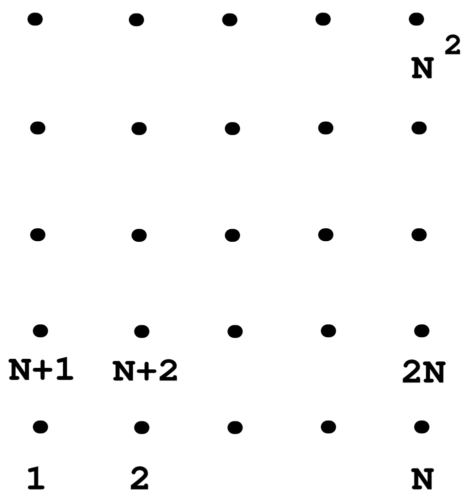


Figure 3.7: Natural ordering for the interior grid points.

Since we have already discretized the left hand side of the equation in 3.19, we simply need to add the right hand side,

$$\nabla^2 u_{i,j} = \frac{1}{h^2}[u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}] = f(x_i, y_j) \tag{3.31}$$

Such that our solution can be written as,

$$-u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} + 4u_{i,j} = -h^2 f(x_i, y_j), \tag{3.32}$$

where we have multiplied through by minus one. As an illustration of the kind of system this produces, we order the points in *natural* or *row wise* ordering and look at the system for $N = 2$.

$$\begin{pmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{pmatrix} \begin{pmatrix} u_{11} \\ u_{21} \\ u_{12} \\ u_{22} \end{pmatrix} = -h^2 \begin{pmatrix} f_{11} \\ f_{21} \\ f_{12} \\ f_{22} \end{pmatrix} + \begin{pmatrix} u_{01} + u_{10} \\ u_{20} + u_{31} \\ u_{02} + u_{13} \\ u_{32} + u_{23} \end{pmatrix} \tag{3.33}$$

Where, for the Dirichlet problem, the boundary data is given as:

$$\begin{aligned} u_{0,j} = g(0, y_j), \quad u_{N+1,j} = g(1, y_j), \quad j = 0, 1, \ldots N+1 \\ u_{i,0} = g(x_i, 0), \quad u_{i,N+1} = g(x_i, 1), \quad j = 0, 1, \ldots N+1 \end{aligned} \tag{3.34}$$

We can then generalize the Poisson problem with Dirichlet boundary conditions as the $N^2 \times N^2$ linear system,

$$\begin{pmatrix} T_N & -I_N & & \\ -I_N & \ddots & \ddots & \\ & \ddots & & -I_N \\ & & -I_N & T_N \end{pmatrix} \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_N \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 - h^2\mathbf{f}_1 \\ \mathbf{b}_2 - h^2\mathbf{f}_2 \\ \vdots \\ \mathbf{b}_N - h^2\mathbf{f}_N \end{pmatrix}. \tag{3.35}$$

where

$$\mathbf{u}_i = (u_{1i}, \ldots, u_{Ni})^T, \quad \mathbf{f}_i = (f_{1i}, \ldots, f_{Ni})^T, \quad i = 1, \ldots, N \tag{3.36}$$

$$\mathbf{b}_1 = (u_{01} + u_{10}, u_{20}, \ldots, u_{N-1,0}, u_{N,0} + u_{N+1,1})^T \tag{3.37}$$

$$\mathbf{b}_i = (u_{0i}, 0, \ldots, 0, u_{N+1,1})^T, \quad i = 2, \ldots, N-1 \tag{3.38}$$

$$\mathbf{b}_N = (u_{0,N} + u_{1,N+1}, u_{2,N+1}, \ldots, u_{N-1,N}, u_{N,N+1} + u_{N+1,N})^T \tag{3.39}$$

3.3.3 Poisson's Equation - A Numerical Example

To illustrate the finite difference method as applied to Poisson's equation, consider same steady-state heat conduction problem defined over our unit square as in our last example. This time we find the finite difference approximation to Poisson's equation subject to the Dirichlet boundary conditions,

$$u(x,0) = u(1,y) = u(x,1) = u(0,y) = 0 \tag{3.40}$$

with the additional source term $f(x,y) = 1$ at the center of the square.

We again discretize our square into a uniform mesh with $h = 0.25$ as shown in Figure 3.3.1. This time, we can the finite difference memthod using equation 3.32.

Upon solving the resulting linear system, we obtain a vector $\mathbf{u}$ of the temperatures. Again we look at the distributions of temperature as a function of the number of elements (and thus step-size, h).

Again we notice that as the number of elements increase (and the step-size decreases), the solution becomes smoother - and our computational costs increase.

3.3.4 Neumann Boundary Conditions

Now let's consider another type of boundary conditiont that occurs frequently in engineering field problems, the Neumann problem. The Neumann boundary condition, which is often denoted as the *natural* boundary condition (as opposed to the Dirichlet boundary condition which is often termed the *essential* boundary condition) is concerned with the *flux* (i.e. the normal of the gradient) at the boundary. The so-called, Neumann problem (when the Neumann boundary condition is applied over the entire boundary) requires the solution of $u(x,y)$ over $\Omega \subset \partial\Omega$ subject to,

$$\nabla u \cdot \mathbf{n} = \frac{\partial u}{\partial n} = f(x,y) \quad \text{on } \partial\Omega \tag{3.41}$$

For many physical situations, $f(x,y)$ is equal to zero. This occurs when there is a so-called, no-flux boundary condition, such that the solution domain is somehow isolated and

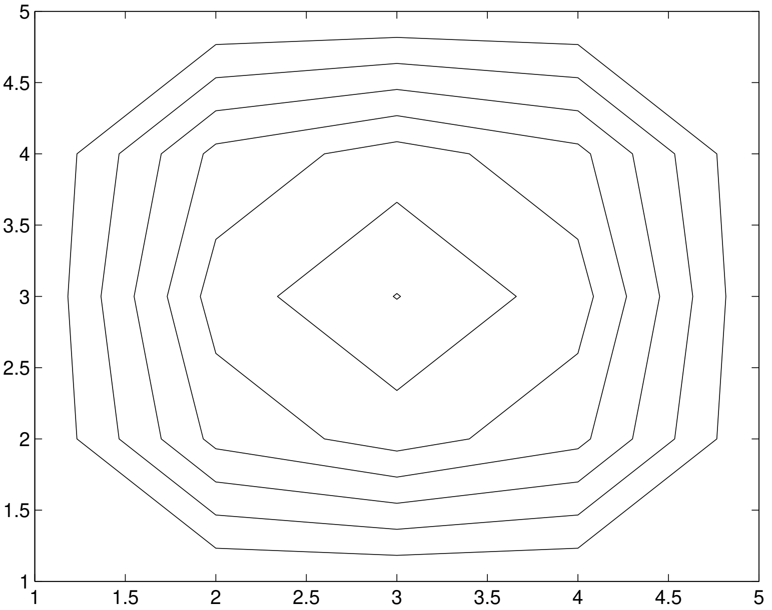


Figure 3.8: Temperature distribution, shown as iso-temperature contours for the solution of Poisson's equation. Here we have divided up the solution domain into 8 panels with $h = 0.25$.

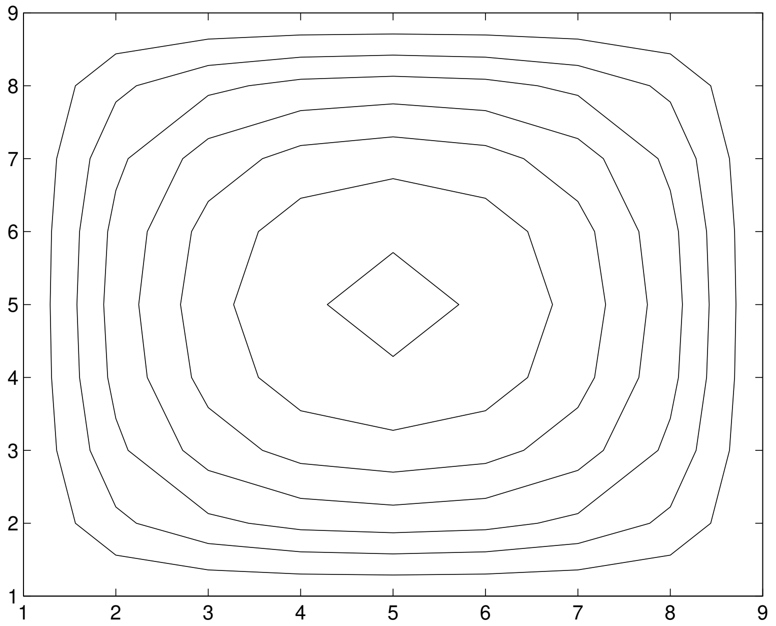


Figure 3.9: Temperature distribution, shown as iso-temperature contours for the solution of Poisson's equation. Here we have divided up the solution domain into 16 panels with $h = 0.125$.

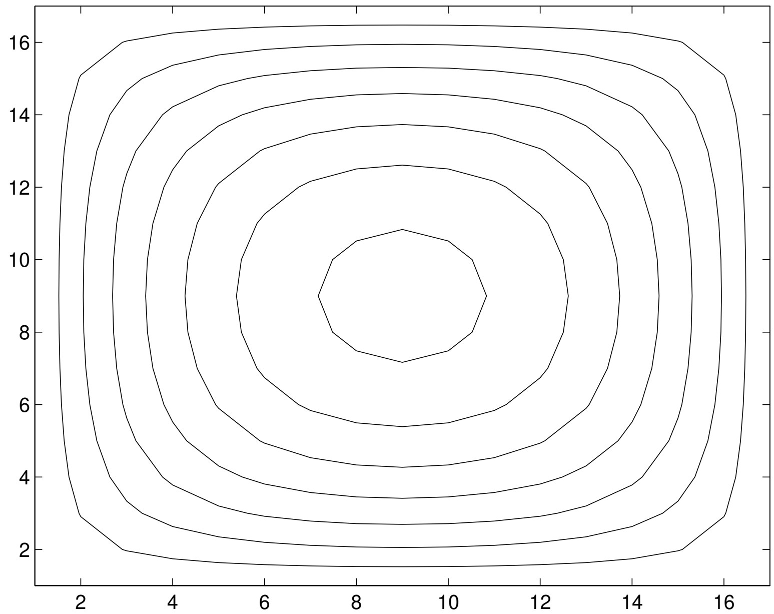
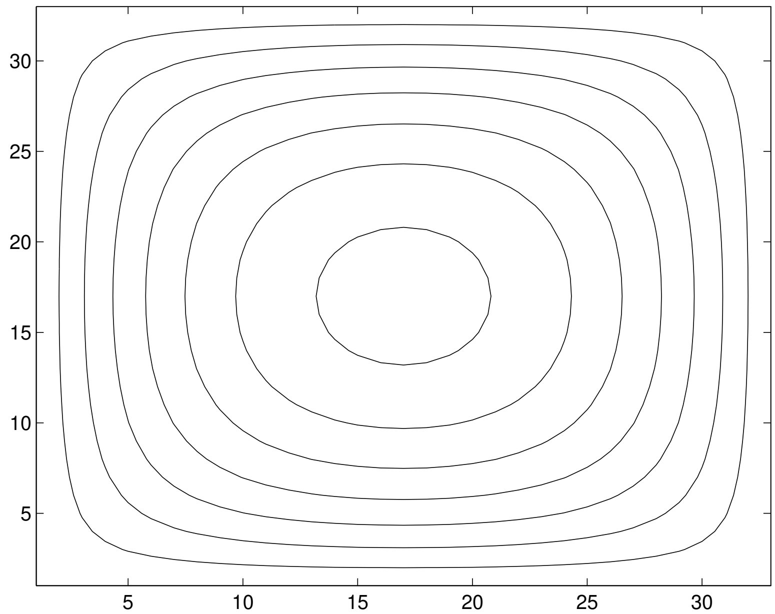


Figure 3.10: Temperature distribution, shown as iso-temperature contours for the solution of Poisson's equation. Here we have divided up the solution domain into 32 panels with $h = 0.0625$.



[

Figure 3.11: Temperature distribution, shown as iso-temperature contours for the solution of Poisson's equation. Here we have divided up the solution domain into 64 panels with $h = 0.03125$.

the field is not allowed to flow outside the domain. A simple example is in electrostatics when the solution domain is surrounded by a non-conducting media (like air) such that the current (flux) cannot flow outside the conducting solution domain. While this is usually the case, it is not always the case, therefore let's look at the general case. To illustrate how to incorporate Neumann boundary conditions into our finite difference formulation, let's consider our example of solving a heat conduction problem on the unit square [22].

3.3.5 Poisson's Equation with Neumann Boundary Data - A Numerical Example

Consider our unit square as a steel plate which is uniformly generating heat at each point a t a rate of $Q = 5cal/cm^3 \cdot sec.$ The thermal conductivity of the plate is $k = 0.16\mathrm{cal/sec} \cdot \mathrm{cm}^2 \cdot \mathrm{C/cm}$. For this problem, Poisson's equation holds

$$\nabla^2 u = -\frac{Q}{k} \tag{3.42}$$

Assume that the top and bottom faces are perfectly insulated so that no heat is lost, but the upper and lower edges lose heat according to

$$\frac{\partial u}{\partial y} = -15\,\mathrm{C/cm} \tag{3.43}$$

in the outward direction as shown in Figure 3.3.5. Furthermore, the right and left edges are held at a constant tempartuure $u = 20$C. Our goal is to find the steady-state tempartures within the area of the plate. In this problem, the upper and lower edge temperatures are also unknown, which increases the number of unknowns we need to solve for. Because of the symmetry, we can limit ourselves to a system of 6 values. We now apply the five-point difference formula to the system. We notice that at the upper and lower edges, we cannot form the five-point combination as there are not enough points outside the boundary. We get around this by extending our mesh to include an additional row (upper and lower) of exterior points. This gives us the following set of equations:

$$\begin{aligned}
(20 + u_a + u_2 + u_3 - 4u_1) &= -\frac{(.25)^2 5.0}{0.16}, \\
(u_1 + u_b + u_1 + u_4 - 4u_2) &= -\frac{(.25)^2 5.0}{0.16}, \\
(20 + u_1 + u_4 + u_5 - 4u_3) &= -\frac{(.25)^2 5.0}{0.16}, \\
(u_3 + u_2 + u_3 + u_6 - 4u_4) &= -\frac{(.25)^2 5.0}{0.16}, \\
(20 + u_3 + u_6 + u_3 - 4u_5) &= -\frac{(.25)^2 5.0}{0.16},
\end{aligned}$$

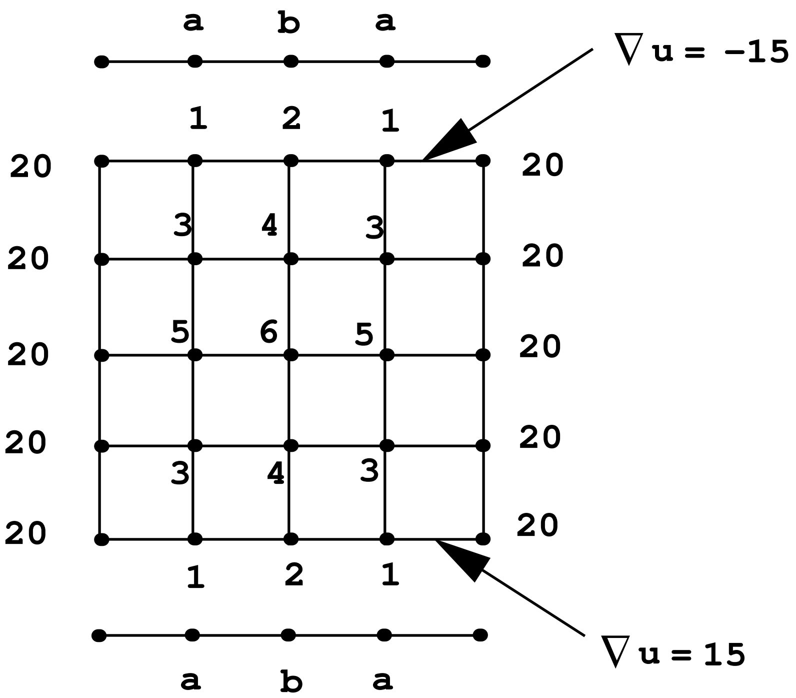


Figure 3.12: Set-up for the the steady-state heat equation with Neuman boundary conditions.

$$(u_5 + u_4 + u_5 + u_4 - 4u_6) = -\frac{(.25)^2 5.0}{0.16}, \tag{3.44}$$
$$\tag{3.45}$$

Now we apply the Neuman boundary condition utilizing the difference formula for the first partial derivative. Note that we make sure to choose the order of the points so that it reflects that the outward normal is negative, since heat is flowing outwardly.

$$\begin{aligned} -(\frac{\partial u}{\partial y})_1 &= \frac{u_3 - u_a}{2(.25)} = 15, \quad u_a = u_3 - 7.5, \\ -(\frac{\partial u}{\partial y})_2 &= \frac{u_4 - u_b}{2(.25)} = 15, \quad u_b = u_4 - 7.5, \end{aligned} \tag{3.46}$$

We now eliminate the exterior rows of points we added and then solve our system of equations.

Now let's generalize things a bit, the Neumann conditions for Laplace's equation on the unit square can be summarized as,

$$-\frac{\partial u}{\partial x} = f(0, y) \quad \frac{\partial u}{\partial x} = f(1, y) \tag{3.47}$$

$$-\frac{\partial u}{\partial y} = f(x, 0) \quad \frac{\partial u}{\partial y} = f(x, 1) \tag{3.48}$$

along the x and y axis respectively. The minus signs denote that the gradients always are pointing outward, away from the boundary. We now apply a finite grid of points to the square (as we did before). Then we apply the five point formula to all of the $(N+2)^2$ grid points within the domain Ω. When we apply our five point formula to the points on $\partial\Omega$ we note that it involes the set of points: $(ih,-h),(ih,1+h),(-h,jh),(1+h,jh)$, where $i,j=0,1,\ldots N+1$, which lie outside the boundary. We use the derivative boundary conditions to "eliminate" these points outside of the boundary. Recalling the Taylor series approximation for first order partial derivatives,

$$\frac{\partial u}{\partial x} = \frac{[u(x+h,y)-u(x-h,y)]}{2h} + O(h^2) \tag{3.49}$$

$$\frac{\partial u}{\partial y} = \frac{[u(x,y+h)-u(x,y-h)]}{2h} + O(h^2) \tag{3.50}$$

we use these as replacements for the exterior points in the five point formula,

$$u_{-1,j} = u_{1,j} + 2hf(0,jh) + O(h^3) \tag{3.51}$$

$$u_{N+2,j} = u_{N,j} + 2hf(1,jh) + O(h^3) \tag{3.52}$$

$$u_{i,-1} = u_{i,1} + 2hf(ih,0) + O(h^3) \tag{3.53}$$

$$u_{i,N+2} = u_{i,N} + 2hf(ih,1) + O(h^3) \tag{3.54}$$

where $i,j=0,1,\ldots,N+1$. These may be used in the five point formula along $x=0, x=1, y=0, y=1$ respectively.

The $(N+2)^2$ equations which arise from the Neumann problem then can be written as,

$$A\mathbf{U} = 2h\mathbf{f} \tag{3.55}$$

where the matrix A now has the form,

$$A = \begin{pmatrix} T_N & 2I_N & & & & \\ I_N & T_N & I_N & & & \\ & \ddots & \ddots & \ddots & & \\ & & & & & \\ & & & I_N & T_N & I_N \\ & & & & 2I_N & T_N \end{pmatrix} \tag{3.56}$$

Here, I is the identity matrix of order $N+2$ and now T_N is the tridiagonal matrix,

$$T_N = \begin{pmatrix} -4 & 2 & & & & \\ 1 & -4 & 1 & & & \\ & 1 & \ddots & \ddots & & \\ & & \ddots & & & \\ & & & & & \\ & & & 1 & -4 & 1 \\ & & & & 2 & -4 \end{pmatrix} \tag{3.57}$$

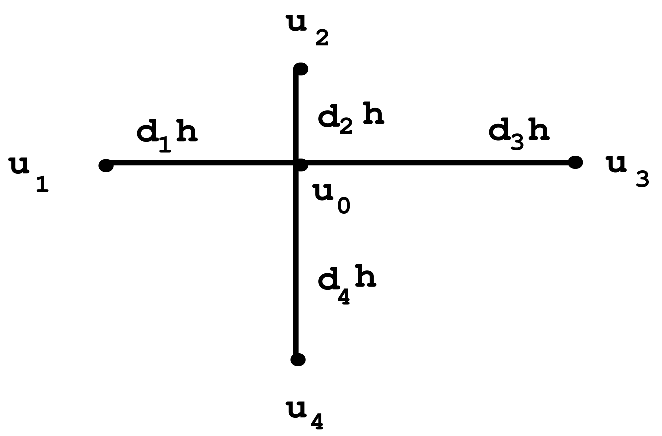


Figure 3.13: Irregular spacing for the five-point finite difference formulation.

thus the solution to the general Neumann problem is,

$$\mathbf{u} = 2hA^{-1}\mathbf{f}. \tag{3.58}$$

One can prove that the matrix A is singular, and thus, does not have a unique solution. Usually the field problems that we encounter in engineering and science have mixed boundary conditions, i.e. portions of the surface have Neumann boundary conditions, while other portions have Dirichlet boundary conditions, thus rendering a nonsingular A matrix (and unique solution).

3.3.6 Irregular Regions

So far we've treated solution domains in which the boundary coincided with nodes from our mesh. For many problems, we will have irregular domains and thus must re-think our approach. One approach to this problem is to continually subdivide our mesh into smaller and smaller grid points such that the uniform mesh can approximate the irregular boundary. While this will usually work, this adds a tremendous amount of additional computation since we must usually add a large number of additional degrees of freedom. Another approach is to redefine our difference formulas to take into account the irregular region. Still other approaches use domain decomposition approaches with meshes of different levels of refinement, interpolation methods, and/or combinations of all the above. Let's explore the option of modifying the difference formulas to account for the irregular boundary.

Consider the case of using the five-point formula but for unequal lengths between nodes as shown in Figure 3.3.6. Along the line from u_1 to u_0 to u_3, we have the following approximations for the first derivatives:

$$(\frac{\partial u}{\partial x})_{1-0} = \frac{u_0 - u_1}{d_1 h} \tag{3.59}$$

$$(\frac{\partial u}{\partial x})_{0-3} = \frac{u_3 - u_0}{d_3 h} \tag{3.60}$$

Since

$$(\frac{\partial^2 u}{\partial x^2}) = \frac{\partial}{\partial x}(\frac{\partial u}{\partial x}) \tag{3.61}$$

we have

$$(\frac{\partial^2 u}{\partial x^2}) = \frac{(u_3 - u_0)/d_3 h - (u_0 - u_1)/d_1 h}{(d_1 + d_3)h/2}. \tag{3.62}$$

A similar expression exists for y. Combining and rearranging, we obtain

$$\nabla^2 u = \frac{2}{h^2}[\frac{u_1}{d_1(d_1+d_3)} + \frac{u_2}{d_2(d_2+d_4)} + \frac{u_3}{d_3(d_1+d_3)} + \frac{u_4}{d_4(d_2+d_4)} - (\frac{1}{d_1 d_3} + \frac{1}{d_2 d_4})u_0] \tag{3.63}$$

We then use this new operator for points adjacent to the boundary points when the boundary points do not coincide with the mesh, instead of our standard five-point rule. If there are Neuman boundary conditions on the curved part of the boundary, then we must go back and re-derive the approximation for the Neuman condition (this gets rather invovled) [21].

3.3.7 Numerical Example: Curved Boundary

Consider the following example of solving the steady-state heat flow equation on a semi-circular plate of radius 1. The temperature at the base is 0 and the temperature on the circumference is 100. The analytic solution is

$$u(r,\theta) = \frac{(4)(100)}{\pi} \sum_{n=1}^{\infty} \frac{1}{2n-1}(\frac{r}{(1)})^{2n-1} \sin(2n-1)\theta \tag{3.64}$$

We now impose a uniform mesh on the semicircular region as shown in Figure 3.3.7. Notice that the mesh points, $u_2, u_3, u_{12}, \ldots$ don't coincide with the boundary. What we do is use the geometry to figure out what the values for $d_2, d_3, \ldots$. It turns out that the short part for $u_2 = .8990h$ and for $u_3 = .5826h$. We continue to apply this method for the rest of the points using our new non-uniform operator. For this particular problem, the finite difference solution is in very good agreement with the analytic solution. As you can imagine, this method would not be very efficient for domains with many different curved regions. A more reasonible way to approach the problem would be to simply use the mesh point which is closest to the point on the boundary - in effect, perturbing the actual region. Obviously one introduces errors by using this method, however, if the distance between the mesh point and boundary is not too large, then the result can be satisfactory.

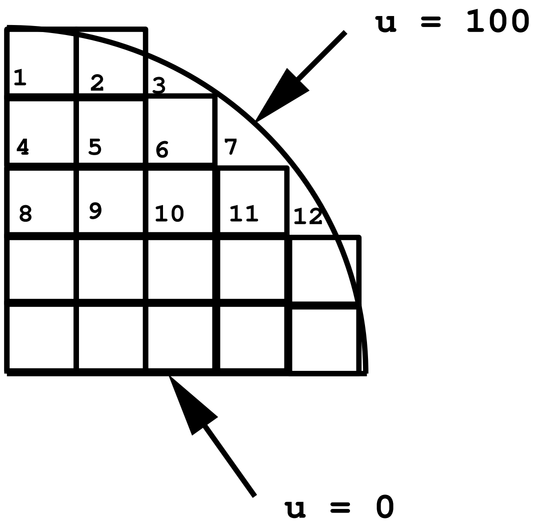


Figure 3.14: Example of a curved boundary.

3.3.8 The Laplacian Operator in Three Dimensions

We can extend the finite difference formulation to three-dimensions by just adding the approximation for the z-axis

$$\nabla^2 u_{i,j,k} = \frac{u_{i+1,j,k} - 2u_{i,j,k} + u_{i-1,j,k}}{h^2} + \frac{u_{i,j+1,k} - 2u_{i,j,k} + u_{i,j-1,k}}{h^2} + \frac{u_{i,j,k+1} - 2u_{i,j,k} + u_{i,j,k-1}}{h^2} \tag{3.65}$$

or

$$\nabla^2 u_{i,j,k} = \frac{1}{h^2}[(u_{i+1,j,k} + u_{i-1,j,k}) + (u_{i,j+1,k} + u_{i,j-1,k}) + (u_{i,j,k+1} + u_{i,j,k-1}) - 6u_{i,j,k}] \tag{3.66}$$

Now the value is the arithmetic average of its six nearest neighboring values instead of four in the two-dimensional case.

While this is simple extension in terms of both the abstract formulation and the resultant code (re)generation, by extending to the third dimension, we computational issues regarding size and speed become more relevant. This has been characterized by Golub [23] as the "curse of dimensionality." For example, in our simple numerical example of the solving Laplace's equation on the unit square 3.3.1, our last calculation used 900 grid points to obtain the solution. We noted that if we use an algorithm, such as Gaussian elimination that is $O(N^3)$ to solve the resulting system, we are looking at 729 MFLOPS (millions of floating point operations) and we would have to store 6.48 Mb of data (assuming we stored the whole matrix and used double precision floats). Let's not extend out to a three-dimensional problem. We now have 27,000 grid points, we would require 1.96×10^{13} floating point operations to solve the system, and would require 5.8 Gb of disk space to store the entire matrix. Obviously, we might have to re-think our solution and storage strategies! We will address these issues in a later section.

3.3.9 Error Estimates

Thus far in our application of the finite difference method to elliptic problems, we have concentracted on implementational details. We have discussed, in an intuitive manner, how one might decrease the error by reducing the size of the elements and have also noted that the five-point formula for the finite difference approximation is of $O(h^2)$. What we have yet to do is discuss error estimates in a more formal way. In general, analyzing errors is somewhat involved. We will just touch the surface of error analysis for finite difference methods. For a more in-depth analysis, look at the references, [18, 24, 25, 17].

Let's analyze the global error for the model problem:

$$\frac{d^2u}{dx^2} - p(x)\frac{du}{dx} - q(x)u = f(x), \tag{3.67}$$

with

$$q(x) \geq K > 0, \quad x \in (a, b), \tag{3.68}$$

with boundary conditions

$$u(a) = \alpha \quad u(b) = \beta. \tag{3.69}$$

We will approximate the solution to this equation using second-order accurate difference equations with mesh size h. We'll denote the true solution as $u(x)$ and the approximate solution as $\hat{u}$. Thus we have

$$\frac{1}{h^2}[\hat{u}(x_{i-1}) - 2\hat{u}(x_i) + \hat{u}(x_{i+1})] - \frac{p(x_i)}{2h}[\hat{u}(x_{i+1}) - \hat{u}(x_{i-1})] - q(x_i)\hat{u}(x_i) = f(x_i). \tag{3.70}$$

We define the **solution error** as $e(x_i) = \hat{u}(x_i) - u(ih)$, where $u(ih)$ denotes the exact solution at node x_i. If we substitute $u(ih)$ in for $\hat{u}$ in the previous equation, we obtain

$$\begin{aligned} \frac{1}{h^2}[u((i-1)h) - 2u(ih)) + u(i+1)h)] - \frac{p(ih)}{2h}[u((i+1)h) - u((i-1)h)] \\ -q(ih)u(ih) = f(ih) + O(h^2). \end{aligned} \tag{3.71}$$

Subtracting and noting that $x_i = ih$, we have

$$\frac{1}{h^2}[e(x_{i-1}) - 2e(x_i) + e(x_{i+1})] - \frac{p(x_i)}{2h}[e(x_{i+1}) - e(x_{i-1})] - q(x_i)e(x_i) = O(h^{2)}. \tag{3.72}$$

Our boundary data are the Dirichlet conditions at a and b. We can then represent this last equation in terms of the matrix formula,

$$\begin{pmatrix} a_1 & -c_1 & & & \\ -b_2 & a_2 & -c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & -b_{N-1} & a_{N-1} \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_{N-1} \end{pmatrix} = -h^2 \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{N-1} \end{pmatrix} \tag{3.73}$$

where

$$a_i = [2 + h^2 q(x_i)] \quad b_i = [1 + \frac{h}{2} p(x_i)], \tag{3.74}$$

and

$$c_i = [1 - \frac{h}{2} p(x_i)], \quad f_i = O(h^2) \tag{3.75}$$

We can rewrite this set of equations as

$$\mathbf{Ae} = -\mathbf{h^2 f} \tag{3.76}$$

and call the vector **e** the **discretization** or **global error** of the difference approximation. Or, assuming that A^{-1} exists, our error is

$$\mathbf{e} = -\mathbf{h^2 A^{-1} f} \tag{3.77}$$

This is the basic relationship between the global and local discretization errors.

Now what we want to determine is the behavior of A^{-1} as $h \to 0$. One difficulty here is that as $h \to 0$, n tends to ∞. However, we can give a simple analysis for a specific case such that $p(x) = 0$ and [23]

$$q(x) \geq \gamma > 0 \quad a \leq x \leq b \tag{3.78}$$

We set our global error, $e = \max |e_i|$ and our local error $f = \max |f_i| = O(h^2)$. From our error matrix equation and the condition on $q(x)$, we have

$$(2 + \gamma h^2)|e_i| \leq 2e + h^2 f \quad i = 1, \ldots, n \tag{3.79}$$

Since this equation must hold for all i, we obtain

$$(2 + \gamma h^2)|e| \leq 2e + h^2 f \quad i = 1, \ldots, n \tag{3.80}$$

Since $f = O(h^2)$ we conclude that

$$e \leq \frac{f}{\gamma} = O(h^2) \tag{3.81}$$

This says that the global discretization error is $O(h^2)$ provided that the local discretization error is $O(h^2)$. It can be shown that this result holds for the more general case, as well as for the important case when $q(x) = 0$. The analysis, however, gets somewhat involved.

Another way to look at the error is to use the eigenvalue $\lambda_{\min}(\mathbf{A})$ of $\mathbf{A}$ having the smallest magnitude to bound the global discretization error. Using the Euclidean norm, we find that

$$\|\mathbf{Ae}\|_2 = -h^2\|\mathbf{f}\|_2 \tag{3.82}$$

Since $\|\mathbf{Ae}\|_2 \geq |\lambda_{\min}(\mathbf{A})|\|\mathbf{e}\|_2$, we can obtain,

$$-h^2\|\mathbf{f}\|_2 \geq |\lambda_{\min}(\mathbf{A})|\|\mathbf{e}\|_2 \tag{3.83}$$

Whenever $\mathbf{A}$ is nonsingular, $\lambda_{\min}(\mathbf{A}) \neq 0$, then we have the following bound on the global error [18]:

$$\|\mathbf{e}\|_2 \leq \frac{-h^2\|\mathbf{f}\|_2}{|\lambda_{\min}(\mathbf{A})|}. \tag{3.84}$$

If $\lambda_{\min}(\mathbf{A})$ approaches a nonzero value as $h \to 0$, then the norm of the global error $\|\mathbf{e}\|_2 \to 0$ at least as fast as the truncation error $\|\mathbf{f}\|_2$.

3.4 Finite Difference Formulation for Parabolic Problems

We now investigate problems which involve both space and time and thus need to satisfy both initial and boundary conditions. Consider the time-dependent heat equation

$$\nabla^2 u = \frac{\partial u}{\partial t} \tag{3.85}$$

with boundary conditions

$$u(x, y, t) = g(x, y) \tag{3.86}$$

where $g(x, y)$ is a specifed function. To get a feeling for the interplay between space and time, let's first consider the one-dimensional version

$$\frac{\partial^2 u}{\partial x^2} = \frac{\partial u}{\partial t} \qquad a \leq x \leq b \tag{3.87}$$

with boundary conditions

$$u(x, 0) = g(x) \qquad u(a, t) = c_1, \qquad u(b, t) = c_2 \tag{3.88}$$

The simplest way to proceed is to use the central difference quotient to replace the second derivative in space and use a foward difference quotient to replace the time derivative [23].

$$\frac{u_j^{m+1} - u_j^m}{\Delta t} = \frac{1}{h^2}(u_{j+1}^m - 2u_j^m + u_{j-1}^m) \tag{3.89}$$

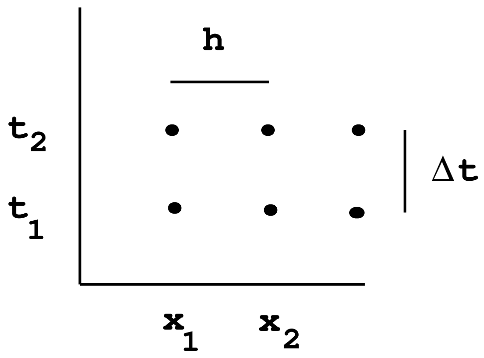


Figure 3.15: Discretization for the Heat Equation.

where u_j^m denotes the approximate solution at $x_j = jh$ and $t_m = m\Delta t$. This is shown in Figure 3.4 This is usually written as

$$u_j^{m+1} = u_j^m + \mu(u_{j+1}^m - 2u_j^m + u_{j-1}^m) \tag{3.90}$$

where $\mu = \frac{\Delta t}{h^2}$. The boundary conditions have values

$$u_0^m = c_1, \ ; \ u_{n+1}^m = c_2, \quad m = 0, 1, \ldots \tag{3.91}$$

and the initial condition is

$$u_j^0 = g(x_j), \quad j = 1, \ldots, n. \tag{3.92}$$

This description gives the approximate solution forward one time step after another. First the values $u_j^1, j = 1, \ldots, n$ are computed and then, knowing these, we can compute $u_j^2, j = 1, \ldots, n$ and continue the process until we have solved for all the u_j^m.

Accuracy of the method can be determined in much the same way as we proceeded with elliptic equations, we substitute the true solution into our finite difference approximation and look at the truncation error. It turns out that the local truncation error is, not surprisingly

$$e = O(\Delta t) + O(h^2) \tag{3.93}$$

which is usually stated that the discretization is first-order accurate in time and second-order accurate in space.

We would like to assume that the discretization error in u_j^m tends to zero as Δt and h tend to zero. Unfortunately, this is not necessarily the case. The statement that the local discretization error tends to zero with Δt and h is essentially a necessary condition for the global discretization error to tend to zero. That this happens is usually termed the consistency of the difference scheme. This consistency is connected with the stability of the difference formulation. In order for the previous difference formulation of the heat equation to be stable, Δt and h must satisfy the relation (see [23] for the derivation of this condition)

$$\Delta t \leq \frac{1}{2}h^2. \tag{3.94}$$

If this condition is not satisfied, then unstable behavior in u_j^m will develop - even in the abscence or rounding errors. That the stability criterion is met is also necessary for the global discretization error to tend to zero as Δt and h go to zero. One notices that the stability condition puts an ever increasing limitation on the size of Δt as h gets smaller. For example, at $h = 0.1$, $\Delta t = 0.5 \times 10^{-2}$ and at $h = 0.001$, $\Delta t = 0.5 \times 10^{-6}$. Therefore, in order to capture the spatially varying changes in our field, we may be having to over specify the time component. This is one of the reasons that so-called *implicit methods* were developed.

The finite difference approximation to the heat equation that we just implemented is called an *explicit method* because the values of u_j^{m+1} at the next time level are obtained by an explicit formula in terms of the values at the previous time level. Consider the following *implicit* difference approximation to the heat equation

$$\frac{u_j^{m+1} - u_j^m}{\Delta t} = \frac{1}{h^2}(u_{j+1}^{m+1} - 2u_j^{m+1} + u_{j-1}^{m+1}) \tag{3.95}$$

where again u_j^m denotes the approximate solution at $x_j = jh$ and $t_m = m\Delta t$. Or using $\mu = \frac{\Delta t}{h^2}$

$$(1 + 2\mu)u_j^{m+1} - \mu(u_{j+1}^{m+1} + u_{j-1}^{m+1}) = u_j^m \quad j = 1, \ldots, n. \tag{3.96}$$

and in vector form

$$(I + 2\mu A)\mathbf{u}^{m+1} = \mathbf{u}^m + \mathbf{b} \quad m = 0, 1, \ldots \tag{3.97}$$

While similar to the previous formulation, there is an important difference - the values of u_j are now evaluated at the $(m+1)$st time interval instead of the mth. This is the differnce between explicit and implicit methods. For explicit methods, we have a formula for u_j^{m+1} in terms of the known values u_j at previous time intervals. For implicit methods, however, we must solve equations to advance to the next time interval. The implicit method is carried out by solving the linear system of equations at each time step to obtain $\mathbf{u}^{m+1}$ from $\mathbf{u}^m$.

Now, what have we gained here? So far, it looks like we'll now have to spend even more time computing. For 1D problems, this isn't much, but for 2D and 3D problems, we'll have to solve large systems of equations for each advance in time. The answer lies in the stability condition. It can be shown that the stability condition for this implicit method is

$$0 < \frac{1}{1 + 2\mu(1 + \cos(k\pi h))} < 1. \tag{3.98}$$

Here, $\mu > 0$ and $1 + \cos(k\pi h) > 0$ are always satisfied. Most importantly, $\mu = \frac{\Delta t}{h^2}$ holds for **any** ratio of Δt and h. This means that the implicit method is unconditionally stable!