Numerical Solutions to Laplace's Equation

There are many elegant analytical solutions to Laplace's equation in special geometries but nowadays real problems are usually solved numerically. Computers and software are now so powerful that it can be easier to obtain a computer solution than to find the exact one in a reference book. There are two common approaches to finding a numerical solution:

1. Finite-Difference Methods

Finite-difference methods superimpose a regular grid on the region of interest and approximate Laplace's equation at each grid-point. The resulting equations are solved by iteration. The method is extremely easy to program. Start by considering a two-dimensional grid of points each separated by a distance h from its four nearest neighbours and the potential at a position (x,y) is $\phi(x,y)$. Then

$$\phi(x+h,y) = \phi(x,y) + h\frac{\partial\phi}{\partial x} + \tfrac{1}{2}h^2\frac{\partial^2\phi}{\partial x^2} + \tfrac{1}{6}h^3\frac{\partial^3\phi}{\partial x^3} + \mathrm{O}\left(h^4\right) \qquad [1a]$$

$$\phi(x-h,y) = \phi(x,y) - h\frac{\partial\phi}{\partial x} + \tfrac{1}{2}h^2\frac{\partial^2\phi}{\partial x^2} - \tfrac{1}{6}h^3\frac{\partial^3\phi}{\partial x^3} + \mathrm{O}\left(h^4\right) \qquad [1b]$$

which when added together give

$$\phi(x+h,y) + \phi(x-h,y) = 2\phi(x,y) + h^2\frac{\partial^2\phi}{\partial x^2} + \mathrm{O}\left(h^4\right). \qquad [2]$$

Adding this equation to the equivalent result along the y-direction

$$\phi(x,y) = \tfrac{1}{4}\left\{\phi(x+h,y) + \phi(x-h,y) + \phi(x,y+h) + \phi(x,y-h)\right\} + \mathrm{O}\left(h^4\right). \qquad [3]$$

because

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = 0 \qquad [4]$$

so the potential at a grid-point is simply the mean potential of its nearest neighbours. To find the solution one must fix the values of the grid-points on the boundaries in accordance the required boundary conditions and then iterate the potentials until successive results agree to within the desired tolerance.

Although the method works well as described, its convergence is a little slow (*i.e.* rather a lot of iterations are required) and it is useful to speed things up by introducing a so-called over-relaxation parameter λ, which has the effect of amplifying each step towards the final solution when $0 < \lambda$. The process of finding the next solution based on the previous is represented by

$$\phi_{n+1}(x,y) = \phi_n(x,y) + (\lambda+1)\Delta_n(x,y) \qquad [5]$$

where $0 \le \lambda < 1$ controls the degree of over-relaxation and

$$\Delta_n(x,y) = \tfrac{1}{4}\{\phi_n(x+h,y) + \phi_n(x-h,y) + \phi_n(x,y+h) + \phi_n(x,y-h)\} - \phi_n(x,y). \qquad [6]$$

If λ is too large the system will become unstable and oscillate; a value of about 0.5 seems to work quite well for the sort of problems being discussed here.

A FORTRAN or C computer program to perform this calculation needs only couple of dozen statements, all but about two of which perform house-keeping functions. A spreadsheet, such as Excel™, which has a natural grid structure and user interface makes finding and plotting solutions to small scale problems very easy (see the appendix).

2. Finite Elements

Finite element methods divide the problem of interest into a mesh of geometric shapes called finite elements. The potential within an element is described by a function that depends on its values at the cell corners and parameters defining the state of the element. Several such cells are assembled to solve the entire problem. A total energy associated with the mesh configuration is found as part of the calculation and this is minimised by adjusting the parameters specifying the elements. The solution can be refined by subdividing the regions of the mesh that contribute most to the total "energy" of the solution. General purpose programmes to perform these calculations are fairly complicated but fortunately very efficient commercial packages are available. I feel that these methods are under-exploited by physicists.

Example 3.6 on page 84 of RMC finds the charge distribution on a long thin conductor of net charge Q comprising N elements each of length $2d$ but with unknown charge. In this problem the charge q_i on the ith element will be adjusted to equalise the potential of each element. The first stage is to find an expression for the potential of the ith element which turns out to be

$$V_i = \frac{1}{8\pi\varepsilon_0 d}\sum_{j=1}^{N} A_{ij} q_i = \frac{1}{8\pi\varepsilon_0 d}\left[2q_i \ln\left(\frac{2d}{a}\right) + \sum_{j\neq i} q_j \ln\left(\frac{2|j-i|+1}{2|j-i|-1}\right)\right] \qquad [7]$$

where the first term is the "self-energy" of the element and the sum runs over all its neighbours. To minimise the total energy we already know that all the V_i must be equal to the potential of the conductor and this leaves N simultaneous linear equations to solve to find the unknown q_i. (In fact, a general purpose FE package would work directly with the total field energy of the system.)

The symmetry of the system is such that the charge on element j is the same as that on the $(N–j+1)$th element therefore when N is even, the matrix form of the problem can be written

$$\mathbf{V} = \begin{pmatrix} \mathbf{v} \\ \mathbf{Jv} \end{pmatrix} = \begin{pmatrix} \mathbf{A_1} & \mathbf{A_2} \\ \mathbf{A_2^T} & \mathbf{A_1} \end{pmatrix} \begin{pmatrix} \mathbf{q} \\ \mathbf{Jq} \end{pmatrix} = \mathbf{AQ} \qquad [8]$$

where **J**, having all its elements zero except for those on the diagonal that runs from top-right to bottom-left which are 1, reverses the order of the elements in a column vector, so

$$\mathbf{v} = \left(\mathbf{A_1} + \mathbf{A_2 J}\right)\mathbf{q} \qquad [9]$$

and the number of equations we have to solve has been halved.

Appendix: Technical notes about the spreadsheets

The Excel-4 spreadsheet files referred to here are available on the WWW from <http://newton.ex.ac.uk/teaching/CDHW/Electromagnetism/>.

Example 3.5 on page 82 of RMC deals with a two-dimensional rectangular region with its boundaries held at fixed potentials. The same problem has been solved on a spreadsheet which can be found in the file CW950226-1.XLS. The chart in the bottom right can be opened and resized by double-clicking, and you may wish to **Add Legend** from the **Chart** menu at the same time. The boundary potentials can be altered by editing the red numbers. For this worksheet iteration must be turned-on, the check-box to do this is on the **Calculation...** dialogue from the **Options** menu.

There are lots ways one could create the matrix A_{ij} in equation 7 to solve example 3.6 of RMC on the spreadsheet. I started by creating a horizontal sequence of integers from 1 to 40 starting in cell I12 to represent the values of *i* and similar vertical for set starting at H13 for *j* and a custom number format makes it clear what they are. After creating the names a, d, i and j the formula

```
=IF(i=j,2*LN(2*d/a),LN((2*ABS(i-j)+1)/(2*ABS(i-j)-1)))
```

goes into cell I13 and is then filled across and down to complete the array which is given the name MatrixA. The elements of the column matrix for q_i are calculated for the values V_i with the formula

```
{=MMULT(MatrixV,MINVERSE(MatrixA))}.
```

The results, which are in the file CW950313-1.pdf, disagree with those of RMC but the error appears to be their's. Although the initial calculation of the results was quite quick, (a few seconds) Excel seems to take a great deal of time (several minutes) to recalculate the results after changes are made to parameters on the worksheet, this is probably a bug in the software.

SOLUTION TO LAPLACE'S EQUATION IN A RECTANGULAR REGION

Solves example 3-5 of Reitz Milford and Christie: a rectangular region is surrounded by conducting walls at different potentials. Maximum iterations and convergence values can be changed with the "Options/Calculation" menu. RMC figure 3-10 seems to have a misprint in one value!

Relaxation Parameter (W-1):	Relax= 0.500
Potentials on the sides:	VTOP= 1.000
	VBOT= 0.000
	VLEFT= 0.300
	VRIGHT= 0.700

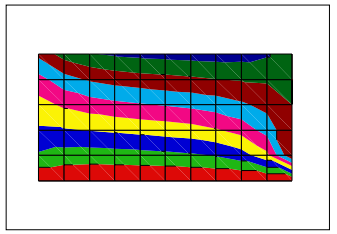
TOP

	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	
0.3000	0.6269	0.7445	0.7924	0.8146	0.8265	0.8344	0.8405	0.8447	0.8419	0.8140	0.7000
0.3000	0.4630	0.5588	0.6105	0.6393	0.6572	0.6706	0.6831	0.6962	0.7088	0.7141	0.7000
LEFT 0.3000	0.3665	0.4170	0.4516	0.4750	0.4923	0.5077	0.5250	0.5484	0.5830	0.6335	0.7000 RIGHT
0.3000	0.2859	0.2912	0.3038	0.3169	0.3294	0.3428	0.3607	0.3895	0.4412	0.5370	0.7000
0.3000	0.1860	0.1581	0.1553	0.1595	0.1656	0.1735	0.1854	0.2076	0.2555	0.3731	0.7000
	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	

BOTTOM

Previous values (used when Relax≠0)

0.6269	0.7445	0.7924	0.8146	0.8265	0.8344	0.8405	0.8447	0.8419	0.8140
0.4630	0.5588	0.6105	0.6393	0.6572	0.6706	0.6831	0.6962	0.7088	0.7141
0.3665	0.4170	0.4516	0.4750	0.4923	0.5077	0.5250	0.5484	0.5830	0.6335
0.2859	0.2912	0.3038	0.3169	0.3294	0.3428	0.3607	0.3895	0.4412	0.5370
0.1860	0.1581	0.1553	0.1595	0.1656	0.1735	0.1854	0.2076	0.2555	0.3731



CDH Williams 19950226/1