Springer Series in
Computational
Mathematics

2

John R. Rice
Ronald F. Boisvert

Solving Elliptic Problems Using ELLPACK

With 53 Illustrations

Springer-Verlag
New York Berlin Heidelberg Tokyo

John R. Rice
Department of Computer Science, Purdue University
West Lafayette, IN 47907, U.S.A.

Ronald F. Boisvert
Scientific Computing Division, National Bureau of Standards
Gaithersburg, MD 20899, U.S.A.

AMS Subject Classifications: 35J99, 65M99, 65N99

Library of Congress Cataloging in Publication Data
Rice, John R.
Solving elliptic problems using ELLPACK.
(Springer series in computational mathematics ; 2)
Bibliography: p.
Includes index.
1. ELLPACK (Computer system) 2. Differential equations, Elliptic—Numerical solutions—Data processing. 3. Boundary value problems—Numerical solutions—Data processing. I. Boisvert, R. F. II. Title. III. Series.
QA377.R53 1984 515.3'53 84-10588

9 8 7 6 5 4 3 2 1

ISBN-13: 978-1-4612-9528-0 e-ISBN-13: 978-1-4612-5018-0
DOI: 10.1007/978-1-4612-5018-0

PREFACE

ELLPACK is a many faceted system for solving elliptic partial differential equations. It is a forerunner of the very high level, problem solving environments or expert systems that will become common in the next decade. While it is still far removed from the goals of the future, it is also far advanced compared to the Fortran library approach in common current use. Many people will find ELLPACK an easy way to solve simple or moderately complex elliptic problems. Others will be able to solve really hard problems by digging a little deeper into ELLPACK.

ELLPACK is a research tool for the study of numerical methods for solving elliptic problems. Its original purpose was for the evaluation and comparison of numerical software for elliptic problems. Simple examples of this use are given in Chapters 9–11. The general conclusion is that there are many ways to solve most elliptic problems, there are large differences in their efficiency and the most common ways are often less efficient, sometimes dramatically so.

ELLPACK is a research effort in the cooperative creation of a large software system. It contains contributions from over 25 people or sources. It is very large, perhaps it has a total of 80,000 lines of Fortran. A great deal of effort has been put into creating a coherent software structure that can adapt to the widely varying requirements of these people and which allows ELLPACK to grow or shrink gracefully as time and objectives change. The scientific systems programmer can examine our solution to the software engineering problems and perhaps avoid some of the mistakes we made in preliminary versions of ELLPACK.

ELLPACK owes a huge debt to its contributors. These are identified in Section 1.C and specific software contributions are associated with their authors in Chapter 6. A huge dept is also owed to the many funding agencies that supported this work. Our personal efforts were supported by the National Science Foundation, the Department of Energy, Purdue University and the National Bureau of Standards; we thank them. The total cost of developing the software in ELLPACK is of the order of $2 million. Due to the generous funding agencies, ELLPACK is made publicly available for a nominal cost.

Special thanks is due to Connie Sutherlin who prepared many versions of this manuscript, and related reports, on the TROFF system at Purdue University. Her diligence and perseverance are much appreciated. We also thank Calvin Ribbens and Roger Crawfis for valuable assistance in preparing materials for this book. Robert Lynch made many improvements in the exposition, the style and the layout of the book; we are grateful for his very substantial help.

John R. Rice
Purdue University

Ronald F. Boisvert
National Bureau of Standards

November, 1984

CONTENTS

Part 1: THE ELLPACK SYSTEM

Part 2: THE ELLPACK MODULES

Part 3: PERFORMANCE EVALUATION

Part 4: CONTRIBUTOR'S GUIDE

Part 5: SYSTEM PROGRAMMING GUIDE

APPENDICES

PART 1

CHAPTERS 1–5

THE ELLPACK SYSTEM

Part 1 is the user's guide to the ELLPACK system. It describes the structure and objectives of the system, the facilities it has and gives many examples of its application.

Chapter 1

INTRODUCTION

Elliptic partial differential equations (PDEs) are important tools for mathematical modellers in a wide variety of fields. Indeed, many important advances in structural mechanics, atmospheric modelling, nuclear reactor design, electrostatics, and chemical engineering have depended on the ability to solve elliptic equations quickly and accurately. As a result, much research activity in the past 30 years has been directed at improving numerical methods for this class of problems, and the fruits of some of this work may now be found in the growing collection of general-purpose mathematical software for solving elliptic problems [Boisvert and Sweet, 1984]. Our aim in this book is to describe one such package in detail.

ELLPACK is a software system for solving elliptic boundary value problems which includes both a very high-level problem statement language and an extensible library of problem-solving modules. The ELLPACK language is an extension to Fortran that allows users to declare boundary value problems. Powerful executable statements are available which invoke predefined modules for each stage of the solution process. Since a variety of modules representing different methods are available at each stage, ELLPACK has become a flexible tool for both experimenting with numerical methods and solving applications problems.

This chapter presents background material, outlining some key concepts and terminology associated with elliptic boundary value problems and their solution by numerical methods. The presentation here is necessarily very brief. For an elementary introduction to elliptic boundary value problems we recommend standard texts such as [Carrier and Pearson, 1976] or [Zachmanoglou and Thoe, 1976]; more advanced treatments can be found in [Courant and Hilbert, 1953, 1962] and [Mikhlin, 1967]. A number of books provide more detailed descriptions of numerical methods for these problems; for example, see [Forsythe and Wasow, 1960], [Mitchell and Wait, 1977], [Gladwell and Wait, 1979], [Lapidus and Pinder, 1982], and [Birkhoff and Lynch, 1984]. Details of the specific algorithms implemented in the ELLPACK problem-solving modules may be found in Part 2 of this book.

1.A MATHEMATICAL PRELIMINARIES

Elliptic equations describe the equilibrium states of physical systems. Well-posed problems of this type require the determination of a function which satisfies a given partial differential equation (PDE) on some domain R, as well as additional conditions along its boundary ∂R. As a simple example consider the problem of determining the steady-state temperature distribution in a nonhomogeneous

isotropic plate. We assume that the plate is thin with insulated plane surfaces so that we may consider the problem to be two-dimensional. The temperature $u(x,y)$ of the plate then satisfies

$$-(k(x,y)u_x)_x-(k(x,y)u_y)_y=0$$

in the interior of the region R defining the plate, where $k(x,y)$ is the thermal conductivity of the material. We use subscripts to denote partial differentiation; thus, the first term above represents $\partial[k(x,y)\partial u/\partial x]/\partial x$. The three most common types of boundary conditions for this problem are

(a) $$u(x,y)=r(x,y) \quad \text{on } \partial R,$$

(b) $$u_n(x,y)=r(x,y) \quad \text{on } \partial R,$$

(c) $$p(x,y)u_n(x,y)+q(x,y)u(x,y)=r(x,y) \quad \text{on } \partial R.$$

In case (a) the edges of R are held at some fixed temperature, while in case (b) the heat flux across the boundary is specified; if $r=0$ the edge is insulated. (Here, u_n denotes the derivative of u in the direction of the outward pointing normal to ∂R.) In case (c) the heat flux is a function of the temperature; this occurs when the effect of radiation of heat into the surrounding medium is being modelled. Partial differential equations and boundary conditions of this form arise in a number of different settings. See [Forsythe and Wasow, 1960] and [Birkhoff and Lynch, 1984] for a selection of applications.

A more general equation of this type is

$$Lu=(au_x)_x+(cu_y)_y+fu=g, \tag{1}$$

where a, c, f, and g are functions of x and y and the **ellipticity condition** $ac>0$ is satisfied in R. Here L denotes the differential **operator** applied to u. Equations written in this way are said to be in **self-adjoint, divergence, or conservation form**. A still more general equation is given by

$$Lu=au_{xx}+2bu_{xy}+cu_{yy}+du_x+eu_y+fu=g, \tag{2}$$

where a, b, c, d, e, f, and g are functions of x and y. The ellipticity condition $b^2-ac<0$ must hold in R for elliptic problems. These are called **second order equations** since they involve at most second order partial derivatives. They are **linear equations** since the coefficient functions a, b, c, d, e, and f and the right side g are independent of u.

Much additional terminology is associated with various special cases of these general equations. If, for instance, $a=c=1$ and $b=d=e=f=0$, we have **Poisson's equation**. If, in addition, $g=0$ the equation is called **Laplace's equation**. When $a=c=1$, $b=d=e=0$, and, $f\neq 0$, we have the **Helmholtz equation** (f constant) or the **generalized Helmholtz equation** (f variable). If Lu is the sum of two one-dimensional operators (i.e., a and d are functions of x alone, c and e are functions of y alone, $b=0$, and f is the sum of a function depending only on x and a function depending only on y), then the equation is called **separable**. Finally, equations with $g=0$ on R are called **homogeneous**. Specialized numerical methods have been developed for each of these cases.

Two types of domains on which the equation is defined are distinguished. The simplest is the **rectangular domain** $R=\{(x,y)\mid x_l<x<x_r,\ y_l<y<y_r\}$, and many numerical methods have been developed to take advantage of its special structure. In our terminology **general domain** means a domain R which is bounded

and nonrectangular; it may have holes or slits removed from its interior (along which boundary conditions must also be specified), but it must be connected; that is, any two points in the interior of R must be connectable by a polygonal line. Elliptic problems may also be posed on unbounded domains (in these cases boundary conditions "at infinity" are required), but the domains are usually truncated when posed for numerical solution.

The most common type of **boundary condition** specified in elliptic problems takes the general form

$$p(x,y)u_n + q(x,y)u = r(x,y)\,. \tag{3}$$

Again, terminology is associated with various special cases. When $p(x,y) = 0$ (solution specified) the condition is known as a **Dirichlet condition**, while $q(x,y) = 0$ (flux specified) yields a **Neumann condition**. When $pq \neq 0$ the condition is known as a third kind, elastic, or **Robbins boundary condition**. When $pq = 0$ and $p^2 + q^2 > 0$, the condition is called **uncoupled**. If $r = 0$ on ∂R then the conditions are termed **homogeneous**. Rectangular domains also admit boundary conditions of **periodic** type,

$$u(x_l,y) = u(x_r,y), \qquad y_l < y < y_r$$

and/or

$$u(x,y_l) = u(x,y_r), \qquad x_l < x < x_r$$

that is, the solution is a periodic function of x, y or both.

Certain terms are also common in describing boundary value problems as a whole (i.e., particular combinations of differential operator, domain, and/or boundary conditions). Among them are

Dirichlet problem

The boundary conditions specify the solution along the entire boundary (i.e., $p = 0$ on ∂R).

Neumann problem

The boundary conditions specify the outward-pointing normal derivative along the entire boundary (i.e., $q = 0$ on ∂R).

Separable problem

The partial differential operator is separable, the domain is rectangular, and the boundary conditions are either periodic or of the form (3) with p and q constant along each side. Separable problems may be solved very efficiently by discrete analogues of separation of variables. These are the so-called "fast direct" solution techniques.

Self-adjoint problem

The problem is based on a variational principle, that is, it arises from the minimization of an integral which describes the energy of a physical system. A problem is self-adjoint if the operator is self-adjoint and the boundary conditions are "natural" for the corresponding variational equation (see [Forsythe and Wasow, 1960, pp. 159–169]). Equation (2) is self-adjoint if and only if $d = a_x + b_y$ and $e = c_y + b_x$. Equation (1) is always self-adjoint. These problems are inherently symmetric and numerical methods often take advantage of this fact, although it is difficult for methods to do so unless the equation is written in the form (1).

In general, we must have $af \leqslant 0$ on R, $pq \geqslant 0$ on ∂R, and at least one of f and q not identically zero in order to guarantee a unique solution to the boundary value problem defined by (1) or (2) and (3). If a and f are of the same sign, then the homogeneous problem may admit eigensolutions so that the solution to the nonhomogeneous problem is nonunique. The Neumann problem with $f = 0$ has no solution unless g and r satisfy certain consistency conditions [Mikhlin, 1967, Chapter 5], and when they do the solution is unique only up to an additive constant.

A number of important elliptic equations do not take the form of (1) or (2). Among them are

Higher order equations

The simplest equation of this type is the fourth order **biharmonic equation**

$$Lu = u_{xxxx} + 2u_{xxyy} + u_{yyyy} = g,$$

which is important in the modelling of stresses in solid mechanics. Two independent boundary conditions are required to determine u in this case.

Quasilinear equations

These are the simplest types of nonlinear equation. They take the general form of (1) or (2), but have coefficients which are also functions of u, u_x, and u_y. For example, the thermal conductivity in heat conduction problems might be a function of the temperature u. Another quasilinear problem is the classical **Plateau equation**

$$(1 + u_x^2)u_{xx} - 2u_x u_y u_{xy} + (1 + u_y^2)u_{yy} = 0,$$

whose solutions model the shape of a soap film spanning a given closed loop.

Systems of equations

The biharmonic equation with boundary conditions u and $u_{xx} + u_{yy}$ given may be rewritten as the following system of two second order elliptic equations in two unknown functions

$$u_{xx} + u_{yy} = v, \qquad v_{xx} + v_{yy} = g.$$

More complex systems are the multigroup diffusion equations of nuclear reactor theory [Birkhoff and Lynch, 1984], which are linear, and the equations modelling semiconductors [Fichtner and Rose, 1981], which are quasilinear.

Three-dimensional problems

Boundary value problems involving three space dimensions are also very important if physical systems are to be realistically modelled. To obtain the most general linear equations of this type, one must add a term in u_{zz} to (1), terms in u_{zz}, u_{xz}, u_{yz}, and u_z to (2), and a term in u_z to (3). In this case all coefficients are functions of x, y, and z. They are also functions of u, u_x, u_y, and u_z in the quasilinear case.

The ELLPACK system solves single linear elliptic equations of the form (1) and (2) on general domains in two dimensions and rectangular domains in three dimensions with boundary conditions of the form (3). Periodic boundary conditions are also allowed in the rectangular case. In addition, many problem solving modules are available for special cases of these general problems.

ELLPACK also can be used to solve some quasilinear equations and elliptic systems, although no problem-solving modules solve these directly. However, various facilities are available which allow ELLPACK users to set up simple iterative procedures which use modules designed for linear problems and a single equation. These techniques are described in detail in Chapters 4 and 5.

1.B NUMERICAL METHODS PRELIMINARIES

There are two stages to the solution of elliptic boundary value problems by numerical methods. The first phase, called discretization, requires the replacement of the continuous problem by a discrete one which approximates it. In the case of equations (1) or (2) and (3) one obtains a system of linear algebraic equations. In general, the larger the system the better the approximation. The second phase is the solution of the algebraic system.

DISCRETIZATION METHODS

In the **method of finite differences** one places a rectangular **grid** $G = \{(x_i, y_j),\ i = 1, \ldots, n, j = 1, \ldots, m\}$ over the domain with the object of determining approximations to the solution at each grid point in $G' = G \cap (\mathrm{R} \cup \partial\mathrm{R})$. To do this we write an algebraic equation for each point in G' which approximates the differential equation locally. The most straightforward approach is to replace the derivatives in the differential equation by simple divided differences.

We illustrate the procedure by considering the case of rectangular R where $G' = G$. In this case the unknowns are $u(x_i, y_j)$, which are denoted by $U_{i,j}$, $i = 1, \ldots, n,\ j = 1, \ldots, m$. In the case of equally spaced grid points the finite difference equation approximating equation (2) at the point (x_i, y_j) is based upon

$$u_{xx} \approx \frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{h^2}, \qquad u_{yy} \approx \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{k^2},$$

$$u_{xy} \approx \frac{U_{i+1,j+1} - U_{i+1,j-1} - U_{i-1,j+1} + U_{i-1,j-1}}{4hk},$$

$$u_x \approx \frac{U_{i+1,j} - U_{i-1,j}}{2h}, \qquad u_y \approx \frac{U_{i,j+1} - U_{i,j-1}}{2k},$$

where h and k are the grid spacings in the x and y directions, respectively. For equation (1) we use

$$(au_x)_x \approx \frac{a_+[U_{i+1,j} - U_{i,j}] - a_-[U_{i,j} - U_{i-1,j}]}{h^2},$$

$$(cu_y)_y \approx \frac{c_+[U_{i,j+1} - U_{i,j}] - c_-[U_{i,j} - U_{i,j-1}]}{k^2},$$

where

$$a_+ = a(x_i + h/2, y_j), \qquad a_- = a(x_i - h/2, y_j),$$
$$c_+ = c(x_i, y_j + k/2), \qquad c_- = c(x_i, y_j - k/2).$$

In the case $b = 0$ these lead to finite difference equations of the form

$$\alpha_0 U_{i,j} + \alpha_1 U_{i+1,j} + \alpha_2 U_{i,j+1} + \alpha_3 U_{i-1,j} + \alpha_4 U_{i,j-1} = g$$

where, in the case of equation (2),

$$\alpha_0 = f - 2\frac{a}{h^2} - 2\frac{c}{k^2},$$

$$\alpha_1 = \frac{a}{h^2} + \frac{d}{2h}, \qquad \alpha_2 = \frac{c}{k^2} + \frac{e}{2k}, \qquad \alpha_3 = \frac{a}{h^2} - \frac{d}{2h}, \qquad \alpha_4 = \frac{c}{k^2} - \frac{e}{2k}.$$

All functions are evaluated at the point (x_i, y_j) in this case. For equation (1) we obtain

$$\alpha_0 = f - \frac{a_+ + a_-}{h^2} - \frac{c_+ + c_-}{k^2},$$

$$\alpha_1 = \frac{a_+}{h^2}, \qquad \alpha_2 = \frac{c_+}{k^2}, \qquad \alpha_3 = \frac{a_-}{h^2} \qquad \alpha_4 = \frac{c_-}{k^2}.$$

Boundary conditions must also be incorporated into the discretization, and Dirichlet and periodic conditions are straightforward to handle. At a point on ∂R where equation (3) with $p \neq 0$ must be satisfied, the simplest approach is to write a finite difference formula for (1) or (2) at this point and then use a finite difference approximation to (3) to eliminate the unknown introduced outside the domain. For example, in the case of a boundary point along the right hand edge (not a corner) one obtains for equation (2)

$$\alpha_0 = f - 2\frac{a}{h^2} - 2\frac{c}{k^2} - \frac{2hq}{p}\left(\frac{a}{h^2} + \frac{d}{2h}\right),$$

$$\alpha_1 = 0, \qquad \alpha_2 = \frac{c}{k^2} + \frac{e}{2k}, \qquad \alpha_3 = \frac{2a}{h^2}, \qquad \alpha_4 = \frac{c}{k^2} - \frac{e}{2k}.$$

where the right side of the equation has also been changed to $-2hr[a/h^2 + d/(2h)]/p$. In the case of equation (1) the corresponding approximation results in

$$\alpha_0 = f - \frac{a_+ + a_-}{h^2} - \frac{c_+ + c_-}{k^2} - \frac{2hq}{p}\left(\frac{a_+}{h^2}\right),$$

$$\alpha_1 = 0, \qquad \alpha_2 = \frac{c_+}{k^2}, \qquad \alpha_3 = \frac{a_+ + a_-}{h^2}, \qquad \alpha_4 = \frac{c_-}{k^2},$$

where the right side is also modified to $g - 2hr[a_+/h^2]/p$.

The finite difference equation at a particular grid point involves at most five unknowns when $b = 0$ (nine otherwise), and hence the formula is known as the **5-point star**. For smooth u and a uniform grid, the error in making these approximations is $O(h^2 + k^2)$, that is, the error goes to zero as fast as $h^2 + k^2$. Thus we say that the difference formulas are **second order accurate**. The difference equations become only slightly more complex for unequally-spaced meshes, and they may easily be generalized to three dimensions (this yields the **7-point star** discretization). Finite difference discretizations with order of accuracy greater than two may be obtained by using the **HODIE method** [Lynch and Rice, 1979] or the techniques of **extrapolation** and **deferred corrections** [Pereyra, 1966].

For general domains the chief source of difficulty is the incorporation of boundary conditions. The need to approximate u_n when the outward-pointing normal vector does not correspond to a grid line leads to one-sided approximations that increase the matrix bandwidth. In fact, unless the domain has some very special shape, symmetry will also be destroyed, even for Dirichlet boundary conditions.

The **finite element method** employs an alternative discretization technique. Instead of obtaining values of the solution at a set of points, one selects a set of basis functions $\{\phi_j(x,y), j = 1, \ldots, N\}$ and then determines coefficients $c_j, j = 1, \ldots, N$ so that the function

$$U(x,y) = \sum_{j=1}^{N} c_j \phi_j(x,y) \tag{4}$$

approximates the solution u as well as possible. (For simplicity we will assume that the boundary conditions imposed on the solution to the differential equation are homogeneous, and that the basis functions ϕ_j satisfy them exactly.) In particular, the coefficients c_j are determined by the requirement that

$$(\mathrm{L}U - g, t_i) = 0, \qquad i = 1, \ldots, N, \tag{5}$$

where $(\cdot,\cdot)$ is the inner product defined by $(f,g) = \int_{\mathrm{R}} fg$ and $\{t_i(x,y), i = 1, \ldots, N\}$ is a given set of test functions. This is a linear system of algebraic equations which is given more explicitly by

$$\sum_{j=1}^{N} c_j(\mathrm{L}\phi_j, t_i) = (g, t_i), \qquad i = 1, \ldots, N. \tag{6}$$

A particular finite element method is obtained by specifying the basis functions and test functions. Intuitively, the ϕ_j are chosen for their ability to approximate u, while the t_i are chosen for their ability to approximate the **residual** $\mathrm{L}U - g$. (The latter observation comes from the fact that (5) requires the residual to be orthogonal to the space spanned by the test functions. If the residual is represented well in this space, then its components orthogonal to it will be small.) In any case, we seek choices of $\{\phi_j\}$ and $\{t_i\}$ so that U can be made arbitrarily close to u by taking N large enough.

The **Galerkin method** or **Rayleigh-Ritz method** uses the choice $t_i = \phi_i$, $i = 1, \ldots, N$. It can be shown that with this choice the matrix in (6) is symmetric whenever the operator L is self-adjoint. The integrations in (6) are often done numerically. If Dirac delta functions are used for the t_i, then the method is called **collocation**. Here we choose a set of N points $p_i = (x_i, y_i)$ in R and define the test functions as $t_i(p) = \delta(p - p_i)$. Putting these functions in (5) yields the requirement that the differential equation be satisfied exactly at the collocation points, that is

$$\mathrm{L}U(x_i, y_i) = g(x_i, y_i) \qquad i = 1, \ldots, N.$$

A very important feature of finite element methods is the use of **piecewise polynomial basis functions**. To define these we first partition the domain into a set of subdomains called **elements**; rectangles or triangles are the most common choices in two dimensions. A piecewise polynomial defined on this partition is then a function which is a polynomial (usually of low degree) on each element separately. A particular space of piecewise polynomials is selected by the choice of the polynomial degree for each element and the degree of continuity across element boundaries. This gives a finite dimensional linear space, and hence there is a set of basis functions ϕ_j such that any piecewise polynomial in the space may be written in the form (4).

In practice one chooses **basis functions with small support**, that is, basis functions that are identically zero except for a small number of elements. If this is

so, the matrix A in (6) becomes very sparse. For Galerkin's method, the matrix entry $A_{ij} = (L\phi_j, \phi_i)$, is nonzero only when the supports of ϕ_i and ϕ_j overlap. For example, consider the **bilinear basis function** defined on a rectangular grid by $\phi(x,y) = \psi_i(x)\psi_j(y)$, where

$$\psi_k(z) = \begin{cases} \dfrac{z - z_{k-1}}{z_k - z_{k-1}}, & z_{k-1} \leqslant z \leqslant z_k, \\ \dfrac{z_{k+1} - z}{z_{k+1} - z_k}, & z_k \leqslant z \leqslant z_{k+1}, \\ 0, & \text{elsewhere}. \end{cases}$$

This element is one at (x_i, y_j) and zero outside the rectangle $x_{i-1} < x < x_{i+1}$, $y_{j-1} < y < y_{j+1}$. It is continuous everywhere, but not differentiable along the lines $x = x_{i-1}$, x_i, x_{i+1} and $y = y_{j-1}$, y_j, y_{j+1}. (Note that we often may apply Green's theorem (integration by parts) to the integral in (5), thus allowing the use of basis functions with less continuity than is required by the differential operator.) We may associate one such basis function with each grid point and hence there are nine nonzeros per row in the resulting matrix.

In collocation, $A_{i,j}$ is nonzero only when the i-th collocation point is in the support of the j-th basis function. For the same set of basis functions, collocation yields fewer non-zeros per equation than for Galerkin's method and computing them is easier since there are no integrals to do. Due to its simplicity, the class of problems to which collocation is easily applied is greater than for the Galerkin method. However, collocation usually does not give a symmetric matrix even in the self-adjoint case. In addition, one must use basis functions of degree at least two in order for $L\phi_j$ to be nonzero.

Solution Methods

The discretization of an elliptic problem produces a (usually) large sparse system of linear algebraic equations which we denote as $A\mathbf{u} = \mathbf{g}$. We now use lower case bold letters to conform to the usual matrix-vector notation. Methods for solving this system can be classified as either direct or iterative (although there are now various hybrid methods). **Direct methods** produce the exact answer in a finite number of steps (in the absence of rounding error), whereas **iterative methods** produce a sequence of approximations which converge to the solution u only in the limit.

The structure of the matrix A depends upon how we number the unknowns and the order in which we write the equations. In finite differences, both equations and unknowns correspond to grid points, and hence numbering the grid points is equivalent to numbering the equations and unknowns in the same way. In the so-called **natural ordering** we number points in G' from left to right and bottom to top. Similar orderings exist in the finite element case, where numbering the equations and unknowns corresponds to numbering the test functions and basis functions. These lead to matrices with a number of desirable properties:

Sparsity

Most entries in the matrix are zero. In the finite difference discretization above, we have at most nine nonzero coefficients per equation (five when $b = 0$), independent of the number of grid points.