MA50177: Scientific Computing
Case Study

Numerical Solution of Elliptic PDEs – Parallel Multigrid

This assignment is about practical aspects of solving sparse systems of linear equations

$$A\boldsymbol{U} \;=\; \boldsymbol{b} \tag{1}$$

arising from discretisations of 2nd–order elliptic PDEs using the multigrid method. In particular we will use parallel multigrid to solve finite diference discretisations of Poisson's equation

$$-\frac{\partial^2 u}{\partial x_1^2} \;-\; \frac{\partial^2 u}{\partial x_2^2} \;=\; g \tag{2}$$

where g is a given function on the square domain $\Omega = [0,1] \times [0,1]$, subject to the boundary condition $u = 0$ on Γ, the boundary of Ω.

The finite difference approximation of this problem on a uniform grid $\mathcal{T}$ with mesh width $h = 1/m$ for some $m \in \mathbb{N}$ is described on the Handout in Week 5. It leads to an $n \times n$ sparse system of linear equations of the form (1) with $n = (m-1)^2$.

A multigrid method applied to this problem was described in the Lecture and in the Tutorial in Week 10. It assumes the existence of a sequence of nested grids $\mathcal{T}_1, \ldots, \mathcal{T}_F$, with $\mathcal{T}_1$ being the coarsest grid and $\mathcal{T}_F$ being the finest, such that the mesh widths $h_{\ell-1}$ and h_ℓ of two consecutive grids $\mathcal{T}_{\ell-1}$ and $\mathcal{T}_\ell$, $\ell = 2, \ldots, F$, satisfy $h_\ell = \frac{1}{2}h_{\ell-1}$. In the following a subscript ℓ on any matrix, vector or scalar refers to the corresponding object arising from a discretisation on grid $\mathcal{T}_\ell$, in particular we denote the discretised Laplacian matrix on each grid by A_ℓ, $\ell = 1, \ldots, F$.

As described in Week 10 we also need intergrid transfer operators (i) to restrict vectors on a grid $\mathcal{T}_\ell$ to vectors on the next coarser grid $\mathcal{T}_{\ell-1}$ and (ii) to prolongate vectors on a grid $\mathcal{T}_{\ell-1}$ to vectors on the next finer grid $\mathcal{T}_\ell$. We will use *trivial injection* for the former and *linear interpolation* for the latter. The restriction operators can be straightforwardly represented by $n_{\ell-1} \times n_\ell$ (sparse) restriction matrices $R_\ell^{\ell-1}$, for each $\ell = 2, \ldots, F$. Similarly the prolongation operators can be represented by $n_\ell \times n_{\ell-1}$ (sparse) prolongation matrices $P_{\ell-1}^{\ell}$. The exact form of these was given in the Tutorial in Week 10.

As outlined in the lectures in Week 10, the idea in the multigrid method is to use a clever interplay between a smoother (here we will always use a damped Jacobi iteration, see the lectures in Week 10) and a correction on the next coarser level. In this way, starting on the finest grid and recursively applying the smoothing and the coarse grid correction step we can "coarsen" the problem until we reach the coarsest grid $\mathcal{T}_1$ where we solve the problem with any method of our choice (here we will use Conjugate Gradients). The full algorithm (called a multigrid V-cycle) is given below. The parameters ω, ν_{pre} and ν_{post} need to be specified.

Algorithm Multigrid: Choose *initial guess* $\boldsymbol{U}_F = \boldsymbol{0} \in \mathbb{R}^{n_F}$, and *tolerance* $\varepsilon > 0$.

```
Set r_F^(0) = b_F.                                   (initial residual)
do  k = 1, ..., maxits
     MGVCycle(A_F, U_F, b_F, F)                      (apply one V-cycle)
     r_F = b_F − A_F U_F                             (residual update)
     if ( ||r_F|| ≤ ε ||r_F^(0)|| ) exit             (stopping criterion)
end do
```

The core of this method is the following **recursive** subroutine:

Recursive Subroutine `MGVcycle`:

INPUT: matrix A_ℓ, initial guess $\boldsymbol{v}_\ell$ (possibly nonzero), right hand side $\boldsymbol{b}_\ell$, grid level ℓ.

OUTPUT: approximation to $\boldsymbol{v}_\ell$ after application of one V-Cycle on grid $\mathcal{T}_\ell$.

```
if  (ℓ = 1)  then
     v_1 = A_1^{-1} b_1                                  (solve problem on coarsest grid T_1)
else
     do k = 1, ..., ν_pre
          v_ℓ ← v_ℓ + ω D_ℓ^{-1} (b_ℓ − A_ℓ v_ℓ)         (ν_pre pre-smoothing steps)
     end do

     d_ℓ = b_ℓ − A_ℓ v_ℓ                                 (calculate the defect on grid T_ℓ)
     d_{ℓ-1} = R_ℓ^{ℓ-1} d_ℓ                             (restrict the defect onto grid T_{ℓ-1})
     v_{ℓ-1} = 0                                         (set initial guess to zero on grid T_{ℓ-1})
     MGVCycle(A_{ℓ-1}, v_{ℓ-1}, d_{ℓ-1}, ℓ − 1)          (apply V-cycle on grid T_{ℓ-1})
     v_ℓ ← v_ℓ + P_{ℓ-1}^ℓ v_{ℓ-1}                       (prolongate and add coarse grid correction)

     do k = 1, ..., ν_post
          v_ℓ ← v_ℓ + ω D_ℓ^{-1} (b_ℓ − A_ℓ v_ℓ)         (ν_post post-smoothing steps)
     end do
end if
```

For more information on multigrid methods see Briggs [1] (probably the best for first-timers), Hackbusch [2], Iserles [3] or any other of the many books on multigrid methods in the library.

References

[1] William L. Briggs, Van Emden Henson, and Steve F. McCormick, *A Multigrid Tutorial*, SIAM, 2000 (4 copies in the library: 2nd edition `513.75 BRI`, 1st edition `513.73 BRI`).

[2] Wolfgang Hackbusch, *Multi-Grid Methods and Applications*, Springer, 1985 (3 copies in the library: `513.75 HAC`).

[3] Arieh Iserles, *A First Course in the Numerical Analysis of Differential Equations*, Cambridge University Press, 1996 (5 copies in the library: `512.97 ISE`).