Courses/FEM/modules/assembly

From Icarus

< Courses | FEM

Contents

Assembly of discrete systems

Precondition

- Science
- Function approximation
- Galerkin's method

Theory

The Galerkin finite element method

We consider Poisson's equation in the case $a \equiv 1$, that is

$$-u'' = f, \quad x \in (0,1)$$
$$u(0) = u(1) = 0$$

and formulate the simplest finite element method for bvp based on continuous piecewise linear approximation.

We let $\mathcal{T}_h : 0 = x_0 < x_1 < \ldots < x_{M+1} = 1$, be a *partition* or (*triangulation*) of $I = (0,1)$ into sub-intervals $I_j = (x_{j-1}, x_j)$ of length $h_j = x_j - x_{j-1}$ and let $V_h = V_h^{(1)}$ denote the set of continuous piecewise linear functions on $\mathcal{T}_h$ that are zero at $x = 0$ and $x = 1$. We show an example of such a function in triexam.

In Chapter [#linalg $[*]$], we saw that V_h is a finite dimensional vector space of dimension M with a basis consisting of the hat functions $\{\phi_j\}_{j=1}^M$ illustrated in hatfunc. The coordinates of a function v in V_h in this basis are the values $v(x_j)$ at the interior nodes x_j, $j = 1, \ldots, M$, and a function $v \in V_h$ can be written

$$v(x) = \sum_{j=1}^{M} v(x_j)\phi_j(x).$$

Note that because $v \in V_h$ is zero at 0 and 1, we do not include ϕ_0 and ϕ_{M+1} in the set of basis functions for V_h.

As in the previous example, Galerkin's method is based on stating the differential equation $-u'' = f$ in the form

$$\int_0^1 (-u'' - f)v\,dx = 0 \quad \text{for all functions} v,$$

(1)

corresponding to the residual $-u'' - f$ being orthogonal to the test functions v, cf. residualzerorabbit. However, since the functions in V_h do not have second derivatives, we can't simply plug a candidate for an approximation of u in the space V_h directly into this equation. To get around this technical difficulty, we use integration by parts to move one derivative from u'' onto v assuming v is differentiable and $v(0) = v(1) = 0$:

$$-\int_0^1 u''v\,dx = -u'(1)v(1) + u'(0)v(0) + \int_0^1 u'v'\,dx = \int_0^1 u'v'\,dx,$$

where we used the boundary conditions on v. We are thus led to the following *variational formulation* of bvp: find the function u with $u(0) = u(1) = 0$ such that

$$\int_0^1 u'v'\,dx = \int_0^1 fv\,dx,$$

for all functions v such that $v(0) = v(1) = 0$. We refer to bvpvar as a *weak form* of contresorth.

The Galerkin finite element method for bvp is the following finite-dimensional analog of bvpvar: find $U \in V_h$ such that

$$\int_0^1 U'v'\,dx = \int_0^1 fv\,dx \quad \text{for all } v \in V_h.$$

We note that the derivatives U' and v' of the functions U and $v \in V_h$ are piecewise constant functions of the form depicted in cpwlderi

and are not defined at the nodes x_i. However, the integral with integrand $U'v'$ is nevertheless uniquely defined as the sum of integrals over the sub-intervals. This is due to the basic fact of integration that two functions that are equal except at a finite number of points, have the same integral. We illustrate this in samearea.

By the same token, the value (or lack of value) of U' and v' at the distinct node points x_i does not affect the value of $\int_0^1 U'v'\,dx$.

The equation contresorth expresses the fact that the residual error $-u'' - f$ of the exact solution is orthogonal to *all* test functions v and bvpvar is a reformulation in weak form. Similarly, bvpfem is a way of forcing in weak form the residual error of the finite element solution U to be orthogonal to the finite dimensional set of test functions v in V_h.

The discrete system of equations

Using the basis of hat functions $\{\phi_j\}_{j=1}^M$, we have

$$U(x) = \sum_{j=1}^M \xi_j \phi_j(x)$$

and determine the nodal values $xi_j = U(x_j)$ using the Galerkin orthogonality bvpfem. Substituting, we get

$$\sum_{j=1}^M \xi_j \int_0^1 \phi_j' v'\,dx = \int_0^1 fv\,dx,$$

for all $v \in V_h$. It suffices to check bvpfemsub for the basis functions $\{\phi_i\}_{i=1}^M$, which gives the $M \times M$ linear system of equations

$$\sum_{j=1}^{M} \xi_j \int_0^1 \phi_j' \phi_i' \, dx = \int_0^1 f \phi_i \, dx, \quad i = 1, ..., M,$$

for the unknown coefficients $\{\xi_j\}$. We let $\xi = (\xi_j)$ denote the vector of unknown coefficients and define the $M \times M$ *stiffness matrix* $A = (a_{ij})$ with coefficients

$$a_{ij} = \int_0^1 \phi_j' \phi_i' \, dx,$$

and the *load vector* $b = (b_i)$ with

$$b_i = \int_0^1 f \phi_i \, dx.$$

These names originate from early applications of the finite element method in structural mechanics. Using this notation, bvpfemeqn is equivalent to the linear system

$$A\xi = b.$$

In order to solve for the coefficients of U , we first have to compute the stiffness matrix A and load vector b . For the stiffness matrix, we note that a_{ij} is zero unless $i = j - 1$, $i = j$, or $i = j + 1$ because otherwise either $\phi_i(x)$ or $\phi_j(x)$ is zero on each sub-interval occurring in the integration. We illustrate this in threehat.

We compute a_{ii} first. Using the definition of the ϕ_i ,

$$\phi_i(x) = (x - x_{i-1})/h_i, x_{i-1} \le x \le x_i,$$
$$(x_{i+1} - x)/h_{i+1}, x_i \le x \le x_{i+1},$$

and $\phi_i(x) = 0$ elsewhere, the integration breaks down into two integrals:

$$a_{ii} = \int_{x_{i-1}}^{x_i} (\frac{1}{h_i})^2 \, dx + \int_{x_i}^{x_{i+1}} (\frac{-1}{h_i})^2 \, dx = \frac{1}{h_i} + \frac{1}{h_{i+1}}$$

since $\phi_i' = 1/h_i$ on (x_{i-1}, x_i) and $\phi_i' = -1/h_{i+1}$ on (x_i, x_{i+1}), and ϕ_i is zero on the rest of the sub-intervals. Similarly,

$$a_{i\,i+1} = \int_{x_i}^{x_{i+1}} \frac{-1}{h_{i+1}} \frac{1}{h_{i+1}} \, dx = -\frac{1}{h_{i+1}}.$$

Problem

Prove that $a_{i-1\,i} = -1/h_i$ for $i = 2, 3, ..., M$.

Problem

Determine the stiffness matrix A in the case of a uniform mesh with meshsize $h_i = h$ for all i.

We compute the coefficients of b in the same way to get

$$b_i = \int_{x_{i-1}}^{x_i} f(x) \frac{x - x_{i-1}}{h_i} dx + \int_{x_i}^{x_{i+1}} f(x) \frac{x_{i+1} - x}{h_{i+1}} dx, \quad i = 1, ..., M.$$

General assembly algorithm

In general the matrix A_h , representing a bilinear form

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

is given by

$$(A_h)_{ij} = a(\varphi_j, \hat{\varphi}_i).$$

and the vector b_h representing a linear form

$$L(v) = (f, v),$$

is given by

$(b_h)_i = L(\hat{\varphi}_i)$.

Computing $(A_h)_{ij}$

Note that

$$
\begin{aligned}
(A_h)_{ij} = & \quad a(\varphi_j, \hat{\varphi}_i) = \int_\Omega A(\varphi_j)\hat{\varphi}_i \; dx \\
= & \sum_{K \in \mathcal{T}} \int_K A(\varphi_j)\hat{\varphi}_i \; dx = \sum_{K \in \mathcal{T}} a(\varphi_j, \hat{\varphi}_i)_K.
\end{aligned}
$$

Iterate over all elements K and for each element K compute the contributions to all $(A_h)_{ij}$, for which φ_j and $\hat{\varphi}_i$ are supported within K .

Assembling A_h

for all elements $K \in \mathcal{T}$

for all test functions $\hat{\varphi}_i$ on K

for all trial functions φ_j on K

1. Compute $I = a(\varphi_j, \hat{\varphi}_i)_K$
2. Add I to $(A_h)_{ij}$

end

end

end

Assembling b

for all elements $K \in \mathcal{T}$

for all test functions $\hat{\varphi}_i$ on K

1. Compute $I = L(\hat{\varphi}_i)_K$

2. Add I to b_i

end

end

Mapping from a reference element - isoparametric mapping

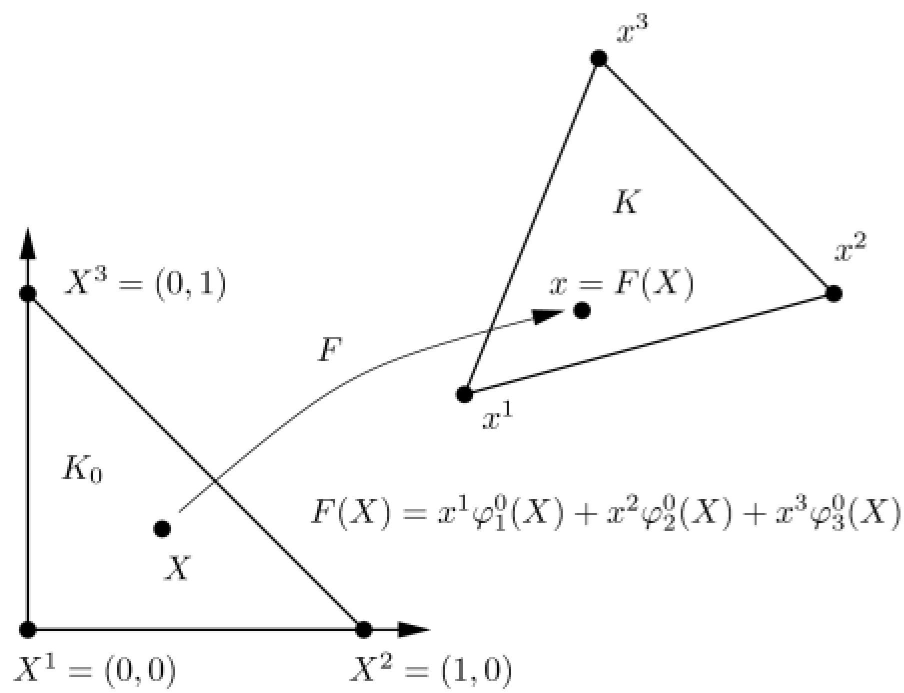
We want to compute basis functions and integrals on a reference element K_0

Most common mapping is isoparametric mapping (use the basis functions also to define the geometry):

$$x(X) = F(X) = \sum_{i=1}^{n} \phi_i(X) x_i$$

Linear basis functions $\Rightarrow$ Affine mapping: $x(X) = F(X) = BX + b$

The mapping $F : K_0 \to K$



Some basic calculus

Let $v = v(x)$ be a function defined on a domain Ω and let

$F\hat{A}\ : \Omega_0 \to \Omega$

be a (differentiable) mapping from a domain Ω_0 to Ω . We then have $x = F(X)$ and

$$\begin{aligned}\int_\Omega v(x)\ dx = & \int_{\Omega_0} v(F(X))\ ||\det \partial F_i/\partial X_j||\ dX \\ = & \int_{\Omega_0} v(F(X))\ ||\det \partial x/\partial X||\ dX.\end{aligned}$$

Affine mapping

When the mapping is affine, the determinant is constant:

$$\begin{aligned}& \int_K \varphi_j(x)\hat{\varphi}_i(x)\ dx \\ = & \int_{K_0} \varphi_j(F(X))\hat{\varphi}_i(F(X))\ ||\det \partial x/\partial X||\ dX \\ = & ||\det \partial x/\partial X|| \int_{K_0} \varphi_j^0(X)\hat{\varphi}_i^0(X)\ dX\end{aligned}$$

Transformation of derivatives

To compute derivatives, we use the transformation

$$\nabla_X = \left(\frac{\partial x}{\partial X}\right)^\top \nabla_x,$$

or

$$\nabla_x = \left(\frac{\partial x}{\partial X}\right)^{-\top} \nabla_X.$$

The stiffness matrix

For the computation of the stiffness matrix, this means that we have

$$\int_K \epsilon(x)\nabla\varphi_j(x)\cdot\nabla\hat{\varphi}_i(x)\ dx$$
$$= \int_{K_0} \epsilon_0(X)\left[(\partial x/\partial X)^{-\top}\nabla_X\varphi_j^0(X)\right]\cdot\left[(\partial x/\partial X)^{-\top}\nabla_X\hat{\varphi}_i^0(X)\right]||\det(\partial x/\partial X)||\ dX.$$

Note that we have used the short notation $\nabla = \nabla_x$.

Software

FEniCS implements the above assembly algorithm as the `assemble()` function.

Postcondition

You should now be familiar with:

- Mapping from a reference cell
- The general assembly algorithm
- How to implement boundary conditions
- How to construct the discrete system for a linear/nonlinear time-independent PDE with Galerkin's method

Exercises

CDE: 8.11, 8.12, 8.13,8.22, 8.23, 14.9

1.1

(Advanced):

Consider the non-linear equation $R(u) = -\Delta u - u^2 = 0$. Derive the weak form and go through the steps of discretization until you have a discrete (algebraic) equation. What is different from the linear case?

Examination

1.1

In Python (or a language of your choice) or with pen and paper (will probably be quickest):

Compute the integral $(\nabla\phi_0, \nabla\phi_0)$ on the reference triangle (with vertices $X_0 = (0,0), \quad X_1 = (1,0), \quad X_2 = (0,1)$, and thus $\phi_0 = 1 - x - y$). Compute the mapping $F(X)$ to a physical triangle of your choice. Compute the integral on the physical triangle using the above formula for coordinate transform.

1.2

Why is it enough to only test against the functions $\phi_i \in V_h$ and not against all functions $v \in V_h$?

[TODO]

- Add dof mapping