A Brief Introduction to the Finite Element Method

Martin Berggren

Version 1.1
October 1, 2013
Preface

These notes introduce the finite-element method for the particular case of a Poisson problem in two space dimensions with mixed Dirichlet and Neumann boundary conditions. Necessary prerequisites are only a good working knowledge in vector calculus and basic linear algebra.

Various versions of this text have been used as course material in scientific computing courses at the universities of Umeå and Uppsala, typically as a part of courses that also involve much other content. Thus, the material presented here covers just the bare bones of the finite-element method; a full course on the method would need to include much more material than provided here.

The sections are intended to be read in order. Sections marked with an asterisk are optional. These notes can freely be copied for educational purposes, provided that the source is clearly acknowledged. I welcome comments, suggestions, and information about any errors in the text.

Umeå, September 2012

Martin Berggren
Department of Computing Science
Umeå University
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Approximation of and representation of functions in FEM</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>The model problem</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>Vector calculus and Green's first identity</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>The Variational Form</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>A minimization formulation</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>Finite-element approximation</td>
<td>13</td>
</tr>
<tr>
<td>8</td>
<td>Properties of the resulting linear system</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>Accuracy</td>
<td>16</td>
</tr>
<tr>
<td>10</td>
<td>Alternative Elements</td>
<td>21</td>
</tr>
<tr>
<td>11</td>
<td>An example of a stiffness matrix on a structured mesh</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>Exercises</td>
<td>26</td>
</tr>
</tbody>
</table>
1 Introduction

Richard Courant introduced in 1943 a method to solve certain boundary-value problems for partial differential equations, a method that today is known as the finite-element method (FEM). Courant’s contribution was largely forgotten when the finite-element method reemerged and developed in the 1950s as a method to calculate elastic deformations and stresses in solids; the idea was now to model an elastic continuum, for instance a piece of metal, as an assemblage of discrete mechanical components, “finite elements”. As a simple example, an elastic membrane (such as a drum skin) can be modeled by numerous point masses interconnected with massless springs, as illustrated in figure 1.1. In the 1960–1970s researchers developed a more abstract way of understanding the method, a way of understanding that is closer to Courant’s original conception than how the method was understood by the pioneers in the 1950s. The advantage with the abstract approach, which is the one that will be presented here, is that it facilitates the use of the method as a general-purpose method for partial differential equations, regardless of the specifics of the application.

Finite-element methods have been developed for all types of partial differential equations that occur in science and engineering applications, such as the equations of solid and fluid mechanics, heat transfer, and for propagation of acoustic and electromagnetic waves. FEM is a particularly dominating technique for solving solid-mechanics problems such as estimating stresses and strains in elastic material under prescribed loads. Most CAD (Computer Aided Design) systems provide finite-element solvers in a highly integrated fashion. The engineer can typically with a few clicks on the computer screen estimate the deformations and stresses of, say, a machine part during the design.

2 Approximation of and representation of functions in FEM

Solutions to partial differential equations are “field values”. That is, they are functions defined on a domain in $d$ dimensions. (Here, we will mostly limit ourselves to $d = 1$ or 2.) The first issue we will consider is therefore how to approximate and represent functions on a domain $\Omega$, which may have a complicated shape, with a finite number of floating point numbers to be handled by the computer. The finite element approach is to use functions that are defined piecewise on a meshing of the domain $\Omega$. A meshing is a division of the domain into elementary shapes, elements. In one dimension, the elements are just intervals, whereas the mesh elements typically consist of squares or triangles in two dimensions, and cubes or tetrahedrons in three dimensions. As an example, consider the domain to the left in figure 2.1.
Subdividing the domain in triangles as to the right in figure 2.2, we obtain a meshing of the domain in the form of a triangulation. In a valid triangulation, the end points of each edge is at a vertex of the mesh. That is, a valid triangulation does not contain “hanging nodes” as illustrated in figure 2.3. How fine the mesh will be is governed by a parameter \( h > 0 \) that indicates the diameter of the largest triangle in the mesh.

The finite-element method approximates a function \( u \) defined on \( \Omega \) with a function \( u_h \) that is glued together from simple functions defined on the elements in the mesh. These simple function are usually polynomials. The easiest example is a function \( u_h \) that is continuous on \( \Omega \) and linear on each triangle in the mesh. Figure 2.4 shows an example of such a function defined on the domain and triangulation of figure 2.2. The function \( u_h \) is uniquely defined by its values at the vertices, \( x_i, i = 1, \ldots, N, \) of the mesh. (The bold face indicates a vector; in particular, \( x_i \), with Cartesian components \( x_i \) and \( y_i \), is a vector from the origin to vertex number \( i \).) The values of \( u_h \) at any point within a triangle can then simply be obtained by linear interpolation of the values at the triangle’s vertices. Thus, it is enough to store the values of \( u_h \) at the vertices in a \( N \)-vector \( u \), the vector of nodal values of the function, in order to recreate the function \( u_h \). For instance, for the function illustrated in figure 2.4, it is enough to store its vertex values in the vector

\[
\mathbf{u} = (10.5 \ 9.00 \ 7.90 \ 10.5 \ 12.7 \ 12.9 \ 12.3 \ 14.3 \ 12.9 \ 13.4 \ 14.7)^T \,. \tag{2.1}
\]

where the \( i \)th component of vector \( \mathbf{u} \) is the value of function \( u_h \) in figure 2.4 at vertex \( i \).

When recreating the function \( u_h \), the nodal values \( \mathbf{u} \) are interpolated to all points in \( \Omega \). This interpolation is conveniently defined through the use of nodal basis functions.
functions \( \phi_j, j = 1, \ldots, N \). Each nodal basis function \( \phi_j(x) \) is a finite-element function—a continuous, piecewise linear function in our case—that evaluates to either one or zero at the nodal points,

\[
\phi_j(x_i) = \begin{cases} 
1 & \text{if } j = i, \\
0 & \text{if } j \neq i,
\end{cases}
\]  

(2.2)

as illustrated in figure 2.3. The function is either one or zero at the nodes, but it is defined everywhere in \( \Omega \) and it values rises from zero to one in the triangles that surround vertex \( x_j \). Function \( u_h \) can be written as a weighted sum of the basis functions with the weights taken from vector \( u \),

\[
u_h(x) = \sum_{j=1}^{N} u_j \phi_j(x).
\]  

(2.3)

where \( u_j \) is the \( j \)th component of vector \( u \). To verify that expansion (2.3) indeed gives the expected result, we first notice that since the expression is a weighted finite sum of continuous, piecewise linear functions (the basis functions \( \phi_j \)), the final sum must also be continuous and piecewise linear. To show that \( u_h \) defined in expression (2.3) is the expected interpolation, we check the values at the vertices,

\[
u_h(x_i) = \sum_{j=1}^{N} u_j \phi_j(x_i) = \left[ \text{by (2.2)} \right] = u_i.
\]  

(2.4)

It is also illustrative to study the partial sums

\[
u_h^{(n)}(x) = \sum_{j=1}^{n} u_j \phi_j(x).
\]  

(2.5)

All partial sums for the example function in figure 2.4 are depicted in figure 2.6. The first partial sum \( u_h^{(1)} \) is just a scaled first basis function. Figure 2.6 illustrates that
$u_h^{(1)}(x) = \sum_{i=1}^{1} u_i \phi_i(x)$

$u_h^{(2)}(x) = \sum_{i=1}^{2} u_i \phi_i(x)$

$u_h^{(3)}(x) = \sum_{i=1}^{3} u_i \phi_i(x)$

$u_h^{(4)}(x) = \sum_{i=1}^{4} u_i \phi_i(x)$

$u_h^{(5)}(x) = \sum_{i=1}^{5} u_i \phi_i(x)$

$u_h^{(6)}(x) = \sum_{i=1}^{6} u_i \phi_i(x)$

$u_h^{(7)}(x) = \sum_{i=1}^{7} u_i \phi_i(x)$

$u_h^{(8)}(x) = \sum_{i=1}^{8} u_i \phi_i(x)$

$u_h^{(9)}(x) = \sum_{i=1}^{9} u_i \phi_i(x)$

$u_h^{(10)}(x) = \sum_{i=1}^{10} u_i \phi_i(x)$

$u_h^{(11)}(x) = \sum_{i=1}^{11} u_i \phi_i(x)$

Figure 2.6. Partial sums of the finite element expansion. Note that when adding the new term to the sum, the values at previous nodes do not change.
when adding an additional term to $u_h^{(n)}$ to obtain $u_h^{(n+1)}$, the values of the function at the previous nodes $x_j, i = j, ..., n + 1$ does not change.

Since we will compute numerical solutions to partial differential equations with the help of finite-element functions, it will be necessary to obtain derivatives of the functions. Since the finite-element function is a polynomial on each element, it is easy to compute derivatives inside each triangle. For the case of continuous, piecewise linear functions $u_h$, the gradient

$$\nabla u_h = \left( \frac{\partial u_h}{\partial x}, \frac{\partial u_h}{\partial y} \right)$$

(2.6)

is constant within each triangle, but these constants will change from one triangle to another. That is, $\nabla u_h$ has jumps across the edges of the triangulation. An example of gradient values is shown in figure 2.7.

Finite-element functions constructed from continuous, piecewise polynomials are differentiable everywhere except at the interfaces between elements. The gradient values displayed to the right in figure 2.7 reveal the local slopes of the function at each triangle. These values supply first-derivative information of the underlying function. However, it is not meaningful to directly calculate second derivatives. For instance, attempting to calculate second derivatives of the piecewise-linear function in figure 2.7 would produce either zeros inside the triangles (since the first derivatives are constant there) or “delta functions” at the triangle edges, since the derivatives have jumps there. These values cannot directly be used to estimate second derivatives. Unfortunately, many of the standard partial differential equations that occur in practice, for instance, the one we will consider below, contains second derivatives (the Hessian). It is possible to construct finite element functions (containing higher-order polynomials on each triangle) for which both the function and its gradient are continuous over the edges. Second derivatives can directly be computed for such functions. However, such finite elements are rather complicated to construct, and they are rarely used in practice. A better choice is to avoid explicit computation of second derivative altogether, which we will see is possible to do also for equations involving second derivatives, by exploiting a technique from vector calculus, Green’s first identity.
3 The model problem

We will introduce the finite element method for the following boundary value problem defined on a domain of the type illustrated in figure 3.8:

\[-\Delta u = f \quad \text{in } \Omega, \quad \text{(3.1a)}
\]
\[u = 0 \quad \text{on } \Gamma_D, \quad \text{(3.1b)}
\]
\[\frac{\partial u}{\partial n} = g \quad \text{on } \Gamma_N. \quad \text{(3.1c)}
\]

We require that the domain \(\Omega\) is bounded so that we can set boundary conditions at all boundaries. Boundary-value problem (3.1) is known as a Poisson problem. When \(f = 0\), the equation becomes Laplace’s equation. The Laplacian \(\Delta\) operator is the sum of second derivatives,

\[\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \quad \text{(3.2)}
\]

(Sometimes the notation \(\nabla^2\) is used instead of \(\Delta\).) The boundary \(\partial \Omega\) of the domain is divided into two parts, \(\Gamma_D\) and \(\Gamma_N\), as illustrated in figure 3.8. The boundary condition on \(\Gamma_D\) is called a homogeneous (that is, zero right-hand side) Dirichlet condition, and the condition on \(\Gamma_N\) an inhomogeneous (that is, non-zero right-hand side) Neumann condition.

Equation (3.1), or variations of it, appears in many applications. For instance, letting \(u\) represent a temperature field, equation (3.1) models steady heat conduction in a material such as a metal or plastic. This particular form of the equation requires that the heat conduction properties of the material are homogeneous (the same at each point) and isotropic (the same in each direction). The boundary conditions say that the temperature is held at zero on \(\Gamma_D\) and that the material is subject to the heat flux \(g\) through \(\Gamma_N\), which for instance can be due to an electric heating element distributed over \(\Gamma_N\). The function \(f\) models heat sources embedded in the domain.

There are conceptually two ideas that combined gives rise to a finite element discretization:

1. A reformulation of a boundary-value problem (such as (3.1)) in an integral form containing so-called “test functions.” This integral form is called the variational form (or weak form) of the boundary-value problem. The variational form combines the partial differential equation and all boundary condition in a single integral expression.

2. An example domain for model problem (3.1). The Dirichlet condition is specified on the outer boundary \(\Gamma_D\) and the Neumann condition on the inner boundary \(\Gamma_N\).
2. Approximating the exact functions in the variational form with piecewise-defined finite-element functions of the type discussed in §2.

We will see that the final outcome of the combination of these ideas when applied to problem (3.1) will be a linear system of equations

\[ Au = b \]  

(3.3)

for the vector \( u \) of nodal values of a finite-element function \( u_h \) that approximates the solution \( u \) of problem (3.1). Matrix \( A \) in equation (3.3) involves integrals of derivatives of the basis functions \( \phi_j \) associated with a triangulation of the domain, and the right-hand-side vector \( b \) involves integrals of functions \( f \) and \( g \) in problem (3.1) weighted with the finite-element basis functions.

4 Vector calculus and Green's first identity

The first step in the finite-element discretization of boundary-value problem (3.1) is to reformulate it as a variational problem. In that process, we will be able to reduce the number of derivatives occurring in the equation from two (the Laplacian operator) to one (only the gradient operator). This procedure is carried out in order to allow the use of continuous, piecewise-polynomial functions of the type introduced in §2, which cannot be differentiated twice. The basic tool for the reduction of the order of the derivatives is Green’s first identity, which is a multidimensional generalization of integration by parts.

Recall from single-variable calculus the integration-by-parts formula for smooth functions \( u, v \) defined on the unit interval,

\[
\left. v(x)u'(x) \right|_{x=0}^{x=1} = \int_0^1 v'u' \, dx + \int_0^1 vu'' \, dx.
\]  

(4.1)

The starting point for deriving formula (4.1) is the product rule of differentiation,

\[
(vu')' = v'u' + vu''.
\]  

(4.2)

Integrating both sides of expression (4.2) and using the fundamental theorem of integral calculus,

\[
\int_0^1 f' \, dx = f(x) \big|_{x=0}^{x=1},
\]  

(4.3)

with \( f = vu' \) yields formula (4.1).

Green’s first identity is derived analogously, by combining the product rule of differentiation with a higher-dimensional generalization of the fundamental theorem of integration, the divergence theorem. But before performing this derivation, we review a few definitions and formulas from vector calculus and introduce some notation.

The “vector” in vector calculus can be thought of as an arrow, or a line segment with a direction. Such a vector can either be confined to the plane, or defined in three-dimensional (Euclidian) space. We will use bold italic symbols like \( \mathbf{a} \) to denote such vectors. If \( \mathbf{a} \) is a vector in the plane, its components in the Cartesian coordinate directions will be denoted \( a_1, a_2 \). The same notation \( \mathbf{a} \) can be used also for vectors in Euclidian space with components \( a_1, a_2, a_3 \). Here, we will for simplicity only consider vectors in the plane when specifying components. However, all coordinate-free vector-analysis formulas given below also hold in Euclidian space.
The dot product between two vectors is defined as

\[ \mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^{2} a_i b_i. \] (4.4)

The differentiation operator \( \nabla \) can be thought of as the "vector operator"

\[ \nabla = \left( \frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2} \right). \] (4.5)

consisting of the partial derivatives in the coordinate directions. When applying this operator on a scalar, differentiable function \( v \) defined in the plane, we obtain the gradient vector

\[ \nabla v = \left( \frac{\partial v}{\partial x_1}, \frac{\partial v}{\partial x_2} \right). \] (4.6)

Linear combinations of the components of the gradient vector,

\[ a_1 \frac{\partial v}{\partial x_1} + a_2 \frac{\partial v}{\partial x_2}, \] (4.7)

can also be formed to obtain derivatives in other directions than the coordinate axes. If \( a = (a_1, a_2) \) is a vector of unit length \((a_1^2 + a_2^2 = 1)\), we obtain the directional derivative \( \partial v / \partial a \), that is, the derivative of \( v \) in the direction of the "arrow" \( a \) through expression

\[ \frac{\partial v}{\partial a} = a_1 \frac{\partial v}{\partial x_1} + a_2 \frac{\partial v}{\partial x_2} = \mathbf{a} \cdot \nabla v. \] (4.8)

We obtain the derivatives in the coordinate-axes directions by choosing \( a = (1, 0) \) and \((0, 1)\), respectively. If \( a = (1, 1)/\sqrt{2} \), we obtain the derivative in the direction at an angle of \(+45^\circ\) with the \( x \) axis.

If \( w \) is a differentiable vector-valued function (a vector field) defined in the plane, the dot product between operator \( \nabla \) and function \( w \) defines the divergence

\[ \nabla \cdot w = \sum_{i=1}^{2} \frac{\partial w_i}{\partial x_i} \] (4.9)

of the vector field.

Now let \( w = (w_1, w_2) \) be a differentiable vector-valued function and \( v \) a scalar-valued function. By the product rule of differentiation, it holds that

\[ \sum_{i=1}^{2} \frac{\partial}{\partial x_i} (vw_i) = \sum_{i=1}^{2} \frac{\partial v}{\partial x_i} w_i + \sum_{i=1}^{2} v \frac{\partial w_i}{\partial x_i}, \] (4.10)

or, in vector form,

\[ \nabla \cdot (vw) = \nabla v \cdot w + v \nabla \cdot w \] (4.11)

The divergence theorem (also known as Gauss' or Ostrogradsky's theorem) says that the domain integral of the divergence of a vector field \( f = (f_1, f_2) \) equals the boundary integral of the flux of the vector field,

\[ \int_{\Omega} \sum_{i=1}^{2} \frac{\partial f_i}{\partial x_i} dV = \int_{\partial \Omega} \sum_{i=1}^{2} n_i f_i \, ds, \] (4.12)
where \( n_i \) is the components of the outward-directed unit normal vector on \( \partial \Omega \). The divergence theorem allows, loosely speaking, the differentiation operator \( \partial / \partial x_i \) inside a domain integral to be replaced by the normal component \( n_i \) when the domain integral is replaced by an integral over the boundary \( \partial \Omega \). Again, using the \( \nabla \) operator and dot products, expression (4.12) may be written in the vector form

\[
\int_{\Omega} \nabla \cdot f \, dV = \int_{\partial \Omega} n \cdot f \, ds,
\]  

(4.13)

with \( n = (n_1, n_2) \). The divergence theorem (4.13) holds for bounded domains \( \Omega \) with piecewise-smooth boundaries and for continuously differentiable vector fields \( f \).

Integrating formula (4.11) and using the divergence theorem (4.13) yields

\[
\int_{\partial \Omega} w \cdot \nu \, ds = \int_{\Omega} \nabla v \cdot \nu \, dV + \int_{\Omega} \nu \cdot \nabla \, dV.
\]  

(4.14)

Now choose \( w = \nabla u \) in expression (4.14), where \( u \) is a twice continuously differentiable function. Then, by definition (4.8) of the directional derivative, and since \( \nabla \cdot \nabla = \Delta \), we obtain Green’s first identity in its standard form

\[
\int_{\partial \Omega} \nu \frac{\partial u}{\partial \nu} \, ds = \int_{\Omega} \nabla v \cdot \nabla u \, dV + \int_{\Omega} \nu \Delta u \, dV.
\]  

(4.15)

Comparing expressions (4.15) and (4.1), we see that Green’s first identity is a generalization of integration by parts to higher dimensions.

5 The Variational Form

A classical solution to the Poisson problem (3.1) is a smooth function \( u \) satisfying both the partial differential equation (3.1a) and the boundary conditions (3.1b), (3.1c).

The precise smoothness required for \( u \) to be a classical solution is that it should be at least twice continuously differentiable inside the domain, and that the function and its first derivatives should be functions that are continuous up to the boundary. For special geometries, it may be possible to use separation of variables to obtain classical solutions in the form of series solutions. Existence of classical solutions can also be proven using through potential theory, fundamental solutions, and Green’s functions. We will now present a procedure that can be used to define a more general class of solutions to problem (3.1) than classical solutions.

The first step is to introduce a another function \( v \) defined on \( \overline{\Omega} = \Omega \cup \partial \Omega \) that we will denote test function. We require that \( v \) is smooth and that it satisfies \( v(x) = 0 \) for each \( x \in \Gamma_D \). This latter requirement can be expressed in terms of a “test function rule”:

The test function should be chosen so that it vanishes at the points on the boundary where the value of the solution \( u \) is known.

The solution \( u \) is known to be zero at \( \Gamma_D \) by boundary condition (3.1b), so therefore we require

\[
v = 0 \quad \text{on } \Gamma_D.
\]  

(5.1)

If the boundary condition on \( \Gamma_D \) would have been \( u = g \) for some given, nonzero function \( g \) we would still chosen the test function to vanish at \( \Gamma_D \), according to the
above rule. The motivation for the test function rule will be given later, after completing the definition of the finite-element discretization. Except for smoothness and the conditions on $\Gamma_D$, the test function is allowed to be any function.

Assume now that function $u$ is a classical solution to the Poisson problem (3.1) and let $v$ be a test function as above. Multiply both sides of equation (3.1a) with $v$, integrate over $\Omega$, and apply Green’s identity (4.13). This yields

$$
\int_{\Omega} vf \, dV = - \int_{\Omega} v\Delta u \, dV
= - \int_{\partial\Omega} v \frac{\partial u}{\partial n} \, ds + \int_{\Omega} \nabla v \cdot \nabla u \, dV
= - \int_{\Gamma_D} v \frac{\partial u}{\partial n} \, ds - \int_{\Gamma_N} v \frac{\partial u}{\partial n} \, ds + \int_{\Omega} \nabla v \cdot \nabla u \, dV
$$

(by (5.1) and (5.1c))

$$
\int_{\Gamma_N} vg \, ds + \int_{\Omega} \nabla v \cdot \nabla u \, dV.
$$

From expression (5.2) immediately follows

**Theorem 1.** If $u$ is a classical solution to the Poisson problem (3.1), then $u$ satisfies

$$
\int_{\Omega} \nabla v \cdot \nabla u \, dV = \int_{\Gamma_N} vg \, ds + \int_{\Omega} vf \, dV,
$$

for each smooth function $v$ vanishing on $\Gamma_D$.

Equation (5.3) is called the *variational form* of the Poisson equation. We derived from the original problem (3.1) using Green’s identity and the boundary conditions on $u$ and $v$. We will now make a conceptual leap and “forget” about the origin of variational expression (5.3). We will use expression (5.3) to define a function $u$ without reference to the differential equation. For this purpose, we introduce the *energy space* as the set of all functions that vanish on $\Gamma_D$ and whose derivatives are square integrable, that is

$$
V = \left\{ v : \Omega \to \mathbb{R} \mid \int_{\Omega} |\nabla v|^2 \, dV < +\infty \text{ and } v|_{\Gamma_D} = 0 \right\},
$$

where $|\nabla v|^2$ denotes the squared length of the gradient vector, that is,

$$
|\nabla v|^2 = \left( \frac{\partial v}{\partial x_1} \right)^2 + \left( \frac{\partial v}{\partial x_2} \right)^2.
$$

The energy space is a *linear space* of functions; that is, if $v, w$ are functions in $V$, so is the linear combination $\alpha v + \beta w$ for any real numbers $\alpha, \beta$. A motivation for the requirement of square-integrable gradients in the definition of $V$ is to make sure that the integral on the left side in the variational form makes sense. The reason for the name “energy space” is that the integral of the squared gradient vector in some applications corresponds to energy of the solution; $V$ will then constitute the space of functions with bounded energy.

The variational problem, now formulated without reference to the differential equation (3.1) is the following.

Find $u \in V$ such that

$$
\int_{\Omega} \nabla v \cdot \nabla u \, dV = \int_{\Gamma_N} vg \, ds + \int_{\Omega} vf \, dV \quad \forall v \in V.
$$
The space of functions in which the solution to a variational problem is defined is called a trial space. For variational problem (5.6), the trial space is the same as the test space, the space of test functions. There are more complicated variational problems for which the trial and test spaces are different.

If \( u \) is a solution to variational problem (5.6), then \( u \) is called a weak solution of the boundary value problem (3.1). From Theorem 1 it follows that classical solutions are weak solutions. The term “weak” refers to the fact that the requirements on functions contained in the definition of \( V \) are weaker than those required of classical solutions. For instance, the second-derivative (the Laplacian operator) that occurs in the partial differential equation is absent from the variational form. However, it is possible to show that weak solutions to the Poisson equation are also classical solutions, provided functions \( f \) and \( g \) as well as boundary \( \partial \) are sufficiently smooth.

Definition (5.4) of function space \( V \) includes the condition that the functions should vanish on \( \Gamma_D \). Such a condition is called an essential boundary condition; the condition is explicitly enforced on functions in the trial and test spaces. We notice that the Neumann boundary condition on \( \Gamma_N \) has “disappeared” in variational problem (5.6); the condition is included in the variational statement itself. Such a condition is called a natural boundary condition. Thus, the Dirichlet boundary condition in the original boundary-value problem becomes an essential boundary condition in the variational form, and the Neumann condition becomes a natural boundary condition.

6 A minimization formulation

Recall that each classical solution to boundary value problem (3.1) satisfies the variational form (5.3). This variational form is all that is needed to define a finite-element discretization, as we will see in § 7. However, a classical solution to boundary-value problem (3.1) also satisfies a certain minimization problem; namely, the classical solution minimizes the quadratic form

\[
F(v) = \frac{1}{2} \int_{\Omega} |\nabla v|^2 \, dV - \int_{\Gamma_N} vg \, ds - \int_{\Omega} vf \, dV.
\]  

(6.1)

Similarly as was done for the variational problem, we can also consider the problem of minimizing \( F \) within the function space \( V \) without reference to classical solutions. That is, we may consider the problem:

Find \( u \in V \) such that

\[
F(u) \leq F(v) \quad \forall v \in V.
\]  

(6.2)

In fact, the variational problem (5.6) and the minimization problem (6.2) are equivalent:

**Theorem 2.** The function \( u \in V \) minimizes \( F \) if and only if it is a solution to variational problem (5.6)

**Remark 1.** The proof below may appear long, but is essentially no more complicated than showing, by differentiation, that the parabola \( F(x) = \frac{1}{2} x^2 - xf \) has its minimum at \( x = f \).

---

2A simple example is the case when there is a nonhomogeneous Dirichlet condition in the original boundary-value problem. This boundary condition may then be included in the definition of the trial functions. Note that the test functions also for this case will vanish on the Dirichlet boundary, in accordance with the test function rule on page 9.
Proof. For any \( u, v \in V \), and any real number \( t \), we have

\[
F(u + tv) = \frac{1}{2} \int_{\Omega} |\nabla u + t \nabla v|^2 \, dV - \int_{\Gamma_N} (u + tv)g \, ds - \int_{\Omega} (u + tv)f \, dV
\]

\[
= \frac{1}{2} \int_{\Omega} [ |\nabla u|^2 + 2t \nabla u \cdot \nabla v + t^2 |\nabla v|^2 ] \, dV
\]

\[
- \int_{\Gamma_N} (u + tv)g \, ds - \int_{\Omega} (u + tv)f \, dV
\]

\[
= F(u) + t \left( \int_{\Omega} \nabla v \cdot \nabla u \, dV - \int_{\Gamma_N} vg \, ds - \int_{\Omega} vf \, dV \right) + \frac{t^2}{2} \int_{\Omega} |\nabla v|^2 \, dV. 
\]

(6.3)

(i) Choose \( t = 1 \) and assume that \( u \in V \) is a solution to variational problem (5.6). Then expression (6.3) reduces to

\[
F(u + v) = F(u) + \frac{1}{2} \int_{\Omega} |\nabla v|^2 \, dV \geq F(u)
\]

for any \( v \in V \), which shows that \( u \) minimizes \( F \).

(ii) Now assume that \( u \in V \) minimizes \( F \). For any real number \( t \) and any \( v \in V \), we define the function \( f(t) = F(u + tv) \), that is, by perturbing \( F \) away from its minimum. Thus, the function \( f \) is minimized for \( t = 0 \). Expression (6.3) shows that \( f \) is a second-order polynomial in \( t \). The leading-term coefficient is positive for nonzero \( v \), so the polynomial has a minimum when the derivative vanishes. Setting \( f'(0) = 0 \), we conclude that

\[
\int_{\Omega} \nabla v \cdot \nabla u \, dV - \int_{\Gamma_N} vg \, ds - \int_{\Omega} vf \, dV = 0.
\]

(6.5)

for any \( v \in V \); that is, \( u \) is a solution to the variational problem (5.6).

Remark 2. The concept of variational form, as derived in the manner of § 5, is more general than minimization forms, in the sense that variational forms can be defined also for equations for which there are no corresponding minimization forms, but all minimization forms have a variational form. For instance, when the differential equation contains first-derivative (“advection”) terms, or for wave-equation-type of equations such as the Helmholtz equation, variational forms can be defined following the same procedure as in § 5. However, for these equations, it is not possible to define minimization forms.

Remark 3. In mechanics application the variational form (5.6) is called the principle of virtual work, and the minimization problem (6.2) is called the principle of minimum potential energy.

Remark 4. The terminology used here, “variational” for expression (5.6) and “minimization” for expression (6.2), is convenient for our purpose and common in finite-element contexts, but it is not the only existing. Quite commonly, it is the the minimization problem that is called “variational”! In fact, the notion of variational forms was first attached to minimizations of functionals like \( F \) in the calculus of variations.
Right: a meshing of the domain in figure 3.8. A total of 17 vertices, marked with •, are located in the interior of the domain and on $\Gamma_N$, and the finite-element functions in $V_h$ are expanded only in the basis functions that are centered at these nodes. This procedure automatically enforces the zero boundary condition on $\Gamma_D$. Left: an example function in $V_h$.

7 Finite-element approximation

We now introduce a triangulation of the domain $\Omega$ as described in § 2, and we define $V_h$ as the space of functions that are continuous over $\Omega$, linear on each triangle, and vanishing on $\Gamma_D$ (the outer Dirichlet boundary). Figure 7.9 shows an example of a mesh and a function of this kind. The mesh of figure 7.9 is very coarse. "Real-life" meshes can easily contain thousands or millions (particularly in 3D) of elements. Note that a triangular meshing of a non-polygonal domain will also involve a domain approximation: a curved boundary will be approximated with piecewise-linear segments. The space $V_h$ is a subspace of the energy space $V$ defined in (5.4) ($V_h \subset V$), since each such function satisfies the boundary condition on $\Gamma_D$, and since each gradient will be square integrable. (Recall that the gradient is piecewise constant, similarly as for the function in figure 2.7, and that the domain is bounded).

We obtain a finite-element discretization of the Poisson problem (3.1) by solving the variational problem (5.6) in $V_h$ instead of $V$:

$$\text{Find } u_h \in V_h \text{ such that } \int_{\Omega} \nabla v_h \cdot \nabla u_h \, dV = \int_{\Gamma_N} v_h g \, ds + \int_{\Omega} v_h f \, dV \quad \forall v_h \in V_h.$$  \hfill (7.1)

This type of approximation—solving a variational problem in a subspace—is called a Galerkin approximation.

In fact, we can now give a precise description of the ingredients that constitute a finite-element discretization: a finite-element discretization is a

- Galerkin approximation, based on
- piecewise-defined functions (almost always polynomial) on a meshed domain,
- applied to a
- variational form of the original problem.

Since each function in $V_h$ by definition vanishes on $\Gamma_D$, it is enough to know the function’s values at the mesh vertices in the interior of the domain and on $\Gamma_N$ (marked with solid dots to the right in figure 7.9) in order to construct the values
of function \( u_h \) at arbitrary points in the domain. The interpolation of these nodal values can be carried out using the basis function expansion described in §3:

\[
u_h(x) = \sum_{j=1}^{N} u_j \phi_j(x), \tag{7.2}\]

where \( u_j \) is the nodal values, and \( N \) here is the number of mesh vertex points in the interior of \( \Omega \) and on \( \Gamma_N \). Since the basis functions located on \( \Gamma_D \) are excluded from the sum (7.2), the boundary condition on \( \Gamma_D \) will automatically be satisfied for \( u_h \) due to the property (2.2) that basis function \( \phi_j \) vanishes at all mesh points except at vertex \( j \).

Substituting expansion (7.2) into equation (7.1) yields that

\[
\sum_{j=1}^{N} u_j \int_\Omega \nabla v_h \cdot \nabla \phi_j \, dV = \int_{\Gamma_N} v_h g \, ds + \int_{\Omega} v_h f \, dV \quad \forall v_h \in V_h. \tag{7.3}\]

(Note that the \( u_j \) is just numbers and can thus be taken out of the integral!) Since equation (7.3) should hold for each \( v_h \in V_h \), it must in particular hold for the choices \( v_h = \phi_i, i = 1, \ldots, N \), which means that

\[
\sum_{j=1}^{N} u_j \int_\Omega \nabla \phi_i \cdot \nabla \phi_j \, dV = \int_{\Gamma_N} \phi_i g \, ds + \int_{\Omega} \phi_i f \, dV \quad i = 1, \ldots, N. \tag{7.4}\]

Problem (7.4) is a system of linear equation in the coefficients \( u_j, j = 1, \ldots, N \), that is,

\[
Au = b, \tag{7.5}\]

where the matrix \( A \) has components

\[
A_{ij} = \int_\Omega \nabla \phi_i \cdot \nabla \phi_j \, dV, \tag{7.6}\]

and

\[
\mathbf{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \int_{\Gamma_N} \phi_1 g \, ds + \int_{\Omega} \phi_1 f \, dV \\ \vdots \\ \int_{\Gamma_N} \phi_N g \, ds + \int_{\Omega} \phi_N f \, dV \end{pmatrix}. \tag{7.7}\]

With a terminology borrowed from solid mechanics, the matrix \( A \) is called the stiffness matrix and the vector \( \mathbf{b} \) the load vector. This terminology is used also for cases, like heat conduction, when the PDE we are discretizing has nothing to do with mechanics! We conclude that a numerical approximation of the Poisson problem with a finite-element method involves setting up and solving the linear system (7.5).

The reason for the test function rule on page 5 can be understood from the structure of the linear system (7.5). To obtain a square linear system, we need as many equations as unknowns. The coefficient vector \( \mathbf{u} \) is of dimension \( N \), which means that matrix \( A \) has \( N \) columns. By choosing test functions associated with nodes for which the solution is unknown, we obtain \( N \) rows in the matrix and a square linear system.
Properties of the resulting linear system

The finite-element discretization of the Poisson problem \((3.1)\) carried out above results in a linear system associated with the stiffness matrix \(A\) with components

\[
A_{ij} = \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_j \, dV. \tag{8.1}
\]

This matrix has some very particular properties that will be discussed in this section. The matrix is (i) \textit{sparse}, (ii) \textit{symmetric}, (iii) \textit{positive definite}, and (iv) \textit{ill conditioned}.

All these properties (except the sparsity) reflects the nature of the boundary-value problem \((3.1)\). Some or all of these properties may thus change if the equation or the boundary conditions are altered.

(i) The \textit{sparse} property is a consequence of the choice of basis functions for \(u_h\): two “hat” functions of the kind illustrated in figure 2.5 rarely overlap, so

\[
\int_{\Omega} \nabla \phi_i \cdot \nabla \phi_j \, dV \neq 0 \tag{8.2}
\]

only when vertices \(i\) and \(j\) are nearest neighbors, that is, when they are connected with each other through an edge. The number of nearest neighbors to a vertex does not increase when the mesh is made finer, as long as the mesh refinements are made in a sensible way; see the discussion in §9. Thus, the number of nonzero elements on each row in the matrix does not increase with the order of the stiffness matrix. That is, the matrix becomes relatively sparser and sparser with increasing matrix order! Since most of the elements in the stiffness matrix will be nonzero, it is a complete waste of resources to store the whole matrix in the computer. It is standard practice in finite-element software to use a \textit{sparse format} for the matrix in order to store only the nonzero components of the matrix together with information about the matrix indices for these values.

(ii) The \textit{symmetry} of the matrix is immediate,

\[
A_{ij} = \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_j \, dV = \int_{\Omega} \nabla \phi_j \cdot \nabla \phi_i \, dV = A_{ji}. \tag{8.3}
\]

This property reflects the particular nature of the boundary-value problem \((3.1)\) and may not hold for other equations.

(iii) Recall that a real matrix \(A\) is \textit{positive definite} if \(v^T A v > 0\) whenever \(v \neq 0\).

\textbf{Theorem 3.} The stiffness matrix \((8.1)\) is positive definite.

\textit{Proof.} Let \(v_h \in V_h\). Expanding \(v_h\) in the “tent” basis functions yields

\[
v_h(x) = \sum_{i=1}^{N} v_i \phi_i(x). \tag{8.4}
\]

Setting

\[
v = (v_1, v_2, \ldots, v_N)^T \tag{8.5}
\]

yields that

\[
v^T A v = \sum_{i=1}^{N} \sum_{j=1}^{N} v_i \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_j \, dV v_j
\]

\[
= \int_{\Omega} \sum_{i=1}^{N} \nabla (v_i \phi_i) \cdot \sum_{j=1}^{N} \nabla (v_j \phi_j) \, dV = \int_{\Omega} |\nabla v_h|^2 \, dV \geq 0. \tag{8.6}
\]
with equality if and only if $\nabla v_h = 0$, that is, if $v_h$ is constant. However, since we know that $v_h$ is zero on $\Gamma_D$ (by definition of $V_h$), it follows that the constant must be zero. Thus expression (8.6) is zero only if $v_h \equiv 0$, that is, when $v = 0$.

One important consequence of Theorem 3 is that equation (7.5) has a unique solution, which follows from the fact that positive-definite matrices are nonsingular. (A linear system has a unique solution for each right-hand side if and only if the matrix is nonsingular.) A singular matrix $A$ has the property that there is a nonzero vector $v$ such that $Av = 0$ (the matrix has a nontrivial null space), which means that $v^T A v = 0$. Thus, singular matrices cannot be positive definite, and positive-definite matrices must therefore be nonsingular. Like the symmetry, the positive-definiteness of the stiffness matrix is also a consequence of the boundary-value problem (6.1) and may not hold for other equations or boundary conditions.

(iv) The condition number of the stiffness matrix depends strongly on $h$. In fact, the growth of the condition number can be estimated to $\text{cond}(A) = O(h^{-2})$ when $h$ is reduced, provided that the the quotient between $h$ and smallest inscribed circle in the triangles of the mesh is kept bounded as the mesh is refined. The exponent $-2$ in $O(h^{-2})$ follows from the highest order of the derivatives in equation (3.1a); we are considering a second-order elliptic problem.

The stiffness matrix thus has a large condition number—is ill conditioned—for fine meshes. Recall from numerical linear algebra that the bound for rounding errors in the solution of linear systems is proportional to the condition number of the matrix. Nevertheless, it turns out that rounding errors are rarely significant in practice; the condition number is typically not large enough to cause problematic amplification of round-off errors. On the other hand, the ill-conditioning is certainly an issue when applying iterative methods for solving the linear system $Au = b$. Iterative techniques will typically converge slowly when applied to linear systems of this kind. In general, the use of iterative methods require particular techniques to speed up the convergence rate, so-called preconditioning. There is also a strategy known as multigrid that exploits the fact that the matrix is ill conditioned to speed up the convergence rate! Using multigrid, large system of ill-conditioned equations can be solved in a very efficient way.

9

Accuracy

So far, we have shown that the finite-element approximation of the Poisson problem (3.1) leads to a linear system (7.5) that has a unique solution in the vector of nodal values $u$. The question how good the finite-element solution is as an approximation of the original problem will briefly be discussed in this section.

Finite-element approximations are geometric in nature, in the sense that geometric concepts such as orthogonality and projections are natural to employ in the analysis. Recall that the inner product (dot product, scalar product) of two vectors $x$, $y$ in the plane are given by

$$x \cdot y = \sum_{i=1}^{2} x_i y_i,$$  \hspace{1cm} (9.1)

and that the vectors are orthogonal (perpendicular) if $x \cdot y = 0$. The norm (length) of vectors are given by

$$|x| = (x \cdot x)^{1/2} = \left( \sum_{i=1}^{2} x_i^2 \right)^{1/2}.$$  \hspace{1cm} (9.2)
The inner product and the norm are related through the Cauchy—Schwarz inequality

$$|x \cdot y| \leq |x||y|.$$  \hspace{1cm} (9.3)

which holds for each $x$ and $y$. (Note that we here use the symbol $|$ both for absolute value and for vector norm!)

The notion of inner product and norm can be generalized to functions. For functions $u, v$ in the energy space $V$ (definition (5.4)), we define the energy inner product

$$(u, v)_V = \int_\Omega \nabla u \cdot \nabla v \, dV$$  \hspace{1cm} (9.4)

and the energy norm

$$\|u\|_V = (u, u)_V^{1/2} = \left( \int_\Omega |\nabla u|^2 \, dV \right)^{1/2}. \hspace{1cm} (9.5)$$

Recall that the length of a vector $x$ is zero ($|x| = 0$) only for the zero vector. Next theorem shows that the same property holds for the energy norm on functions in $V$.

**Theorem 4.** For $u \in V$, if $\|u\|_V = 0$ then $u = 0$.

**Proof.** Assume that $\|u\|_V = 0$. By definition (9.3),

$$\|u\|_V^2 = \int_\Omega |\nabla u|^2 \, dV = 0,$$  \hspace{1cm} (9.6)

from which it follows that $\nabla u = 0$, which means that for all $x \in \Omega, u(x) = C$ for some constant $C$. But by definition (5.4) of $V$, we know that $u = 0$ on $\Gamma_D$, from which we conclude that $u = 0$ throughout $\Omega$.

Note that the proof above shows that the requirement of a vanishing Dirichlet condition on a part of the boundary is an essential property in the definition of $V$; this property makes it possible to generate a norm from the energy inner product.

The Cauchy—Schwarz inequality also holds for the energy inner product and norm,

$$|(u, v)_V| = \left| \int_\Omega \nabla u \cdot \nabla v \, dV \right| \leq \left( \int_\Omega |\nabla u|^2 \, dV \right)^{1/2} \left( \int_\Omega |\nabla v|^2 \, dV \right)^{1/2} \hspace{1cm} (9.7)$$

Recall now that the following variational expression defines the finite-element approximation of boundary value problem (5.1),

$$\int_\Omega \nabla v_h \cdot \nabla u_h \, dV = \int_{\Gamma_N} v_h g \, ds + \int_\Omega v_h f \, dV \quad \forall v_h \in V_h.$$  \hspace{1cm} (9.8)

The original variational expression (5.6) also holds for the particular choose of test function $v_h \in V_h$ since we have required that $V_h \subset V$. That is, it holds that

$$\int_\Omega \nabla v_h \cdot \nabla u \, dV = \int_{\Gamma_N} v_h g \, ds + \int_\Omega v_h f \, dV \quad \forall v_h \in V_h.$$  \hspace{1cm} (9.9)
Subtracting expressions (9.3) and (9.8) reveals that

$$\int_{\Omega} \nabla v_h \cdot \nabla (u - u_h) \, dV = 0 \quad \forall v_h \in V_h. \quad (9.10)$$

Property (9.10) is known as *Galerkin orthogonality*, which is a fundamental *consistency relation* for Galerkin approximation. The relation says that the error $u - u_h$ is orthogonal, in the energy inner product, to each vector $v_h$ in the approximating space $V_h$. The Galerkin orthogonality property can be illustrated graphically as in figure 9.10. From the picture we see that condition (9.10) means that $u_h$ is the orthogonal projection on the space $V_h$ with respect to the energy inner product and that $u_h$ is the element in $V_h$ that minimizes $u - u_h$ in the energy norm, that is,

$$\|u - u_h\|_V \leq \|u - v_h\|_V \quad \forall v_h \in V_h. \quad (9.11)$$

This is a quite remarkable conclusion! The Galerkin approximation of variational form (5.6) has the property that it yield the *smallest possible error* in the energy norm of all possible choices of functions in the approximating space!

Above, we motivated inequality (9.11) simply by figure 9.10. However, the inequality can of course also be proven algebraically:

**Theorem 5.** Let $u \in V$ be the solution to variational problem (5.6) and $u_h \in V_h$ the solution to problem (7.1) for any $V_h \subset V$. Then

$$\|u - u_h\|_V \leq \|u - v_h\|_V \quad \forall v_h \in V_h. \quad (9.12)$$

**Proof.** Let $v_h \in V_h$ be arbitrary. Since $u_h - v_h \in V_h$, we first note that by Galerkin orthogonality (9.10),

$$\int_{\Omega} \nabla (u_h - v_h) \cdot \nabla (u - u_h) \, dV = 0. \quad (9.13)$$
By definition of the energy norm,

$$
\|u - u_h\|^2_V = \int_{\Omega} |\nabla (u - u_h)|^2 \, dV = \int_{\Omega} \nabla (u - u_h) \cdot \nabla (u - u_h) \, dV
$$

$$
= \int_{\Omega} \nabla (u - u_h) \cdot \nabla (u - u_h) \, dV + \int_{\Omega} \nabla (u - v_h) \cdot \nabla (u - u_h) \, dV
$$

$$
= \int_{\Omega} \nabla (u - v_h) \cdot \nabla (u - u_h) \, dV \leq \|u - v_h\|_V \|u - u_h\|_V, \quad (9.14)
$$

where the last inequality follows from the Cauchy—Schwarz inequality (9.7). Division with $\|u - u_h\|_V$ yields the conclusion.

The reason for the remarkable energy-norm optimality property (9.12) is that the energy inner product appears in the right-hand side of variational problem (5.6). This kind of strong optimality holds for those boundary-value problems that can be written as minimization problems (§ 6). For those boundary-value problems that can be written as variational problems but not minimization problems, the best that can be shown in general for Galerkin approximations is optimality “up to a constant”, which means that there is a constant $C$ such that

$$
\|u - u_h\|_V \leq C \|u - v_h\|_V \quad \forall v_h \in V_h. \quad (9.15)
$$

Note that we did not use the property that $V_h$ is a space of continuous, piecewise polynomials in order to prove error bound (9.13), only the fact that $V_h \subset V$. However, it would be helpful to obtain a more quantitative assessment of the error, for instance information on how the error depends on the mesh size parameter $h$. To obtain such information, we need information about the approximating properties of the finite-element functions $V_h$. Such analysis typically involves interpolants. The interpolant $u_h^I$ of a given function $u$ in the space $V_h$ is the function in $V_h$ that agrees with $u$ at the node points $x_i$ in the mesh. That is, $u_h^I$ is the function in $V_h$ such that $u_h^I(x_i) = u(x_i)$ for $i = 1, \ldots, N$, where $N$ is the number of nodes in the mesh. Figure 9.11 shows a 1D example of an interpolant of a given function onto the space of continuous, piecewise linear functions. By approximation theory it can be shown that there is a constant $C$ such that

$$
\|u - u_h^I\|_V = \left( \int_{\Omega} |\nabla u - \nabla u_h^I|^2 \, dV \right)^{1/2} \leq Ch. \quad (9.16a)
$$

$$
\|u - u_h^I\|_{L^2(\Omega)} = \left( \int_{\Omega} (u - u_h^I)^2 \, dV \right)^{1/2} \leq Ch^2. \quad (9.16b)
$$

Interpolation bound (9.16a) gives the “root-mean-square” (RMS) value of the error, usually denoted “$L^2$-error” in the finite-element context, whereas interpolation bound (9.16b) gives the energy-norm error (which also is the RMS value of the error in the gradient). We note that the $L^2$-error decreases faster to zero than the energy-norm error when $h$ is reduced. The constant $C$ involves integrals of the second derivatives of the exact solution $u$. For interpolation error bounds (9.16) to hold, assumptions have to be made on the mesh quality and the smoothness of the given function $u$:
A 1D example of an interpolant $u_h^I \in V_h$ of a given function $u$ when $V_h$ consists of continuous, piecewise linear functions.

A good way to make a uniform refinement of a triangular mesh while retaining mesh quality is to subdivide each triangle into four subtriangles.

(i) (Mesh quality.) The largest angle in any of the triangles should not approach $180^\circ$ as the mesh is refined. In practice, this requirement is accomplished by making sure that very thin triangles are avoided when generating the mesh.

(ii) (Smoothness.) Since the constant $C$ involves second derivatives of $u$, the solution needs to be smooth, otherwise the convergence rate (the exponent on $h$) will be reduced.

Since error bound (9.11) holds for any function $v_h$, we may choose $v_h = u_h^I$, where $u_h^I \in V_h$ is the interpolant of the exact solution. Provided that the exact solution is sufficiently smooth and the mesh quality is uniform with respect to $h$, expressions (9.11) and (9.16a) together yield that there is a $C$ such that

$$\|u - u_h\|_V \leq C h.$$  \hfill (9.17)

The error measured in the energy norm thus scales linearly in mesh parameter $h$. Further analysis (not provided here) shows that the error scales quadratically in the $L^2$ norm:

$$\|u - u_h\|_{L^2(\Omega)} \leq C h^2.$$  \hfill (9.18)

A good way to make mesh refinements that preserves the mesh quality is to subdivide each triangle into four new triangles, as illustrated in figure 9.12. By this uniform refinement, the mesh size parameter is cut in half, $h \rightarrow h/2$. According to error bounds (9.17) and (9.18), the energy-norm error will thus be reduced by roughly be a factor of $1/2$ and the $L^2$ error by a factor of $1/4$.

Higher accuracy can thus be obtained through refinement of the mesh. The price to pay for the smaller error associated with a finer mesh is an increase in size of the linear system (7.5) and the increased computational cost to solve it. An alternative to the uniform refinement strategy illustrated in figure 9.12 is to use adaptive refinement of the mesh ("$h$ adaptation") in regions where it is mostly needed, for instance where the solution is expected to have large gradients. In this way, the accuracy can be improved without the matrix becoming too large. Adaptive algorithms have been developed that solves a sequence of problem with finer and finer meshes. The algorithm uses the properties of the numerical solution at the coarser mesh level to decide which regions in the domain that should be refined.
A continuous, piecewise quadratic function can be constructed by interpolating nodal values at each vertex and each edge midpoint of the mesh. Higher accuracy can also be obtained by increasing the local polynomial order on the triangles. For instance, a space $V_h$ of continuous functions that is glued together from quadratic polynomials on each triangle can be defined by interpolating nodal values at each vertex and each edge of the mesh (figure 9.13). The 6 nodes that will be located in each triangle are enough to uniquely define the 6 coefficients $a_0, \ldots, a_5$ in a quadratic polynomial in two variables,

$$p(x, y) = a_0 + a_1 x + a_2 y + a_3 x y + a_4 x^2 + a_5 y^2.$$  

(9.19)

The error estimate for finite element functions that are continuous, piecewise-quadratic polynomials will be improved one order, compared with the bounds (9.17), (9.18) for piecewise linears,

$$\| u - u_h \|_V \leq C h^2,$$  

(9.20a)

$$\| u - u_h \|_{L^2(\Omega)} \leq C h^3,$$  

(9.20b)

as long as the mesh quality is retained when refining the mesh and provided that the exact solution is sufficiently smooth. (Here, constant $C$ will contain third derivatives).

So if an increased accuracy is needed, what is better, refining the mesh or increasing the local polynomial order? The rule of thumb is that increasing accuracy by using increasing the local polynomial order is particularly effective for smooth solutions, whereas if the solution is not smooth, it is better to refine the mesh. There are advanced methods ("h–p methods") that do both, use a higher polynomial order $p$ in regions in which the solution can be expected to be smooth and refines the mesh where the solution is expected to be nonsmooth.

### 10 Alternative Elements

A quadrilateral is a geometric figure obtained by connecting four different points in the plane by straight lines that do not cross. Quadrilaterals can be used to partition the domain instead of triangles, see figure 10.14. An approximating space $V_h$ build on a quadrilateral mesh can be build by globally continuous functions who, at lowest order, vary linearly along the edges of each quadrilateral. However, the functions will no longer be linear within the elements. In the special case when the quadrilaterals are rectangles oriented in the coordinate directions, the function restricted on each rectangle $K$ will be bilinear; that is, of the form

$$v_h(x, y)|_K = a + bx + cy + dxy$$  

(10.1)

on each element. The nodal values of $v_h$ (the values of $v_h$ at the four corners of the rectangle) uniquely determine the four coefficients above.
A quadrilateral mesh.

Meshes in three space dimensions can be composed of nonoverlapping tetrahedrals (left) or hexahedrals (right).

The general interpolation error bounds for continuous bilinear functions on rectangular meshes is of the same order as for continuous, piecewise-linear functions (quadratic in $L^2$ and linear in energy norm). However, rectangular elements yields a regular structure that may give high solution accuracy and allow efficient solutions of the associated linear systems. The fact that the local function (10.1) is incompletely quadratic (the quadratic term $dxy$ is present) means that derivatives will be somewhat better approximated than when using piecewise-linear functions. It is, however, hard to generate quadrilateral meshes automatically on complicated geometries in comparison to the case with triangular meshes.

For three space dimensions, triangular and quadrilateral meshes generalize to tetrahedral and hexahedral meshes (figure 10.13) with advantages and limitations as for corresponding meshes in two space dimensions.

An example of a stiffness matrix on a structured mesh

Let the domain $\Omega$ be the unit square, and consider the structured mesh of figure 11.16. We will compute the elements of the stiffness matrix

$$A_{ij} = \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_j \, dV \quad (11.1)$$

for this particular case, when the $\phi_i$s are basis functions for continuous, piecewise-linear functions. Here, there is no Neumann boundary $\Gamma_N$ (cf. figure 3.3), so the
A structured meshing of the unit square.

A basis function associated with the mesh in figure 11.16.

There is no overlap in the support for basis functions \( \phi_i \) and \( \phi_{i+2} \).

The nearest neighbors to node \( i \) are the six nodes marked with black dots. Thus, \( A_{ij} \) can be nonzero only when \( j \) corresponds to one of the black dots.

A zero Dirichlet condition covers the full boundary. There are \( J \) internal nodes in both directions, and the length of the legs of each triangle is \( h = 1/(J + 1) \). There is a total of \( J^2 = N \) internal nodes, assumed to be numbered in the row-wise direction as indicated in figure 11.16. The basis functions \( \phi_i \) have the shape indicated in figure 11.17. The support of each basis function, that is, the nonzero region of the function, is the 6 neighboring triangles that surrounds node \( i \). Note that a consequence of the small support is that most of the stiffness matrix elements are zero. For instance, \( A_{i,i+2} = 0 \) since there is no overlap in the support for the functions \( \phi_i \) and \( \phi_{i+2} \); see figure 11.18. In fact, \( A_{ij} \) can be nonzero only when \( i \) and \( j \) are associated with nearest-neighboring nodes (figure 11.19).

To calculate the stiffness-matrix elements, we need to know the gradients of the basis functions,

\[
\nabla \phi_i = \left( \frac{\partial \phi_i}{\partial x}, \frac{\partial \phi_i}{\partial y} \right).
\]

The gradient is constant at each triangle since \( \phi_i \) is composed of planar surfaces. Letting the \( x \) and \( y \) directions be oriented in the horizontal and vertical directions, respectively, the values of the gradient at the support of the basis function are indi-
The gradient of basis function \( \phi_i \) is piecewise constant on each triangle. The \( x \)- and \( y \)-coordinates are given as the pair \((.,.)\) at each triangle of the support of the function.

cated in figure 11.20. Since the basis function is oriented along the coordinate axes, the \( x \) and \( y \) derivatives and can simply be read off as the slope of the "tent" function along the legs of the triangles. With the aid of the gradients given in figure 11.20, we can compute the diagonal elements in the stiffness matrix,

\[
A_{ii} = \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_i \, dV = \sum_{k=1}^{6} \int_{T_k} \nabla \phi_i \cdot \nabla \phi_i \, dV \\
= \frac{1}{h^2} |T_1| + \frac{1}{h^2} |T_2| + \frac{1}{h^2} |T_3| + \frac{1}{h^2} |T_4| + \frac{1}{h^2} |T_5| + \frac{1}{h^2} |T_6| \\
= \frac{8}{h^2} \left( \frac{1}{2} \right) = 4,
\]

where \( |T_n| \) denotes the area of triangle \( T_n \). To compute \( A_{i,j+1} \), note that \( \nabla \phi_i \cdot \nabla \phi_{i+1} \neq 0 \) only in the two triangles visualized in figure 11.21, from which it follows that

\[
\begin{align*}
on T_1: \quad \nabla \phi_i &= \left( \frac{1}{h} \cdot \frac{1}{h} \right), & \nabla \phi_{i+1} &= \left( \frac{1}{h} \cdot 0 \right); \\
on T_2: \quad \nabla \phi_i &= \left( \frac{1}{h} \cdot 0 \right), & \nabla \phi_{i+1} &= \left( \frac{1}{h} \cdot \frac{1}{h} \right);
\end{align*}
\]

and thus

\[
A_{i,j+1} = \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_{i+1} \, dV = \sum_{k=1}^{2} \int_{T_k} \nabla \phi_i \cdot \nabla \phi_{i+1} \, dV \\
= -\frac{1}{h^2} |T_1| - \frac{1}{h^2} |T_2| = -\frac{2}{h^2} = -1.
\]

Similar calculations yield that

\[
A_{i,j-1} = A_{i,j+1} = A_{i,j-1} = -1, \quad A_{i,j+1} = A_{i,j-1} = 0.
\]
(Recall that the matrix $A$ is symmetric: $A_{ij} = A_{ji}$.) Altogether, we obtain the following block triangular structure of the stiffness matrix (empty space means zeros!)

$$A = \begin{pmatrix} T & -I & & & \\ -I & T & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & T & -I \\ & & & -I & T \end{pmatrix}.$$ \hspace{1cm} (11.7)

where $T$ and $I$ are the $J$-by-$J$ matrices

$$T = \begin{pmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{pmatrix}, \quad I = \begin{pmatrix} 1 & & & & \\ & 1 & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots \\ & & & 1 & \\ & & & & 1 \end{pmatrix}.$$ \hspace{1cm} (11.8)

Let us consider row $i$ of the matrix–vector product $Au$, obtained by multiplying from the left with vector $e_i^T$, the $i$th row of the identity matrix. We assume that this row corresponds to a vertex that is not located on the first or last row or column of internal nodes. (In the small mesh of figure 11.16 there is only one such vertex!) This row will be

$$e_i^T Au = 4u_i - u_{i+1} - u_{i-1} - u_{i+J} - u_{i-J}.$$ \hspace{1cm} (11.9)

Node $i + 1$ and $i - 1$ is located to the right and left, respectively, of node $i$, whereas nodes $i + J$ and $i - J$ are above and below node $i$. Formula (11.9) can be compared with the classical five-point finite-difference formula

$$- \Delta u(x_i) \approx \frac{4u_i - u_{i+1} - u_{i-1} - u_{i+J} - u_{i-J}}{h^2}$$ \hspace{1cm} (11.10)

for approximation of the Laplacian operator.

Thus, we conclude that the finite-element discretization of the Laplace operator using continuous, piecewise-linear functions on the structured mesh of figure 11.16 yields an equivalent discretization as when applying the five-point finite-difference stencil. This is a special case, however. If a general unstructured mesh is used, there is no simple interpretation of the finite-element discretization as a finite-difference method.
Exercises

1. For following boundary-value problems, derive weak formulations, define a FE approximation using continuous, piecewise-linear functions on a uniform grid, and specify the linear system associated with the FE approximation in terms of a mesh-size parameter $h$.

   (a) 
   $-u'' = f$  \quad \text{in} \quad (0, 1),
   u(0) = 0,
   u'(1) = 0.

   (b) 
   $-u'' = f$  \quad \text{in} \quad (0, 1),
   u(0) = g,
   u(1) = 0.

   (c) 
   $-u'' + au' = f$  \quad \text{in} \quad (0, 1),
   u(0) = u(1) = 0.

   (d) 
   $-u'' + u = f$  \quad \text{in} \quad (0, 1),
   u'(0) = u'(1) = 0. \quad (11.11)

   (e) 
   $-(c(x)u')' = f$  \quad \text{in} \quad (0, 1),
   u(0) = u(1) = 0,
   \quad \text{where} \quad c(x) > 0 \quad \text{on} \quad [0, 1].$

2. Assume that $f$ in equation (11.11) is a function in the space of approximations, that is,
   \[ f(x) = \sum_{i=1}^{I} f_i \phi_i(x), \]
   where the $\phi_i$’s are the standard “hat” functions.

   (a) Determine the mass matrix $M$ such that the linear system associated with the FE approximation of equation (11.11) can be written
   \[ Ku = Mf, \]
   where $f = (f_1, f_2, \ldots, f_I)^T$.

   (b) When computing the mass matrix, use the trapezoidal rule to evaluate the integrals involved and compare with above.

3. Give a reason why the following boundary-value problem is not well posed in general:

   $-u'' = f$  \quad \text{in} \quad (0, 1),
   u'(0) = u'(1) = 0.

   What happens if a FE discretization is applied and one tries to solve the associated linear system?
4. Calculate expressions for the element stiffness matrix

\[ A^k_{ij} = \int_{T_k} \nabla \phi_i \cdot \nabla \phi_j \, dV \]

and the element load vector

\[ f^k_i = \int_{T_k} f \phi_i \, dV \]

associated with a generic triangle \( T_h \) spanned by the corner points \( x_k, x_l, \) and \( x_m \), as in figure 1. The basis functions are the standard "tent" functions for continuous, piecewise-linear functions on a triangular mesh. For the element load vector, use the following quadrature rule:

\[ \int_{T_k} g \, dV \approx \frac{g(x_k) + g(x_l) + g(x_m)}{3} \text{ area } (T_k) \]

(trapezoidal rule).

Figure 1. A triangle spanned by the points \( x_k, x_l, \) and \( x_m \).

5. Consider a finite-element approximation of the boundary-value problem

\[ -u'' = f \quad \text{in } (0, 1), \]
\[ u(0) = u(1) = 0, \]

using continuous, piecewise-quadratic functions on a uniform grid. A nodal basis consists of values at the grid points \( x_i \) together with the midpoints \( x_{i+1/2} = (x_i + x_{i+1})/2 \).

(a) Specify and sketch the basis functions.
(b) Specify the sparsity pattern of the stiffness matrix.

6. Let \( \Omega \) be a open, bounded, and connected domain in the plane with boundary \( \partial \Omega \). Consider the boundary-value problem

\[ -\Delta u = f \quad \text{in } \Omega, \]
\[ au + \frac{\partial u}{\partial n} = ag \quad \text{on } \partial \Omega, \]

where \( g \) is a given function defined on \( \partial \Omega \) and \( a > 0 \). Derive a weak formulation of the problem and define a FE approximation. What happens when \( a \) becomes large? Can \( a = 0 \) be allowed?