Iterative Solvers for Linear Systems of Equations

While Solving Differential equations it happens a lot that we reach to equations which are not easy to solve or not at all solvable , or sometimes we reach to a point where we need to solve linear systems of equations which might take lots of time to solve analytically, there might be lots of other cases like these where we might think of computers to helps us solving the problem numerically and fast enough to be able to go on.

To solve differential equations or linear system of equations there are lots of methods such as Jacobi Method, gauss Seidl Method, etc.

In the following hand out first I will start with the Poisson equation, discretise it to get a linear system of equations and then try to solve it with Jacobi Method and analyze the convergence of the Jacobi Method and then introduce the Multigrid Technique and compare them with each other.

Poisson Equation

Assume while solving our physical problem we reach to a point that we need to solve the following Poisson equation

$$\Delta u = \nabla^2 u = f$$

To solve this equation numerically we need to bring it into finite difference notation. After bringing it into finite difference notation we get to

$$\frac{-u_{j-1} + 2u_j - u_{j+1}}{h^2} = f(x_j) = f_j \quad , \quad h = \frac{b-a}{n}$$

or

$$\frac{1}{h^2}\begin{bmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 2 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ \vdots \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ \vdots \\ \vdots \\ f_n \end{bmatrix} \rightarrow Au = f$$

Now we have changed our continuous Poisson equation into a linear system of equations which is easier

to solve numerically. The next step is to find a way to solve this system.

Jacobi Method

Now that we have our system formulated we need to specify the method we will use to solve this system. We can simply rewrite the matrix A in the following way,

$$A=D-L-U$$
$$\rightarrow (D-L-U)u=f \rightarrow Du=(L+U)u+f$$
$$\rightarrow u=D^{-1}((L+U)u+f)$$

As it is seen above the matrix A is decomposed into the Diagonal , Lower triangle and Upper triangle part.
One might think that the calculation of the D-inverse is hard enough to leave this method but in fact it is not, while D is a Diagonal matrix and the inverse is simply devision by the diagonal element!
Now we can define our Jacobi iteration matrix as

$$P_j=D^{-1}(U+L)$$

Then our Jacobi method becomes

$$u^{(1)}=P_j u^{(1)}+D^{-1}f$$

Although the relation above helps us solve our system but as it will be shown later it converges slow which makes it almost impossible when the number of equation get higher. Therefore we need to find an alternative which converges faster to use.
As an alternative one might think of some modification to our Jacobi method, and the modification is just to add a factor of the diagonal element to calculate itself for the next iteration

$$u^{(1)}=(1-w)u^{(0)}+w(P_j u^{(0)}+D^{(-1)}f)$$

Where the w is called damping factor and one can see that with choosing w=1 we get to the original Jacobi formulation. For simplicity of the following proofs we rewrite the modified Jacobi formula in the following way

$$u^{(1)}=P_w u^{(0)}+wD^{-1}f\,,\,P_w=(1-w)I+wP_j$$

And we have

$$P_w = I - \frac{w}{2}\begin{bmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 2 \end{bmatrix}$$

And this shows us that the eigenvalues of A and P_w are related as follows

$$\lambda(P_w) = 1 - \frac{w}{2}\lambda(A)$$

So all we have to do is to find the eigenvalues of A. The eigenvalues of A are

$$\lambda_k(A) = 4\sin^2\left(\frac{k\pi}{2N}\right) \quad , \quad 1 \leqslant k \leqslant N-1$$
$$\rightarrow \lambda_k(P_w) = 1 - 2\sin^2\left(\frac{k\pi}{2N}\right)$$

And they both have the same eigenvectors which are

$$w_{jk} = \sin\left(\frac{jk\pi}{N}\right) \quad , \quad 1 \leqslant j \leqslant N$$

Now all we have to do is to find the best damping factor which leeds to a fast converging modified Jacobi. So we expand the error for the initial guess in the basis of A's Eigenvectors

$$e^{(0)} = \sum_{k=1}^{N-1} c_k w_k \quad , \quad c_k \in \mathbb{R}$$

After n iterations of the modified Jacobi the error is going to be

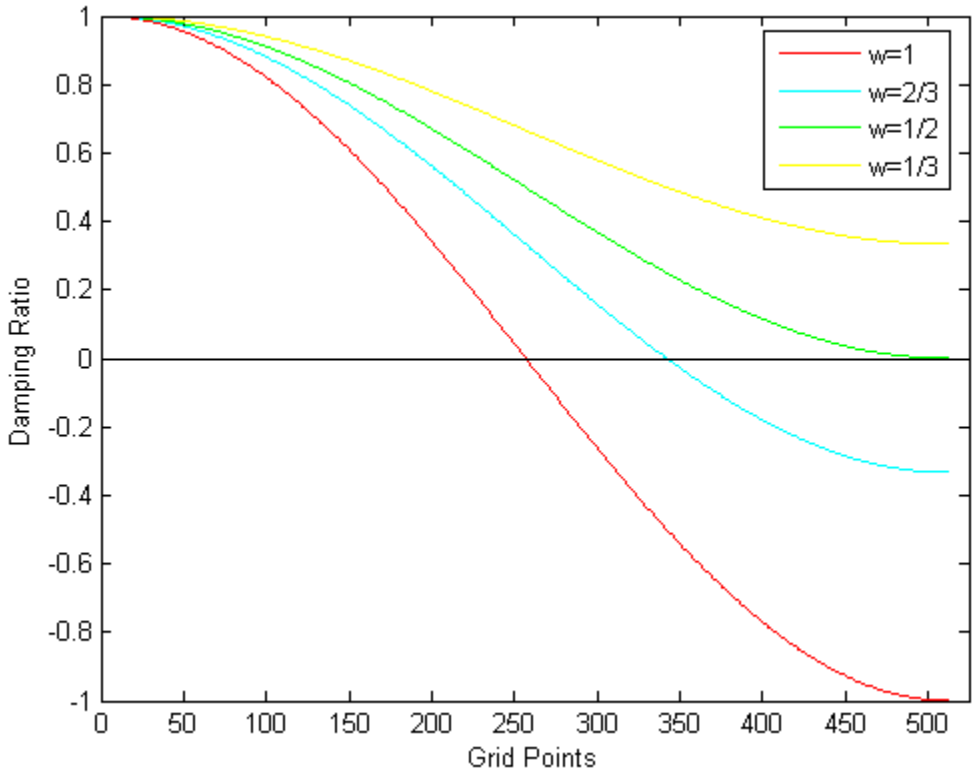
$$e^{(n)} = P_w e^{(0)} = \sum_{k=1}^{N-1} c_k P_w w_k = \sum_{k=1}^{N-1} c_k \lambda_k^n(P_w) w_k$$

As shown above the modes don't mix but they only damp by a factor, and this happens while the A and P_w have the same Eigenvectors w_k.

Since we want our error to damp to zero we can easily see that the condition for Jacobi method to converge is

$$|\lambda_k(P_w)| < 1 \rightarrow 0 < w \leqslant 1$$

Now just by solving the last equation for the error after n iterations one can easily find that by choosing the w to be 0.5 we can simply damp the high frequencies fast.
In order to solve the system with this formulation, we need to have an initial guess, apply the Jacobi Iteration matrix on it iteratively and after each iteration check the stopping criteria to see if we should stop or continue.
But still no matter what the factor is the low frequencies remain and damp slow enough to make our method converge slow.
This problem makes us think about another method which not only damps the high frequencies but also does the same to the low frequencies.
In the next graph the ratio of damping for 4 different w is shown on the plot and it is seen that the w=0.5 is the optimal factor.



The Multigrid

As we saw in the Jacobi method the modified Jacobi is fast for high frequencies but slow for low frequencies. But the smooth modes in the fine grid are oscillatory in the next coarse grids.
The low damping effect of Jacobi occurs while we use in the finite difference method only the close neighbors to calculate the differentiation. But by going to a coarser grid and doing the same process we are actually using the second close neighbors to calculate the differentiation. and as we go more down to the coarser grids we will use the n^{th} closest neighbors. Therefore the low frequencies in the finer grids seem to be high frequencies in the coarser ones, while the second close neighbors in the current gird are the closest neighbors in the next coarser grid.
So the basis of the Multigrid is to go all the way down to the coarsest grid and damp the high frequencies at each coarse grid and once we are in the coarsest grid find a way to return to the finest grid where the original problem was.
So now all we do is to perform some iterations of the Modified Jacobi in the finest grid to get a rather smooth solution and then calculate the residual and pass the residual every second point to the next coarser grid to do more damping there.
But how do we calculate the residual? Or in other words how should we go to the coarser grid?

Residual

When we have done some iterations of modified Jacobi on the finest grid, we have a rather smooth but slow converging error which we prefer to deal with in the next coarser grid, lets take a look at the error before going to the coarser grid, we know that

$$e = u - v$$

u being the exact solution, v the approximation and e the approximation error, but since we don't know the exact solution we can't solve the above relation to find the error, but we know the following equation

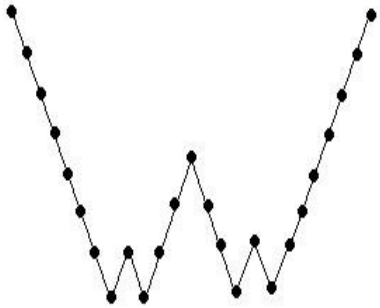
$$Ae = r = Au - Av = f - Av$$

This tells us that we can do the smoothing in the next coarser grid directly on the error.
Now in the finest grid after doing some iterations of the modified Jacobi, we find r (right hand side of the above equation) which is straight forward and pass it to the coarser grid by choosing every second point and then there we try to solve the above equation to find the error and add it as a correction to the results achieved in the finest grid.
But as mentioned before we don't go back at this stage to the finest grid, we keep on going down to the coarsest grid by solving the error equation at each coarser grid.
Now that we are in the coarsest grid and done with the corrections how should we go back to the finer grids and add the correction? we don't have the same grid points to go simply back!

At this point we need to find a way to go back, and the interpolation might help us through this.
This means we can interpolate the intermediate points which we don't have and get the same grid point numbers as the upper finer grid.
So all we do in going back is

$$u_{2j+1}^{fine\ grid} = \frac{u_j^{coarse\ grid} + u_{j+1}^{coarse\ grid}}{2} \quad , \quad u_{2j}^{fine\ grid} = u_j^{coarse\ grid} \quad , \quad u_{2j+2}^{fine\ grid} = u_{j+1}^{coarse\ grid}$$

After calculating the intermediate points we pass the results to the finer grid and add them to the previous result and perform some iterations of the modified Jacobi as post smoothing to avoid the probable high frequencies occurred by going down and coming back up and do the same process to go back to the original grid which is the finest.
After reaching the finest grid we check our stopping criteria to see if our result is satisfactory for us, if so we return the result as the solution else we keep on doing the same as we did till the stopping criteria is fulfilled.
The complete cycle of Multigrid is also called a V-cycle while we go all the way down and come back up again, but also other ways to do the Multigrid are possible such as the W-cycle.



References

- Prof. Bungartz, Numerical Programming course, TUM CSE (WS 07)
- Dr. Bader , Scientific Computing course, TUM CSE (WS 07)
- Dr Mehl, Scientific Computing Lab course , TUM CSE (WS 07)
- A Multigrid tutorial, William L. Briggs