

# 1

# A Block Red-Black SOR Method for a Two-Dimensional Parabolic Equation Using Hermite Collocation

Stephen H. Brill<sup>1</sup> and George F. Pinder<sup>2</sup>

<sup>1</sup>Department of Mathematics and Statistics  
University of Vermont  
Burlington, Vermont 05405  
U. S. A.

<sup>2</sup>Department of Civil and Environmental Engineering  
University of Vermont  
Burlington, Vermont 05405  
U. S. A.

## 1 Introduction

In [LHHR95], Lai *et al.* study a block Jacobi method to solve the two-dimensional Poisson equation

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = H(x, y) \quad (1)$$

defined on the interior of the unit square  $\mathcal{S} = [0, 1] \times [0, 1]$ , discretized by the collocation method with a uniform mesh, given Dirichlet boundary conditions

$$u(x, y) = C(x, y), \quad (x, y) \in \partial\mathcal{S}. \quad (2)$$

They determine eigenvalues for the iteration matrix of their block Jacobi method and then use the theory in [You71] to determine a formula for  $\omega_{\text{opt}}$ , the optimal relaxation factor  $\omega$  for the block SOR method associated with their block Jacobi scheme. In this paper, we explain how to extend their work to ensure that the optimal SOR method is parallelizable by using a red-black ordering scheme. We then use these ideas to efficiently solve the two-dimensional parabolic equation

$$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = -H(x, y, t)$$

with Dirichlet boundary conditions.

## 2 Hermite Cubic Polynomials

### 2.1 One-dimensional formulation

Let  $u(x)$  be a function defined on the interval  $[0, 1]$ . Partition the interval using  $n$  equally spaced nodes  $0 = x_0, x_1, \dots, x_m = 1$ , where  $m = n - 1$  is the number of elements. Let  $h = 1/m$ .

Consider the functions (cf. [Pic94])

$$f_j(x) = \begin{cases} \frac{(x - x_{j-1})^2}{h^3} [2(x_j - x) + h], & x_{j-1} \leq x \leq x_j \\ \frac{(x_{j+1} - x)^2}{h^3} [3h - 2(x_{j+1} - x)], & x_j \leq x \leq x_{j+1} \\ 0, & \text{otherwise} \end{cases}$$

and

$$g_j(x) = \begin{cases} \frac{(x - x_{j-1})^2(x - x_j)}{h^2}, & x_{j-1} \leq x \leq x_j \\ \frac{(x_{j+1} - x)^2(x - x_j)}{h^2}, & x_j \leq x \leq x_{j+1} \\ 0, & \text{otherwise} \end{cases}$$

These are the Hermite cubic polynomials that we use as basis functions in our collocation approach. Notice that

$$\begin{aligned} f_j(x_i) &= \delta_{i,j} \\ \frac{df_j}{dx}(x_i) &= f'_j(x_i) = 0 \quad \forall i, j \\ g_j(x_i) &= 0 \quad \forall i, j \\ \frac{dg_j}{dx}(x_i) &= g'_j(x_i) = \delta_{i,j}, \end{aligned}$$

where  $\delta_{i,j}$  is the Kronecker symbol.

Let  $u_j = u(x_j)$  and let  $u'_j = u'(x_j) = \frac{du}{dx}(x_j)$  for  $j = 0, 1, \dots, m$ . Then the cubic polynomial interpolating the  $u_j$ 's and the  $u'_j$ 's is

$$\hat{u}(x) = \sum_{j=0}^m (u_j f_j(x) + u'_j g_j(x)). \quad (3)$$

In [Pap83], Papatheodorou uses  $g_j^*(x) = \frac{g_j(x)}{h}$  (in place of  $g_j(x)$ ) when forming (3). He makes this choice (also used in [LHHR95]) because eigenvalue analysis is much easier using  $g_j^*(x)$  instead of  $g_j(x)$ . It is easily seen that the iteration matrices studied herein that one obtains using  $g_j(x)$  and  $g_j^*(x)$  are identical. In this paper, we use  $g_j(x)$  in the computer code that generates the numerical results and employ  $g_j^*(x)$  for our analysis.

### 2.2 Two-dimensional formulation

Let  $u(x, y)$  be a function defined on  $\mathcal{S}$ . Partition  $\partial\mathcal{S}$  by using  $n$  equally spaced nodes in both the  $x$ - and  $y$ -directions. Letting  $m = n - 1$  and  $h = 1/m$ , we partition  $\mathcal{S}$  into  $m^2$  square elements, where the dimensions of each element are  $h \times h$ . If we consider two-dimensional bi-cubic Hermite basis polynomials, we obtain, by analogy to (3):

$$\hat{u}(x, y) = \sum_{q=0}^m \sum_{r=0}^m [u_{qr} f_q(x) f_r(y) + u_{qr}^x g_q(x) f_r(y) + u_{qr}^y f_q(x) g_r(y) + u_{qr}^{xy} g_q(x) g_r(y)], \quad (4)$$

where

$$\begin{aligned} u_{qr} &= u(x_q, y_r) \\ u_{qr}^x &= \frac{\partial u}{\partial x}(x_q, y_r) \\ u_{qr}^y &= \frac{\partial u}{\partial y}(x_q, y_r) \\ u_{qr}^{xy} &= \frac{\partial^2 u}{\partial x \partial y}(x_q, y_r). \end{aligned}$$

We see that  $\hat{u}(x, y)$  interpolates the functions  $u$ ,  $\frac{\partial u}{\partial x}$ ,  $\frac{\partial u}{\partial y}$ , and  $\frac{\partial^2 u}{\partial x \partial y}$  at the grid points  $(x_q, y_r)$ , for  $q, r = 0, \dots, m$ .

### 3 Collocation Discretization of the PDE

If the interpolating polynomial (4) is introduced into the governing equation (1), we obtain

$$\frac{\partial^2 \hat{u}}{\partial x^2} + \frac{\partial^2 \hat{u}}{\partial y^2} - H(x, y) = E(x, y),$$

where  $E(x, y)$  is an error function.

We see that at each of the  $n^2$  grid points  $(x_q, y_r)$ , we have four degrees of freedom, namely  $u_{qr}$ ,  $u_{qr}^x$ ,  $u_{qr}^y$ , and  $u_{qr}^{xy}$ . However, on the boundary  $\partial\mathcal{S}$ , many of these values are known. In particular, we know (from (2))

$$u_{qr} = u(x_q, y_r) = C(x_q, y_r)$$

for all nodes (grid points) on  $\partial\mathcal{S}$ . In addition, we can calculate

$$u_{qr}^x = \frac{\partial u}{\partial x}(x_q, y_r) = \frac{\partial C}{\partial x}(x_q, y_r)$$

on the north and south boundaries and

$$u_{qr}^y = \frac{\partial u}{\partial y}(x_q, y_r) = \frac{\partial C}{\partial y}(x_q, y_r)$$

on the east and west boundaries. We therefore know the values of a total of  $8n - 4$  degrees of freedom and do not know the values of  $4m^2$  degrees of freedom. Therefore, to uniquely determine these  $4m^2$  degrees of freedom we require  $4m^2$  equations, or 4 equations per element. To achieve this, we choose four points  $(x_k, y_\ell)$  in the interior of each element and enforce  $E(x_k, y_\ell) = 0$  at each of these  $4m^2$  ‘‘collocation points’’. It is known (from [Cel83]) that the optimal choices for the collocation points for the symmetric differential operator given in (1) are the so-called ‘‘Gauss points’’. On the interval  $[-1, 1]$ , the Gauss points are  $\pm z$ , where  $z = 3^{-1/2}$ . On the square element  $[-1, 1] \times [-1, 1]$ , the Gauss points are  $(-z, -z)$ ,  $(-z, z)$ ,  $(z, -z)$ , and  $(z, z)$ . Transforming these four Gauss points into each of the  $m^2$  elements of our mesh defines the full set of  $4m^2$  ‘‘collocation’’ equations. These can be written

$$\begin{aligned} & \sum_{q=0}^m \sum_{r=0}^m \{ [f_q''(x_k) f_r(y_\ell) + f_q(x_k) f_r''(y_\ell)] u_{qr} + [g_q''(x_k) f_r(y_\ell) + g_q(x_k) f_r''(y_\ell)] u_{qr}^x \\ & + [f_q''(x_k) g_r(y_\ell) + f_q(x_k) g_r''(y_\ell)] u_{qr}^y + [g_q''(x_k) g_r(y_\ell) + g_q(x_k) g_r''(y_\ell)] u_{qr}^{xy} \} = H(x_k, y_\ell), \quad (5) \end{aligned}$$

where  $(x_k, y_\ell)$  varies over all  $4m^2$  collocation points.

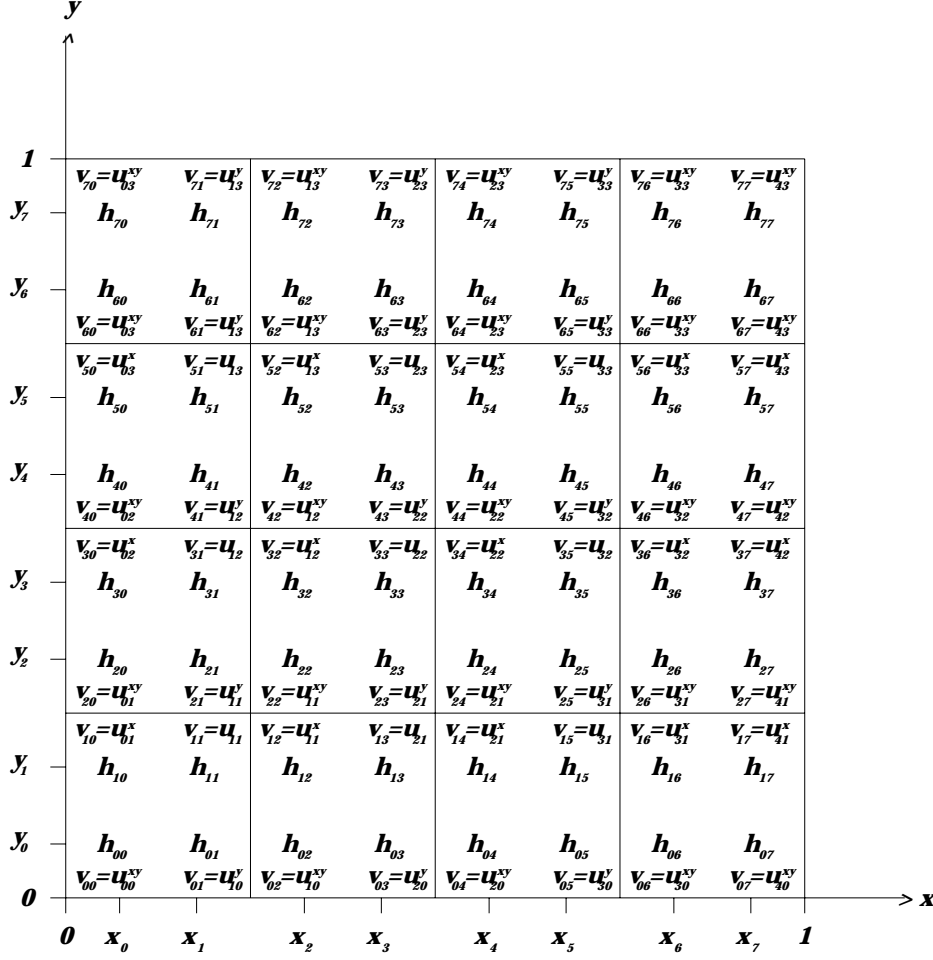


Figure I numbering of equations and unknowns

There are many ways in which to number the unknowns and equations. Each numbering system will define a different structure for the matrix arising from the system of linear equations (5) that we must solve. We use a numbering proposed by [Cel83] and by [LHHR95], which is depicted pictorially in Figure I for the case of  $n = 5$ . In the figure,  $h_{ij}$  indicates the approximate location of collocation point  $(x_j, y_i)$ . It is seen that the matrix equation that arises from this numbering for  $n = 5$  is

$$\tilde{M}\tilde{v} = \tilde{k}, \quad (6)$$





it is clear that  $\check{A}$  is block tridiagonal, with the blocks being  $2 \times 2$  matrices. For example, consider

$$\check{A} = A_F = A_2 = \left[ \begin{array}{ccc|cc|cc} a_{22} & a_{23} & -a_{24} & & & & \\ a_{24} & a_{21} & -a_{22} & & & & \\ \hline a_{21} & a_{22} & a_{23} & -a_{24} & & & \\ a_{23} & a_{24} & a_{21} & -a_{22} & & & \\ \hline & & a_{21} & a_{22} & a_{23} & -a_{24} & \\ & & a_{23} & a_{24} & a_{21} & -a_{22} & \\ \hline & & & a_{21} & a_{22} & -a_{24} & \\ & & & a_{23} & a_{24} & -a_{22} & \end{array} \right].$$

We employ a direct block tridiagonal solver to obtain  $\check{\mathbf{v}}_k$ .

The case where  $\check{A} = A$  is just slightly more complicated. Here we see that

$$\check{A} = \begin{bmatrix} A_1 & -A_2 \\ A_1 & A_2 \end{bmatrix},$$

which has the structure

$$\check{A} = \left[ \begin{array}{cccc|cccc} \times & \times & \times & & \times & \times & \times & & \\ \times & \times & \times & & \times & \times & \times & & \\ & \times & \times & \times & \times & \times & \times & \times & \\ & & \times & \times & \times & \times & \times & \times & \\ & & & \times & \times & \times & \times & \times & \\ & & & & \times & \times & \times & \times & \\ & & & & & \times & \times & \times & \\ \hline \times & \times & \times & & \times & \times & \times & & \\ \times & \times & \times & & \times & \times & \times & & \\ & \times & \times & \times & \times & \times & \times & \times & \\ & \times & \times & \times & \times & \times & \times & \times & \\ & & \times & \times & \times & \times & \times & \times & \\ & & & \times & \times & \times & \times & \times & \\ & & & & \times & \times & \times & \times & \\ & & & & & \times & \times & \times & \\ & & & & & & \times & \times & \\ & & & & & & & \times & \times & \times \end{array} \right].$$

Permuting the rows and columns of  $\check{A}$  via a similarity transformation (see [LHHR95]) gives

$$A' = \left[ \begin{array}{cccc|cccc} \times & \times & \times & \times & \times & \times & & & \\ \times & \times & \times & \times & \times & \times & & & \\ \times & \times & \times & \times & \times & \times & & & \\ \times & \times & \times & \times & \times & \times & & & \\ \hline & & \times & \times & \times & \times & \times & \times & \\ & & \times & \times & \times & \times & \times & \times & \\ & & \times & \times & \times & \times & \times & \times & \\ & & \times & \times & \times & \times & \times & \times & \\ \hline & & & & \times & \times & \times & \times & \times & \times \\ & & & & \times & \times & \times & \times & \times & \times \\ & & & & \times & \times & \times & \times & \times & \times \\ & & & & \times & \times & \times & \times & \times & \times \end{array} \right],$$



$$D_R \mathbf{v}_R^{(p+1)} = (1 - \omega) D_R \mathbf{v}_R^{(p)} - \omega M_U \mathbf{v}_B^{(p)} + \omega \mathbf{k}_R \quad (13)$$

and

$$\omega M_L \mathbf{v}_R^{(p+1)} + D_B \mathbf{v}_B^{(p+1)} = (1 - \omega) D_B \mathbf{v}_B^{(p)} + \omega \mathbf{k}_B. \quad (14)$$

We now introduce the vectors

$$\mathbf{z}_c^{(p+1)} = \mathbf{v}_c^{(p+1)} - \mathbf{v}_c^{(p)},$$

where the color subscript  $c = R$  or  $B$ . We also introduce the *color dependent residual vectors*  $\mathbf{r}_c^{(a,b)}$ , defined as

$$\mathbf{r}_R^{(a,b)} = \mathbf{k}_R - \left( D_R \mathbf{v}_R^{(a)} + M_U \mathbf{v}_B^{(b)} \right)$$

and

$$\mathbf{r}_B^{(a,b)} = \mathbf{k}_B - \left( M_L \mathbf{v}_R^{(a)} + D_B \mathbf{v}_B^{(b)} \right),$$

where the superscripts  $(a)$  and  $(b)$  denote iteration level. By considering (11), it is clear that these residual vectors measure how close the approximants  $\mathbf{v}_R^{(a)}$  and  $\mathbf{v}_B^{(b)}$  are to  $\mathbf{v}_R$  and  $\mathbf{v}_B$ , components of the true solution of (11). Algebraically manipulating (13) and (14) and using the notation introduced above yields

$$D_R \mathbf{z}_R^{(p+1)} = \omega \mathbf{r}_R^{(p,p)} \quad (15)$$

and

$$D_B \mathbf{z}_B^{(p+1)} = \omega \mathbf{r}_B^{(p+1,p)}, \quad (16)$$

which are of a form and structure very similar to that of (9). In the SOR algorithm, we compute  $\mathbf{v}^{(p+1)}$  using (15) and (16).

It is clear that we have still maintained a high degree of parallelism by using this red-black SOR scheme. Evidently, all of the red equations in (13) may be solved simultaneously in parallel. Once we have obtained  $\mathbf{v}_R^{(p+1)}$  from (13), we may solve all the black equations in (14) simultaneously in parallel, obtaining  $\mathbf{v}_B^{(p+1)}$ .

Numerical results are illustrated in Figure II. We ran our version of the algorithm for both the Jacobi method and the red-black SOR method using various values of  $\omega$ . We chose  $m = 10$  and chose the boundary conditions and the function  $H(x, y)$  such that  $u = x^3 \sin \pi y$ . Letting

$$\mathbf{r}^{(p,p)} = \left[ \left( \mathbf{r}_R^{(p,p)} \right)^T \quad \left( \mathbf{r}_B^{(p,p)} \right)^T \right]^T, \quad \text{our convergence criterion was that } \|\mathbf{r}^{(p,p)}\|_\infty < 0.001.$$

The Jacobi method needed 151 iterations to converge, which is indicated by an asterisk in the middle of the graph. By comparison, for  $\omega = 1.52$ , the SOR method required only 19 iterations to converge. Indeed, according to the theory in [LHHR95] and [You71], the optimal  $\omega$  for this problem is  $\omega_{\text{opt}} \approx 1.5068$ , which agrees well with our numerical investigation.

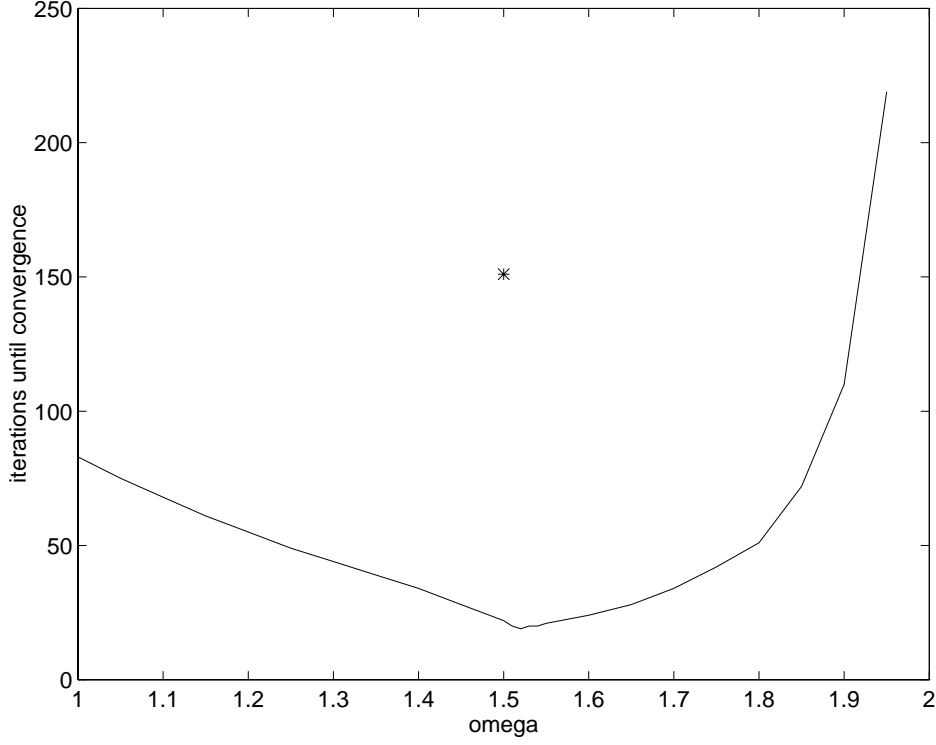
## 6 The Parabolic Equation

We now seek to solve the parabolic equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - H(x, y, t), \quad (17)$$

defined on the interior of  $\mathcal{S}$ , discretized by the collocation method with a uniform mesh, given Dirichlet boundary conditions

$$u(x, y, t) = C(x, y, t), \quad (x, y) \in \partial \mathcal{S}.$$



**Figure II** results using red-black SOR to solve Poisson's equation

We approximate the time derivative by

$$\frac{\partial u}{\partial t} = \frac{u^{(q+1)} - u^{(q)}}{\Delta t}, \quad (18)$$

where the superscript  $(q)$  indicates the value of  $u$  after  $q$  time steps.

Recalling (6), we see that matrix  $\tilde{M}$  was formed by evaluating  $\frac{\partial^2 \hat{u}}{\partial x^2} + \frac{\partial^2 \hat{u}}{\partial y^2}$  at the collocation points. Correspondingly, we form matrix  $\tilde{P}$  by evaluating  $\hat{u}$  at the collocation points. Clearly,  $\tilde{P}$  has precisely the same structure as that of  $\tilde{M}$ . Letting  $p_{ij}$  be the non-trivial entries of  $\tilde{P}$  (just as the  $a_{ij}$ 's in (7) are the non-trivial entries of  $\tilde{M}$ ), we see that the numbers  $p_{ij}$  are given by

$$\begin{array}{cccc} \bar{p}_{11} = 86 + 48\sqrt{3} & \bar{p}_{12} = 13 + 7\sqrt{3} & \bar{p}_{13} = 22 & \bar{p}_{14} = 5 + \sqrt{3} \\ \bar{p}_{21} = 13 + 7\sqrt{3} & \bar{p}_{22} = 2 + \sqrt{3} & \bar{p}_{23} = 5 - \sqrt{3} & \bar{p}_{24} = 1 \\ \bar{p}_{31} = 22 & \bar{p}_{32} = 5 - \sqrt{3} & \bar{p}_{33} = 86 - 48\sqrt{3} & \bar{p}_{34} = 13 - 7\sqrt{3} \\ \bar{p}_{41} = 5 + \sqrt{3} & \bar{p}_{42} = 1 & \bar{p}_{43} = 13 - 7\sqrt{3} & \bar{p}_{44} = 2 - \sqrt{3} \end{array}$$

If we now introduce (18) and the interpolating polynomial (4)<sup>1</sup> into (17) and evaluate the right side of (17) at the collocation (Gauss) points at time  $\theta t^{(q+1)} + (1 - \theta)t^{(q)}$ , where

<sup>1</sup>The interpolating polynomial (4) and forcing function now have time dependence, i.e.  $u_{qr}$ ,  $u_{qr}^x$ ,  $u_{qr}^y$ ,  $u_{qr}^{xy}$ , and  $H$  are now functions also of  $t$ .

$0 \leq \theta \leq 1$ , then we obtain the matrix form of the collocation discretization of the parabolic PDE:

$$\frac{\tilde{P}\tilde{\mathbf{v}}^{(q+1)} - \tilde{P}\tilde{\mathbf{v}}^{(q)}}{\Delta t} = \theta [\tilde{M}\tilde{\mathbf{v}}^{(q+1)} - \tilde{\mathbf{k}}^{(q+1)}] + (1 - \theta) [\tilde{M}\tilde{\mathbf{v}}^{(q)} - \tilde{\mathbf{k}}^{(q)}]. \quad (19)$$

Letting  $\tau = \theta \Delta t$  and  $\bar{\tau} = (1 - \theta) \Delta t$ , we may express (19) as

$$(\tilde{P} - \tau\tilde{M})\tilde{\mathbf{v}}^{(q+1)} = (\tilde{P} + \bar{\tau}\tilde{M})\tilde{\mathbf{v}}^{(q)} - (\tau\tilde{\mathbf{k}}^{(q+1)} + \bar{\tau}\tilde{\mathbf{k}}^{(q)}). \quad (20)$$

In examining (20), we see that this equation defines how we may move from time step ( $q$ ) to time step ( $q + 1$ ). In particular, at time step ( $q$ ), all the vectors on the right side of (20) contain known values. Letting  $\tilde{Q} = (\tilde{P} - \tau\tilde{M})$  and  $\tilde{\mathbf{b}}^{(q)}$  be the right side of (20), we may write (20) as

$$\tilde{Q}\tilde{\mathbf{v}}^{(q+1)} = \tilde{\mathbf{b}}^{(q)}, \quad (21)$$

which is of a form and structure identical to those of (6). We may therefore apply to (21) the block red-black SOR algorithm that we developed for (6). That is, at each time step in (21) we iterate to convergence using block red-black SOR.

## 7 Eigenvalues and Results

Using the work in [LHHR95] as a guide, we determined the eigenvalues of the block Jacobi matrix one would use to solve (21). These eigenvalues may be computed using the following recipe:

$$\begin{aligned} \theta_k &= \frac{k\pi}{m} \\ c_k &= \cos \theta_k \\ r_k &= \sqrt{43 + 40c_k - 2c_k^2} \\ \alpha_k^\pm &= \frac{(3 - \sqrt{3})[(7 - c_k)\gamma^2 + 24(8 + c_k)\gamma - 288\sqrt{3}(3\sqrt{3} \pm r_k)]}{(3 + \sqrt{3})(7 - c_k)\gamma^2 + 24(45 + 29\sqrt{3} - 2\sqrt{3}c_k)\gamma + 1728(10 + 6\sqrt{3} - c_k)} \\ \beta_k^\pm &= \frac{(19 - 9\sqrt{3})[11(7 - c_k)\gamma^2 + 24(4 + 23c_k)\gamma + 864(-37 - 8c_k \pm 3\sqrt{3}r_k)]}{(169 + 69\sqrt{3})(7 - c_k)\gamma^2 + 24(1247 + 993\sqrt{3} + 184c_k + 6\sqrt{3}c_k)\gamma + 1728(122 + 54\sqrt{3} - 59c_k)} \end{aligned}$$

where  $k = 1, \dots, m - 1$  and  $\gamma = \frac{h^2}{\tau}$ . Then form the sets

$$\{\lambda_1, \lambda_2, \dots, \lambda_{2m}\} = \left\{ \frac{(3 - \sqrt{3})\gamma + 24(3 - 2\sqrt{3})}{(3 + \sqrt{3})\gamma + 24(3 + 2\sqrt{3})}, \frac{(\sqrt{3} - 1)\gamma + 24(2\sqrt{3} - 3)}{(\sqrt{3} + 1)\gamma + 24(2\sqrt{3} + 3)}, \alpha_1^+, \alpha_1^-, \dots, \alpha_{m-1}^+, \alpha_{m-1}^- \right\}$$

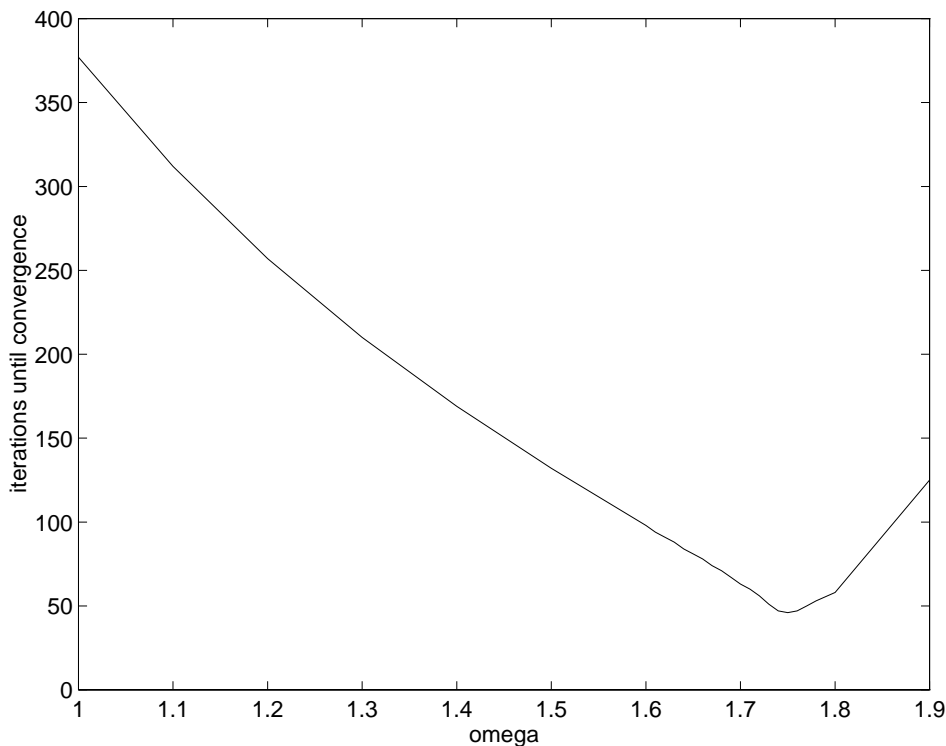
and

$$\{\bar{\lambda}_1, \bar{\lambda}_2, \dots, \bar{\lambda}_{2m}\} = \left\{ \frac{(9 - 4\sqrt{3})\gamma + 12(9 - 7\sqrt{3})}{(9 + 4\sqrt{3})\gamma + 12(9 + 7\sqrt{3})}, \frac{(-4 + 3\sqrt{3})\gamma + 36(-5 + 3\sqrt{3})}{(4 + 3\sqrt{3})\gamma + 36(5 + 3\sqrt{3})}, \beta_1^-, \beta_1^+, \dots, \beta_{m-1}^-, \beta_{m-1}^+ \right\}.$$

Now let  $\phi_{jk} = (\lambda_j - \bar{\lambda}_j) c_k$  for  $k = 1, \dots, m - 1$  and  $j = 1, \dots, 2m$ . Then  $\sigma(J)$ , the set of eigenvalues of the block Jacobi matrix, is (cf. [LHHR95])

$$\begin{aligned} \sigma(J) &= \{\mu : \mu = \pm\lambda_j, j = 1, \dots, 2m\} \\ &\cup \left\{ \mu : \mu = \frac{1}{2} \left( \phi_{jk} \pm \sqrt{\phi_{jk}^2 + 4\lambda_j \bar{\lambda}_j} \right), j = 1, \dots, 2m, k = 1, \dots, m - 1 \right\}. \end{aligned}$$

Given this recipe for the computation of eigenvalues of the Jacobi matrix,<sup>2</sup> one can use the theory in [LHHR95] and [You71] to compute  $\omega_{\text{opt}}$  for the optimal block SOR method.



**Figure III** results using red-black SOR to solve the model parabolic equation

For an example, we ran both the block Jacobi method and block SOR method for various values of  $\omega$  on the parabolic problem. The boundary conditions and function  $H(x, y, t)$  were chosen such that  $u = x^2 y^3 (1 + e^{-t})$ . We chose  $m = 32$ ,  $\theta = \frac{1}{2}$  and let the code run over one time step, from  $t = 0$  to  $t = \Delta t = 0.1$ . The convergence criterion was that the infinity norm of the residual vector had to be less than  $10^{-5}$ . For the Jacobi method, 747 iterations were required for convergence. The number of iterations needed for convergence of the SOR method is illustrated in Figure III for various values of  $\omega$ . The value of  $\omega$  that gave us the fewest number of iterations (namely 46 iterations) was  $\omega = 1.75$ . This agrees well with the value of  $\omega_{\text{opt}}$  given by the theory, namely  $\omega_{\text{opt}} \approx 1.7384$ .

## 8 Summary

Given the work of Lai *et al.*, we developed herein a fast and parallelizable SOR method for the numerical solution of Poisson's equation on the unit square with uniform mesh and Dirichlet boundary conditions. We then extended these techniques to the numerical solution

<sup>2</sup> It can also be shown that all these eigenvalues must have modulus less than unity, irrespective of the value of  $\gamma$ . Thus, the Jacobi method for the model parabolic problem must converge for any  $\gamma$ .

of a model parabolic equation. Our numerical results agree with our analytic results, showing that using our block red-block SOR method on the parabolic equation with appropriately chosen relaxation factor  $\omega$  gives much faster results than does the block Jacobi method.



# References

- [Cel83] Celia M. A. (1983) *Collocation on Deformed Finite Elements and Alternating Direction Collocation Methods*. PhD thesis, Princeton University.
- [LHHR95] Lai Y.-L., Hadjidimos A., Houstis E. N., and Rice J. R. (1995) On the Iterative Solution of Hermite Collocation Equations. *SIAM J. Matrix Anal. Appl.* 16: 254–277. (Also Technical Report, Purdue University, 1992).
- [Pap83] Papatheodorou T. S. (1983) Block AOR Iteration for Nonsymmetric Matrices. *Math. Comp* 41: 511–525.
- [Pic94] Piccirilli D. T. (1994) Using the Collocation Method with Splines under Tension and Upstream Weighting to Solve the One-Dimensional Convection-Diffusion Equation. Master's thesis, University of Vermont.
- [You71] Young D. M. (1971) *Iterative Solution of Large Linear Systems*. Academic Press, New York.