1 Numerical Methods

1.6 Multigrid Methods (10 units)

Knowledge of Part II Numerical Analysis would be advantageous for this project.

1 Solution of Poisson's Equation by Relaxation Methods

We consider the problem of solving Poisson's equation in a square domain with homogeneous Dirichlet boundary conditions

$$\nabla^2 u = f \qquad \text{in} \qquad 0 < x < 1, \quad 0 < y < 1, \tag{1}$$

with $u = 0$ on $x = 0$, $x = 1$, $y = 0$ and $y = 1$.

A numerical solution is attempted by finding values for u at grid points in a square $N \times N$ mesh. The (i,j)th point is given by $(x_i, y_j) = (ih, jh)$ where $h = 1/(N-1)$. The value of $\nabla^2 u$ is approximated at each of the interior points by a finite-difference formula

$$(\nabla^2 u)_{i,j} \simeq \frac{1}{h^2}\left[u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}\right]. \tag{2}$$

By requiring that $(\nabla^2 u)_{i,j}$ is equal to $f(x_i, y_j)$ at each of the interior points, we obtain $(N-2)^2$ linear equations for the $(N-2)^2$ unknowns $u_{i,j}$, $(1 \leqslant i \leqslant N-2,\ 1 \leqslant j \leqslant N-2)$, of the form

$$\frac{1}{h^2}\left[u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}\right] = f(x_i, y_j)\ . \tag{3}$$

The values of $u_{i,j}$ at the boundary points are set by the boundary conditions. Here $u_{i,j}$ is equal to zero at each boundary point.

We now have to solve these linear equations as quickly and as accurately as possible. Note that even if the solution of the linear equations were obtained with perfect accuracy, it would still be only an approximate solution to the original partial differential equation, since (2) is only an approximation to equation (1).

For larger values of N it is impractical to solve the $(N-2)^2$ linear equations by direct methods, such as Gaussian elimination, because of storage limitations. An alternative approach is to use an iterative "relaxation" method. Equation (3) may be reordered to suggest the iteration scheme

$$u^{n+1}_{i,j} = \frac{1}{4}\left[u^{n+1}_{i-1,j} + u^{n+1}_{i,j-1} + u^{n}_{i+1,j} + u^{n}_{i,j+1} - h^2 f(x_i, y_j)\right] \quad \text{for } i,j = 1,\ldots,N-2\,, \tag{4}$$

where the superscripts denote the number of the iteration; this is conventionally called the Gauss-Seidel scheme. Note the appearance of $(n+1)$th iterates on the right-hand side. The calculation works through the grid with i and j increasing, and updated values are used as soon as they become available.

Question 1 Take $f(x,y) = x(1-x)y^2(1-y)$. Write a program to apply the Gauss-Seidel scheme (4) to solve (3) on an arbitrary sized $(N \times N)$ mesh. Use your program to investigate the convergence properties of the scheme as N varies. In particular, after a reasonably large number of iterations you should calculate:

(a) the variation over the grid of the residual error, $\epsilon^n_{i,j}$, where the residual error is the amount by which (3) is not satisfied, i.e.

$$\epsilon^n_{i,j} = \frac{1}{h^2}\left[u^n_{i+1,j} + u^n_{i-1,j} + u^n_{i,j+1} + u^n_{i,j-1} - 4u^n_{i,j}\right] - f(x_i, y_j)\,, \tag{5}$$

(b) an approximation for the asymptotic rate of convergence, i.e.

$$r_\infty = -\log\left(\lim_{n\to\infty}\left\{\frac{\max_{i,j}|\epsilon^{n+1}_{i,j}|}{\max_{i,j}|\epsilon^n_{i,j}|}\right\}\right)\,. \tag{6}$$

Why is this a good definition of the rate of convergence?

What do you conclude about the number of iterations needed for convergence to a specified accuracy (e.g. for the magnitude of residual error to be less than a given tolerance at each point)? Estimate as a power of N the number of operations (i.e. additions, multiplications and divisions) needed for such convergence. Check your answer by measuring the computational time in different cases.[*] Suggested values for N that you might try are 9, 17, 33, 65, etc. Also estimate as a power of N the number of operations needed for convergence to an accuracy consistent with the truncation error of the discretisation (3) of equation (1).

2 The Multigrid Method

Your calculations should show that the part of the error that decays slowest for each N (and therefore that which dominates after a large number of iterations) has a form very similar to the lowest Fourier mode that will fit into the domain. The convergence is thus limited by large scales, not by small scales.

This motivates the multigrid method described below. The basic idea is that the error left after a few iterations is on scales much larger than the grid scale. The correction needed to the approximate solution to remove this error may therefore be determined more efficiently by transferring the error to a coarser grid, iterating on the coarser grid where convergence is more rapid, then transferring the calculated correction back to the finer grid, updating the approximate solution, and iterating on the finer grid again. The whole procedure is then repeated until the required convergence is achieved. Furthermore the procedure need not be confined to two grids. It is natural to improve the convergence of the coarse grid problem by transferring the error in that to a coarser grid still, and so on.

The multigrid procedure may be defined more exactly as follows. Assume that we have a sequence of K grids, labelled by $J = 1, \ldots, K$ in increasing order of fineness, the Jth grid having size $N_J \times N_J$. It is convenient to take the mesh spacing of the $(J-1)$th grid to be twice that of the Jth grid, i.e. $N_J = 2N_{J-1} - 1$.

On the Jth grid we wish to solve the linear system

$$\mathcal{L}_J \mathbf{u}^{(J)} = \mathbf{r}^{(J)}\,, \tag{7}$$

where the operator $\mathcal{L}_J$ corresponds to that acting on the left-hand side of (3), if $N_J = N$. Note that it is important when writing down the form of $\mathcal{L}$ for arbitrary J to remember that h in (3) must be replaced by $1/(N_J - 1)$.

[*] *Hint:* given the speed of current computers, the timing of a *single* run of your program might be dominated by start/end overheads.

Descending part of multigrid cycle

(A) Apply the Gauss-Seidel iteration scheme (hereafter G-S) ν_1 times to obtain an approximate solution $\tilde{\mathbf{u}}^{(J)}$. The error $\mathbf{v}^{(J)}$ in this solution therefore satisfies

$$\mathcal{L}_J \mathbf{v}^{(J)} = \mathbf{r}^{(J)} - \mathcal{L}_J \tilde{\mathbf{u}}^{(J)}. \tag{8}$$

(B) Transfer the problem of determining $\mathbf{v}^{(J)}$ to the coarser $(J-1)$th grid as

$$\mathcal{L}_{J-1} \mathbf{u}^{(J-1)} = \mathcal{R}(\mathbf{r}^{(J)} - \mathcal{L}_J \tilde{\mathbf{u}}^{(J)}) = \mathbf{r}^{(J-1)}\,, \tag{9}$$

where the operator $\mathcal{R}$ is known as the *restriction* operator (see below).

The descending part of the cycle repeats (A) and (B), transferring the correction at each stage to coarser and coarser grids, starting with $J = K$ and ending with $J = 2$.

Coarsest grid

(C) On the coarsest grid apply G-S ν_2 times to obtain an approximate solution $\tilde{\mathbf{u}}^{(1)}$.

Ascending part of cycle

(D) Transfer the approximate solution on the $(J-1)$th grid to the Jth grid to give a new approximation to the solution to the problem on that grid

$$\tilde{\mathbf{u}}^{(J)}_{new} = \tilde{\mathbf{u}}^{(J)}_{old} + \mathcal{P}\tilde{\mathbf{u}}^{(J-1)}\,, \tag{10}$$

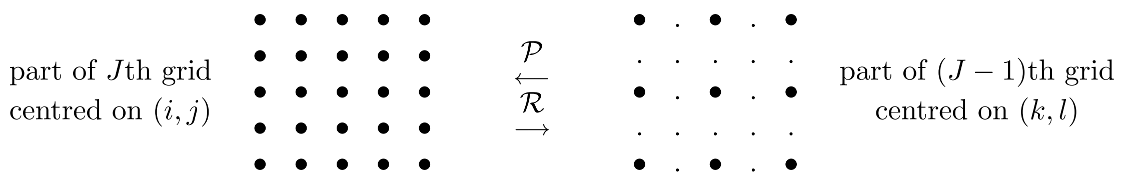
where $\mathcal{P}$ is the prolongation operator (see below).

(E) Apply G-S ν_3 times on the Jth grid to improve the approximation $\tilde{\mathbf{u}}^{(J)}$.

The ascending part of the cycle repeats (D) and (E), starting with $J = 2$ and ending with $J = K$ to leave an approximate solution to the full problem.

Note that within each multigrid cycle, the approximate solution $\tilde{\mathbf{u}}^{(J)}$ and the right-hand side $\mathbf{r}^{(J)}$ are generated from the problem on the $(J+1)$th grid during the descending part of the cycle and must be stored for use again at the Jth level during the ascending part of the cycle. Each $\mathbf{r}^{(J)}$ changes from cycle to cycle, except $\mathbf{r}^{(K)}$ which is always equal to $\mathbf{f}^{(K)}$ (i.e. the vector whose elements are f evaluated at each internal point of the Kth grid).

It remains to specify the restriction and prolongation operators $\mathcal{R}$ and $\mathcal{P}$ that you should use. It is natural to take both to be linear operators. Consider the following two sets of points.


part of Jth grid centred on (i,j)
$\mathcal{P}$ $\longleftarrow$
$\mathcal{R}$ $\longrightarrow$
part of $(J-1)$th grid centred on (k,l)

That on the left is a set of points in the Jth grid with the centre point labelled (i,j). That on the right is the same region in the $(J-1)$th grid. In the latter only those points marked with a • are included in the grid, with the centre point now labelled (k,l) say.

The prolongation operator $\mathcal{P}$ maps a function defined on points in the $(J-1)$th grid onto the points in the Jth grid. Similarly the restriction operator $\mathcal{R}$ maps a function defined on points in the Jth grid onto the points in the $(J-1)$th grid. It is convenient to represent both by the "masks"

$$\mathcal{P} = \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}, \qquad \mathcal{R} = \begin{bmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{bmatrix}.$$

That for $\mathcal{P}$ means that if $f = 0$ for all points in the $(J-1)$th grid except that labelled (k,l), then $\mathcal{P}f$ will be zero at all points on the Jth grid except for the square of nine points centred on (i,j) where it will take the values in the "mask". $\mathcal{P}f$ may be evaluated for general functions f by linearity. Each number in the mask for $\mathcal{R}$ represents the contribution from a point in the Jth grid, e.g. the square of nine points centred on (i,j) to the point (k,l) in the $(J-1)$th grid; note that points outside this square make no contribution.

Question 2 Write a program to apply the multigrid method as specified above. You will probably find it useful to have separate procedures/subprograms, working on grids of arbitrary resolution, to carry out each of the operations of prolongation and restriction, to calculate the residual in the difference equations and to apply the Gauss-Seidel iteration (exploit your existing program from question 1 here).

Apply the multigrid method to the solution of the same equation as in question 1. Investigate the rate of convergence associated with a single multigrid cycle for a fixed resolution of the finest grid, particularly its dependence on

(i) the resolution of the coarsest grid;

(ii) the number of times that the G-S iteration is applied at each stage, i.e. ν_1 (on each grid during the descending cycle), ν_2 (on the coarsest grid), and ν_3 (on each grid during the ascending cycle).

To start with, a suggested value for N_K is 65, for N_{K-1} is 33, etc. In each case estimate the total number of operations in a complete cycle, and give a measure of the numerical efficiency. ***Justify carefully the measure of efficiency that you are using*** (e.g. remember to include the cost of all operations within a cycle).

What are your conclusions about the best choices for the resolution of the coarsest grid, and for the numbers ν_1, ν_2 and ν_3? Next, choose suitable values of N_1, ν_1, ν_2 and ν_3, and investigate the dependence of the rate of convergence on N_K. Finally, discuss the improvement in efficiency of multigrid over the simple Gauss-Seidel iteration in question 1 when the aim is convergence to an accuracy consistent with the truncation error of the discretisation (3) of equation (1).

References

[1] Briggs, William L., Henson, Van Emden, McCormick, Steve F. (2000) *A Multigrid Tutorial*, SIAM (ISBN 0-89871-462-1).