace last row by $c_N = D$

$$
\frac{1}{h}
\begin{pmatrix}
1 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\
-1 & 2 & -1 & \ddots & & & & & \vdots \\
0 & -1 & 2 & -1 & \ddots & & & & \vdots \\
\vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots \\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\
\vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots \\
\vdots & & & & \ddots & \ddots & \ddots & \ddots & 0 \\
\vdots & & & & & \ddots & \ddots & \ddots & -1 \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
c_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ c_N
\end{pmatrix}
=
\begin{pmatrix}
0 \\ 2h \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 2h \\ D
\end{pmatrix}
\tag{45}
$$

Modification of the linear system; element matrix/vector

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

$$\tilde{A}^{(0)} = A = \frac{1}{h} \left(\begin{array}{rr} 1 & 0 \\ -1 & 1 \end{array} \right), \quad \tilde{b}^{(0)} = \left(\begin{array}{c} 0 \\ h \end{array} \right) \tag{46}$$

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

$$\tilde{A}^{(N_e)} = A = \frac{1}{h} \left(\begin{array}{rr} 1 & -1 \\ 0 & 1 \end{array} \right), \quad \tilde{b}^{(N_e)} = \left(\begin{array}{c} h \\ D \end{array} \right) \tag{47}$$

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 $b_i = U_i$

Symmetric modification of the linear system; example

$$\frac{1}{h}\begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 2 & -1 & \ddots & & & & & \vdots \\ 0 & -1 & 2 & -1 & \ddots & & & & \vdots \\ \vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & 0 & -1 & 2 & -1 & \ddots & \vdots \\ \vdots & & & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & & & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} 0 \\ 2h \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 2h + D/ \\ D \end{pmatrix} \tag{48}$$

Modification of the element matrix and vector

- Modification of the linear system can be done in the the element matrix and vector instead
- Exactly the same procedure

Last degree of freedom in the last element is prescribed:

Boundary conditions: specified derivative

Focus: how to incorporate $u'(0) = C$ with finite elements.