Diffusion in 2 Dimensions

Christopher Pattison

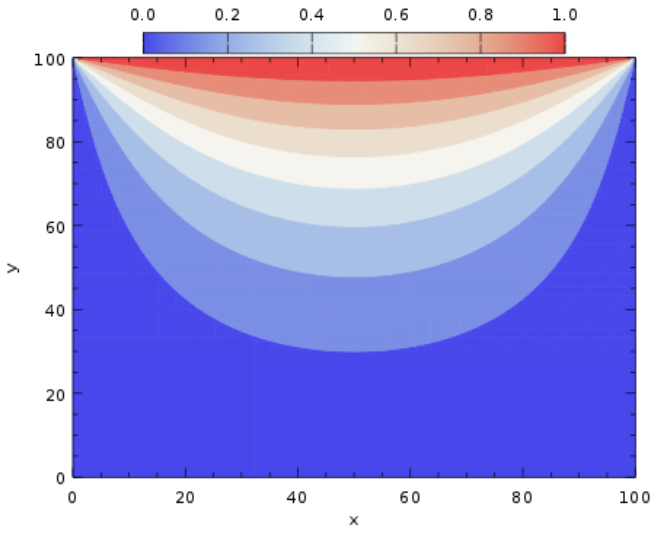

Figure 1: Solution on a 100x100 cell domain

Derivation

The Poisson equation was descretized using the finite volume method.

$$\int_V (\nabla \cdot k\nabla T - q)dV = 0 \tag{1}$$

As usual, a residual is defined and then integrated over the control volume. Using the divergence theorem and the assumption that q and k are invariant, this can be restated as equation (2).

$$\sum_N \vec{S}_N \cdot k\nabla T - qV = 0 \tag{2}$$

The last step is to approximate the gradient ∇T which is taken as equation (3) where $\vec{x}$ is the node position and S is the area of the cell face.

$$\nabla T_n \approx \frac{T_N - T_P}{|\vec{x}_N - \vec{x}_P|} S \tag{3}$$

On a structured quadrilateral grid with no skewed cells, $\vec{x}$ can be treated as a scalar. If the grid is uniform, then $|\vec{x}_N - \vec{x}_P| = S$. The coefficients are then simply

$$a_N = a_S = a_E = a_W = k \tag{4}$$

$$a_P = -\sum_N a_N \tag{5}$$

Solvers

The resulting coefficient matrix consists of a tridiagonal portion and two bands. Since the coefficient matrix is no longer tridiagonal, a direct solver becomes extremely expensive.

Gauss-Seidel

Gauss-Seidel is a good choice of solver since the simplicity means that no copy operations have to be carried out. Moreover, there is no inconvenience to leaving the coefficient matrix in a form that allows easy handling of the solution vector.

In this solver, T was a 2-dimensional array with indicies corresponding to the location of the node in the domain. Conveniently, node positions as well as cell neighbors do not have to be stored.

$$T_{i,j} = (T_{i+1,j}a_E \ldots T_{i,j-1}a_S - b_{i,j})/a_P \tag{6}$$

Line Solver

Since TDMA cannot be used on a banded matrix, one option is to hold some of the bands constant which allows the matrix to be put in tridiagonal form. While this should result in faster dissemination of the boundary conditions into the domain, a coefficient matrix must be explicitly formed. This tradeoff can result in poor performance.

Hybrid Solver

Two hybrid solvers were created that use the line solver before Gauss-Seidel based on the theory that the line solver is much quicker than Gauss-Seidel to provide a rough solution.

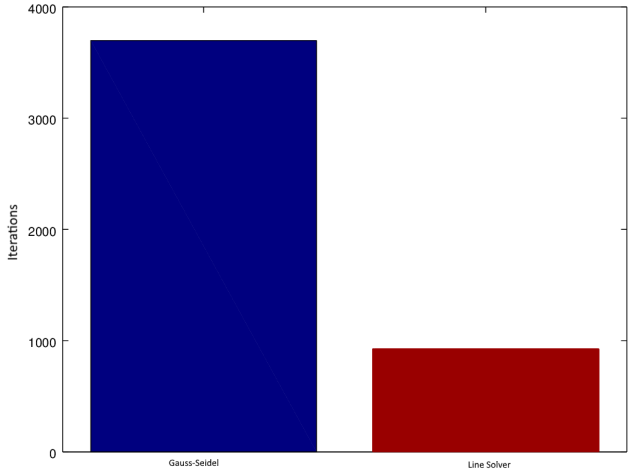

Figure 2: Iteration Comparison

A single iteration of Hybrid 1 uses two iterations of the line solver sweeping through the X and Y directions respectively following up with 16 iterations of Gauss-Seidel.

Hybrid 2 uses the line solver to approximate a solution with a high residual, it then iterates with Gauss-Seidel until convergence.

Performance

The comparisons were made on a 100x100 uniform grid solving for the solution shown in figure 1. The required residual for convergence was 10^{-6}.

While the line solver used fewer iterations to obtain a solution, the computational cost per iteration was higher, resulting in a longer time to completion.

Since the hybrid solvers use many sub-iterations, it is difficult to make a meaningful comparison on the basis of iteration. However, the wall time can be compared quite easily.

It is notable that Hybrid 2 performed the best as it supplies a rough initial guess obtained with the line solver to Gauss-Seidel, supporting the basis on which the hybrid solvers were implemented.

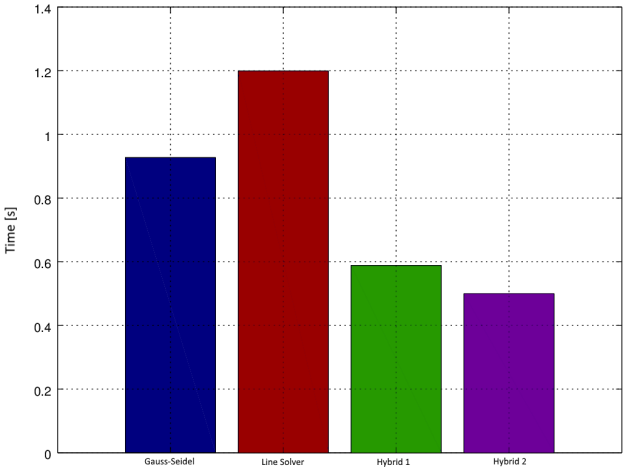

Figure 3: Wall Time Comparison