The role of preconditioning in the solution of evolutionary partial differential equations by implicit Fourier pseudospectral methods

L.S. Mulholland * and D.M. Sloan

Department of Mathematics, University of Strathclyde, Glasgow, United Kingdom

Received 28 December 1989
Revised 20 June 1991

Abstract


This paper examines the role of preconditioning in the solution of time-dependent partial differential equations by implicit Fourier pseudospectral methods. Both diffusive and dispersive systems are considered and it is shown in the linear analyses that preconditioning techniques can substantially improve the convergence properties of iterative methods. Computational results indicate that a similar improvement can also be achieved in the solution of model nonlinear problems. This paper also provides clarification on the use of a “staggered grid” technique for dispersive problems. The technique has been outlined in the literature, but not applied to dispersive equations.

Keywords: Preconditioning, pseudospectral methods.

1. Introduction

There are many partial differential equations describing real phenomena which support differing numbers of conservation properties; the Korteweg–de Vries equation (KdV), for example, satisfies an infinite number of such properties. Fully-explicit difference schemes are limited by their inability to preserve the conservation conditions of the equations they are approximating, an inability which often results in an eventual blow-up of solution when the integration is carried over a long period of time. It is therefore necessary, in these instances, to consider implicit schemes which are able to preserve at least some of the conservative...
properties of the corresponding partial differential equations. In turn, consideration of such
implicit schemes requires a knowledge of efficient iterative methods for their solution.

There has been extensive research into iterative methods for solving linear systems. A good
overview of much of this work is provided in [4]. Many of the analyses on this subject arc for
symmetric, positive-definite systems; unfortunately, this is not usually applicable to spectral
equations which are used to attain high spatial accuracy. Although the unsymmetric spectral
system
\[ AU = F \]  \hspace{1cm} (1.1.1)
can be transformed into a positive-definite system
\[ A^T A U = A^T F, \]  \hspace{1cm} (1.1.2)
it is most likely that an effective alternative to this approach can be found. This is due to the
fact that the condition number of \( A^T A \) is the square of the already large condition number of
\( A \), and that in the normal equation (1.1.2) the operator \( A \) must be applied twice.

In this study we shall consider only the stationary Richardson method [7] for solving the
linear system (1.1.1). Given an initial guess \( U^0 \) to \( U \), the sequence \( \{U^k\}_{k=0}^\infty \) is generated by
\[ U^{k+1} = U^k + \omega (F - AU^k), \]  \hspace{1cm} (1.1.3)
where \( \omega \) is a relaxation parameter. The sequence \( \{U^k\} \) converges to \( U \) provided
\[ |1 - \omega \lambda| < 1 \]
for all eigenvalues \( \lambda \) of \( A \). If all the eigenvalues of \( A \) are real, then the optimal relaxation
parameter is
\[ \omega_{\text{opt}} = \frac{2}{\lambda_{\text{max}} + \lambda_{\text{min}}}, \]  \hspace{1cm} (1.1.4)
which provides the spectral radius of the iteration matrix
\[ \rho = \frac{\lambda_{\text{max}} - \lambda_{\text{min}}}{\lambda_{\text{max}} + \lambda_{\text{min}}}. \]

The main problem with the basic Richardson method, when applied to pseudospectral
approximation, is that the value of \( \rho \) is very close to unity and the rate of convergence is thus
extremely slow. In many cases it is possible to overcome this difficulty by "preconditioning" the
problem (1.1.1); we will discuss this technique further in the next section. An extensive
investigation into preconditioning techniques for finite-difference and finite-element methods
is given in [3], and in [5] a form of preconditioning is proposed for spectral methods. A brief
overview of, and further references for, a variety of preconditioning techniques for spectral
methods is given in [1].

So far we have mentioned only the iterative solution of linear systems, but the same
techniques may be incorporated into Newton-iterative methods for the solution of systems of
nonlinear equations. A variety of Newton-iterative methods is considered in [6,8]. Sherman [8]
discovered that, for certain nonlinear systems, the Newton–Richardson (NR) methods he
presented had certain computational advantages over other Newton-iterative methods. This
paper examines the effectiveness of preconditioning an NR method for solving both Burgers’
equation and the KdV.
The aims of this paper are twofold. Firstly, we wish to highlight the difficulties which arise when the ideas of preconditioning, usually built up around diffusive models, are transferred to dispersive systems. Secondly, we wish to show that preconditioning is useful for the solution of nonlinear diffusive and dispersive systems. The difficulties in applications to dispersive systems are largely due to the fact that, rather than reducing the large real eigenvalues of a diffusive linear system, we are now preconditioning to damp down the large imaginary parts of complex eigenvalues without unduly affecting the corresponding real parts — a task which requires the formation of additional techniques.

2. Diffusion equations

2.1. Linear Burgers’ equation

The first aim of this paper is to highlight the difficulties which arise when preconditioning ideas are transferred from diffusive to dispersive models. Accordingly, we introduce preconditioning by referring to the linear diffusive system

\[ u_t + au_x - u_{xx} = 0, \quad u(x + 2\pi, t) = u(x, t), \quad u(x, 0) = u_0(x), \]  

where \( u = u(x, t) \) with \( (x, t) \in \mathbb{R} \times [0, T] \). To solve (2.1.1) by a Fourier pseudospectral method, the interval \([0, 2\pi]\) is discretised by \( N + 1 \) equidistant points with spacing \( \Delta x = 2\pi/N \), and \( u(\cdot, t) \) is approximated by \( \hat{U}(\cdot, t) \in \mathbb{R}^N \), which has the value \( U(x_j, t) \) at \( x = x_j = j \Delta x \) for \( j = 0, 1, \ldots, N - 1 \). If \( N \) is assumed to be even, with \( M = \frac{N}{2} \), the vector \( U(\cdot, t) \) is transformed to discrete Fourier space by

\[ \hat{U}(p, t) = \mathcal{F}U(\cdot, t)(p) = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} U(x_j, t) e^{-2\pi ipj/N}, \]  

for \( p = -M, \ -M + 1, \ldots, M - 1 \). The inversion formula for the discrete transform (2.1.2) is

\[ U(x_j, t) = \left( \mathcal{F}^{-1}\hat{U}(\cdot, t) \right)(x_j) = \frac{1}{\sqrt{N}} \sum_{p=-M}^{M-1} \hat{U}(p, t) e^{2\pi ipj/N}, \]  

for \( j = 0, 1, \ldots, N - 1 \).

Now we approximate the \( r \)th derivative of \( u \) with respect to \( x \) at \( (x_j, t) \) by

\[ \left( \mathcal{F}^{-1}\hat{U}(\cdot, t) \right)(x_j), \quad \text{where} \quad \hat{V}(p, t) = (ip)^r \hat{U}(p, t). \]

Throughout this paper, the fully-discrete approximations will result from a Crank–Nicolson treatment of the semi-discrete pseudospectral equations. Hence, the fully-discrete analogue of (2.1.1) has the form

\[ U^{n+1} - U^n + \frac{1}{2} \Delta t \left( \frac{U^{n+1}_{x} + U^n_{x}}{2} \right) = \frac{1}{2} \Delta t \left( \frac{U^{n+1}_{xx} + U^n_{xx}}{2} \right), \]

where \( U^n = U(\cdot, n \Delta t) \) and \( U^{n+1}, U^n_x \) are the vectors approximating first and second derivatives at \( t = n \Delta t \). We may rewrite the fully-discrete equation in the form

\[ \left[ C^{(0)} + \frac{1}{2} \Delta t \left( aC^{(1)} - C^{(2)} \right) \right] U^{n+1} = \left[ C^{(0)} - \frac{1}{2} \Delta t \left( aC^{(1)} - C^{(2)} \right) \right] U^n, \]

or

\[ AU^{n+1} = F^n. \]
Here $C^{(0)}$ is the identity matrix and $C^{(1)}$ and $C^{(2)}$ are Fourier pseudospectral differentiation matrices, such that $C^{(1)}U^n = U_x^n$ and $C^{(2)}U^n = U_{xx}^n$. The elements of $C^{(1)}$ are given by

$$C_{ij}^{(1)} = \begin{cases} \frac{1}{2}(-1)^{i-j} \cot \left( \frac{i-j}{N} \right), & i \neq j, \\ 0, & i = j, \end{cases} \quad (2.1.6)$$

and those of $C^{(2)}$ are given by

$$C_{ij}^{(2)} = \begin{cases} -\frac{1}{2}(-1)^{i-j} \csc \left( \frac{i-j}{N} \right), & i \neq j, \\ -\frac{1}{6}(1 + 2M^2), & i = j. \end{cases} \quad (2.1.7)$$

The eigenvalues of $A$ are $\lambda_p = 1 + \frac{1}{2} \Delta t (a \pi + p^2)$ for $p = -M, -M + 1, \ldots, M - 1$. Since $A$ is independent of $n$, it would normally be advisable to consider direct factorisation methods. However, we shall consider iterative methods to illustrate the approach which might be used for the nonlinear analogue of (2.1.1). The iterative solution of (2.1.5) by preconditioned methods is simplified if $A$ has real eigenvalues and we therefore simplify the presentation by assuming $a = 0$. Equation (2.1.1) is then reduced to the simple heat conduction equation $u_t = u_{xx}$.

A straightforward Richardson method will only have a good rate of convergence if we impose the time-step restriction $\Delta t = O(1/N^2)$; we shall consider this restriction to be an unsatisfactory one. Consequently, a Richardson method will be inefficient owing to the rapid increase of the spectral condition number $\lambda_{max}/\lambda_{min}$ of $A$ as $N$ increases. The efficiency may be improved if the linear algebra problem is preconditoned. Instead of solving (2.1.9), we seek a solution of

$$H^{-1}AU^{n+1} = H^{-1}U^n,$$

where $H$ is a preconditioning matrix having the properties:

(i) $H$ is easily inverted,

(ii) $H^{-1}$ is a good approximation of $A^{-1}$,

(iii) $H^{-1}A$ has a small spectral condition number.

The preconditioned linear system may be solved by Richardson's method having a relaxation parameter $\omega$ with optimal value

$$\omega_{opt} = \frac{2}{\lambda_{max} + \lambda_{min}},$$

where $\Lambda$ denotes an eigenvalue of $H^{-1}A$. The effect of a well-chosen matrix $H$ is to ensure that the eigenvalues of $H^{-1}A$ are strictly positive and less than some small number, independent of $N$. This, in turn, ensures that the spectral condition number is small and that the spectral radius of the iteration matrix is significantly smaller than unity.

### 2.1.1. A single diagonal preconditioner

The main diagonal of $A$ is constant and given by $A_{jj} = 1 + \frac{1}{12}(2M^2 + 1) \Delta t$ for $j = 0, 1, \ldots, N - 1$. Define the matrix under consideration by

$$H_{ij} = \begin{cases} 0, & i \neq j, \\ A_{jj}, & i = j. \end{cases} \quad (2.1.10)$$
The eigenvalues of $H^{-1}A$ are then

$$
\lambda_p^{(0)} = \frac{12 + 6 \Delta t \Delta t_p^2}{12 + \Delta t (2M^2 + 1)}, \quad \text{for } p = -M, -M + 1, \ldots, M - 1,
$$

and from this we have that

$$
\lambda_{\text{max}}^{(0)} = \frac{12 + 6M^2 \Delta t}{12 + (2M^2 + 1) \Delta t} < 3 \quad \text{and} \quad \lambda_{\text{min}}^{(0)} = \frac{12}{12 + (2M^2 + 1) \Delta t}.
$$

The main difficulty with this simplest form of preconditioning is that the spectral condition number is very large unless $\Delta t = O(\Delta x^2)$. Hence, in order to achieve a good rate of convergence, a time-step restriction must be imposed which is precisely the restriction we are trying to avoid by introducing a preconditioner.

2.1.2. A second-order finite-difference approximation to $A$

Define $H^{(2)}$ using second-order finite-difference approximations as follows:

$$
\left[ H^{(2)}U^n \right](j) = U^n_j - \epsilon(U^n_{j+1} - 2U^n_j + U^n_{j-1}), \quad \text{for } j = 0, 1, \ldots, N - 1,
$$

where $\epsilon = \Delta t/(2(\Delta x)^2)$. Using this as a preconditioner, the effective eigenvalues are given by

$$
(1 + \frac{1}{2} \Delta t \Delta t^2) = \lambda_p^{(2)}\{-\epsilon e^{ip\Delta x} + (1 + 2\epsilon) - \epsilon e^{-ip\Delta x}\},
$$

for $p = -M, -M + 1, \ldots, M - 1$.

This yields

$$
\lambda_p^{(2)} = \frac{2(\Delta x)^2 + \Delta t (p \Delta x)^2}{2(\Delta x)^2 + 2 \Delta t (1 - \cos(p \Delta x))}, \quad \text{for } p = -M, -M + 1, \ldots, M - 1,
$$

from which can be shown that

$$
\lambda_{\text{max}}^{(2)} = \frac{2(\Delta x)^2 + \Delta t \pi^2}{2(\Delta x)^2 + 4 \Delta t} < \frac{1}{4} \pi^2 \quad \text{and} \quad \lambda_{\text{min}}^{(2)} = 1.
$$

Thus, (2.1.9) provides a very good preconditioner which has all the properties (2.1.8). The matrix $H^{(2)}$ is constant and cyclic-tridiagonal: its inversion can therefore be achieved by an initial special LU factorisation followed by a double-sweep substitution at each iteration.

2.1.3. A fourth-order finite-difference approximation to $A$

Consider $H^{(4)}$ defined by

$$
\left[ H^{(4)}U^n \right](j) = U^n_j + \frac{\Delta t}{24(\Delta x)^2}(U^n_{j+2} - 16U^n_{j+1} + 30U^n_j - 16U^n_{j-1} - U^n_{j-2}),
$$

for $j = 0, 1, \ldots, N - 1$.

The effective eigenvalues are now

$$
\lambda_p^{(4)} = \frac{6(\Delta x)^2 + 3 \Delta t (p \Delta x)^2}{6(\Delta x)^2 + \Delta t [(\cos(p \Delta x))^2 - 9]}, \quad \text{for } p = -M, -M + 1, \ldots, M - 1.
$$
It is readily shown that $H^{(4)}$ has properties fairly similar to those of $H^{(2)}$. One disadvantage of $H^{(4)}$, however, is that it is pentadiagonal whereas $H^{(2)}$ is tridiagonal. For this reason we favour $H^{(2)}$ rather than $H^{(4)}$.

2.1.4. Effect of preconditioning

Equation (2.1.1), with $a = 0$ and $u_0(x) = \cos x$, was solved using the scheme (2.1.4). At each time-step (2.1.5) was solved using a Richardson iteration preconditioned by means of (2.1.9). Even at small values of $N$ the advantages offered by preconditioning are significant. For example, with $N = 16$ and $\Delta t = 0.1$ a solution was obtained at $T = 4.0$ which had a maximum pointwise error of $1.5 \cdot 10^{-3}$. The ratio of unpreconditioned to preconditioned CPU-times required to obtain this solution was 8.3. The number of iterations per time-step, maximised over all steps, was reduced from 246 to 17 by the preconditioning; here, iterations were continued until point-wise changes were less than $1 \cdot 10^{-6}$. Detailed results are not presented since they are very much in line with theoretical expectations for this simple linear model.

The second-order finite-difference preconditioner also proves effective for (2.1.1) when $a \neq 0$. In this case $H$ is defined by

$$[H U^n](j) = U^n_j + \frac{a \Delta t}{4 \Delta x} (U^n_{j+1} - U^n_{j-1}) - \epsilon (U^n_{j+1} - 2 U^n_j + U^n_{j-1}), \quad (2.1.11)$$

where $\epsilon = \Delta t/(2(\Delta x)^2)$. The eigenvalues of the preconditioned operator are

$$\Lambda_p = \frac{2(\Delta x)^2 + \Delta t \left(p \Delta x\right)^2 + ia \Delta t \Delta x \left(p \Delta x\right)}{2(\Delta x)^2 + \Delta t \left[2 + ia \Delta x \sin(p \Delta x) - 2 \cos(p \Delta x)\right]},$$

for $p = -M, -M + 1, \ldots, M - 1$. If $a$ is not too large, the real part of $\Lambda_p$ is very close in value to $\Lambda_p^{(2)}$ given in Section 2.1.2. It is found that $H$ defined by (2.1.11) provides a good preconditioner for the system (2.1.4). Numerical results are not presented since the ideas are incorporated in the nonlinear Burgers' equation in the following section, and there, the effect of preconditioning is demonstrated.

2.2. Burgers' equation

Here we consider a nonlinear diffusive model with the differential equation in (2.1.1) replaced by

$$u_t + uu_x - \nu u_{xx} = 0, \quad (2.2.1)$$

where $\nu > 0$ is the coefficient of viscosity. This is the simplest equation combining both nonlinear and diffusive effects.

If we denote the approximations to $u$, $u_t$, $u_x$ and $u_{xx}$ at $(x_i, t)$ by $U(x_i, t)$, $U_t(x_i, t)$, $U_x(x_i, t)$ and $U_{xx}(x_i, t)$, respectively, the semi-discrete Fourier-pseudospectral equations are

$$U_t(x_i, t) + \frac{1}{2} \theta W(x_i, t) + (1 - \theta) U(x_i, t) U_x(x_i, t) - \nu U_{xx}(x_i, t) = 0, \quad (2.2.2)$$

where $W(x_i, t) = |U(x_i, t)|^2$ and $\theta \in [0, 1]$. It is readily shown that the $2\pi$-periodic solution of (2.2.1) satisfies the condition $\frac{d}{dt} \|u(\cdot, t)\|^2 \leq 0$, where $\|u(\cdot, t)\|^2 = \int_0^{2\pi} u^2(s, t) \, ds$. The real parameter introduced in (2.2.2) gives an extra degree of freedom which may be used to achieve an analogous condition on the norm of the semi-discrete solution.
The simplest course to take is to treat the nonlinear term explicitly so that, in the fully-discrete case, we are effectively solving the quasi-linear system $A U^{n+1} = F^n$. This system may be solved as in Section 2.1 using the preconditioner defined by (2.1.9).

### 2.2.1. The fully-discrete equations

If the nonlinear terms in (2.2.2) are treated implicitly, and if \( \theta \) is conveniently chosen to be unity, then the fully-discrete equations are

\[
F_i = U_i^{n+1} - U_i^n + \frac{1}{8} \Delta t \left[ (U_i^n + U_i^{n+1})^2 \right]_x - \frac{1}{2} \nu \Delta t \left[ U_i^n + U_i^{n+1} \right]_{xx} = 0, \quad (2.2.3)
\]

for \( i = 0, 1, \ldots, N - 1 \).

In (2.2.3) \( U_i^n \) denotes the approximation to \( U(x_i, t) \) at \( t = n \Delta t \) and a term such as \( [(U_i^n + U_i^{n+1})^2]_x \) denotes the first pseudospectral derivative formed from a vector with components \( (U_i^n + U_i^{n+1})^2 \) for \( i = 0, 1, \ldots, N - 1 \). Here \( \theta \) has been set to unity to simplify the treatment, but it should be emphasised that the full advantages of the implicit treatment can only be achieved if \( \theta \) is chosen such that \( \| U^{n+1} \| < \| U^n \| \). For the remainder of Section 2 it will be convenient for us to use the notation \( V_i \equiv U_i^{n+1}, U_i \equiv U_i^n, i = 0, 1, \ldots, N - 1 \), and \( V \equiv U^{n+1}, U \equiv U^n \).

For this nonlinear case, the system is solved at each time-step by a sequence of Newton-like iterations; each Newton iteration, in turn, is a linear system to be solved iteratively by Richardson’s method; it is these inner iterations that are preconditioned. The system to be solved by the inner iterations is therefore of the form

\[
\mathcal{F}_i^k = G. \quad (2.2.4)
\]

where \( \mathcal{F}_{ij} = (\partial / \partial V^k)F_i(U, V^k), G = \mathcal{F}(U, V^k)V^k - F(U, V^k) \) and the superscript \( k \) on \( V \) or \( V_i \) indicates the \( k \)th Newton approximations.

Methods which solve each Newton step (2.2.4) iteratively are known as Newton-iterative methods; where (2.2.4) is solved using Richardson’s method we have a Newton–Richardson method; if \( \mathcal{F} \) is evaluated only on \( V^0 \), a crude first approximation to \( V \), we have a quasi-Newton–Richardson method (QNR).

Substituting (2.2.3) into (2.2.4) we obtain

\[
\mathcal{F}_{ij} = \begin{cases} \frac{1}{8} \Delta t (U_i + V_j)C_{ij}^{(1)} - \frac{1}{2} \nu \Delta t C_{ij}^{(2)}, & i \neq j, \\ 1 - \frac{1}{2} \nu \Delta t C_{ij}^{(2)} = 1 + \frac{1}{12} \nu \Delta t (2M^2 + 1), & i = j. \end{cases} \quad (2.2.5)
\]

The matrix product \( \mathcal{F}_i^k \) in (2.2.4) may be written as

\[
\mathcal{F}_i^k = \frac{1}{4} \Delta t \left[ (U_i + V_j)^2 \right]_x - \frac{1}{2} \nu \Delta t \bar{\xi}_{xx} + \bar{\xi}, \quad (2.2.6)
\]

where \( [(U_i + V_j)^2]_x \) denotes the vector with components

\[
\sum_{j=0}^{N-1} C_{ij}^{(1)} [(U_i + V_j)^2], \quad \text{for } i = 0, 1, \ldots, N - 1. \quad (2.2.7)
\]

The product \( \mathcal{F}_i^k \) can be evaluated using a Fast Fourier Transform (FFT) algorithm in \( O(N \log N) \) operations.
A possible QNR algorithm with preconditioning is:

1. \( U := u(x, 0), n := 0, U^n := U; \)
2. \( V^0 := U + \Delta t \left( -\frac{1}{2}U_x^2 + \nu U_{xx} \right), \)
   \( V := W := V^0, \)
   form and factorise \( H(U, V^0); \)
3. calculate \( F(U, V) \) using (2.2.3),
   \( Y := -F, \)
   goto (5);
4. \( Y := \mathcal{J}(U, V^0)(V - W) - F; \)
5. \( W := W + \omega H^{-1}Y; \)
6. if \( \| \omega H^{-1}Y \| > \text{tolerance} \) goto (4);
7. if \( \| V - W \| > \text{tolerance} \) then \( (V := W, \text{goto } (3)); \)
8. \( U^{n+1} := W, n := n + 1, U := W; \)
9. if \( n \Delta t < T \) goto (2).

A few points concerning this algorithm and worthy of note are:

(i) Step (2) includes an Euler first approximation for the Newton iterations; both the Jacobian and its preconditioner \( H \) are evaluated using this approximation and are kept throughout the time-step.
(ii) At step (3), \( V = W \) and so the full evaluation of step (4) is unnecessary for the first Richardson iteration.
(iii) \( V^0, F \), and \( \mathcal{J}(V - W) \) may all be evaluated using an FFT algorithm.
(iv) In a full Newton-iterative method, \( H \) would be formed and factorised at step (3) and \( \mathcal{J} = \mathcal{J}(U, V). \)

2.2.2. Possible preconditioners

Here we will consider two types of preconditioner that may be used in (2.2.8). The first type is a simple pick-up of the main diagonal and first few sub- and superdiagonals of the main Jacobian (2.2.5): that is,

\[
H_{ij} = \begin{cases} 
\mathcal{J}_{ij}, & |i - j| \leq l, \\
0, & |i - j| > l.
\end{cases}
\]  

(where \( l \in \{1, 2\}. \)

The second type of preconditioner is obtained by approximating \( F, \) from (2.2.3), using second- or fourth-order finite-differences; this approximation then provides, on differentiating, an approximation to the Jacobian. Each of these preconditioners will be either cyclic-tridiagonal or cyclic-pentadiagonal and may be factorised using specialised LU-factorisation routines.

2.2.3. A comparison of methods for solving Burgers’ equation

Equation (2.2.1) has been shown in [9] to have the following \( 2\pi \)-periodic exact solution:

\[
u(x, t) = \frac{1}{t + 1} \sum_{r = -\infty}^{\infty} \left[ x - (2r + 1)\pi \right] \exp\left\{ -\left[ x - (2r + 1)\pi \right]^2/(4\nu(t + 1)) \right\} \]

\[
\sum_{r = -\infty}^{\infty} \exp\left\{ -\left[ x - (2r + 1)\pi \right]^2/(4\nu(t + 1)) \right\}.
\]  

(2.2.10)
Presented here are the numerical results obtained using seven methods for finding a solution approximating (2.2.10); the first five of these methods are based on the algorithm (2.2.8). The seven methods are as follows.

(i) UNPNR. This method uses (2.2.8) with $H = I$, the identity matrix, and is therefore an unpreconditioned QNR method.

(ii) TDPuQNR. This method uses (2.2.8) with $H$ given by (2.2.9), $l = 1$; it is a QNR method preconditioned by a tridiagonal pickup of the Jacobian.

(iii) PDPuQNR. As TDPuQNR but with a pentadiagonal pickup of the Jacobian.

(iv) TDFDQNR. This method uses (2.2.8) with $H$ determined by second-order finite-difference approximations to the spatial derivatives.

(v) PDFDQNR. The fourth-order equivalent of TDFDQNR; $H$ is therefore cyclic-pentadiagonal.

(vi) QLPR. This is a quasi-linear method where the nonlinear term is treated explicitly; it thus reduces to a straight Richardson iterative procedure and employs the preconditioner defined by (2.19).

Table 2.1
Comparison of methods for solving Burgers' equation with $\nu = 0.2$ and initial condition from (2.2.10); $L_\infty$ is the maximum pointwise error at $T = 0.4$ and $E$ is normalised CPU-time

<table>
<thead>
<tr>
<th>Method</th>
<th>$\Delta t = 0.1$</th>
<th>$\Delta t = 0.01$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\omega$</td>
<td>$L_\infty$</td>
</tr>
<tr>
<td>UNPNR</td>
<td>0.4</td>
<td>0.036</td>
</tr>
<tr>
<td>TDPuQNR</td>
<td>1.0</td>
<td>0.036</td>
</tr>
<tr>
<td>PDPuQNR</td>
<td>1.1</td>
<td>0.036</td>
</tr>
<tr>
<td>TDFDQNR</td>
<td>0.7</td>
<td>0.036</td>
</tr>
<tr>
<td>PDFDQNR</td>
<td>0.85</td>
<td>0.036</td>
</tr>
<tr>
<td>QLPR</td>
<td>0.5768</td>
<td>0.063</td>
</tr>
<tr>
<td>CKB</td>
<td>-</td>
<td>0.166</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>$\Delta t = 0.1$</th>
<th>$\Delta t = 0.01$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\omega$</td>
<td>$L_\infty$</td>
</tr>
<tr>
<td>UNPNR</td>
<td>0.1</td>
<td>0.0032</td>
</tr>
<tr>
<td>TDPuQNR</td>
<td>0.03</td>
<td>0.0032</td>
</tr>
<tr>
<td>PDPuQNR</td>
<td>1.2</td>
<td>0.0032</td>
</tr>
<tr>
<td>TDFDQNR</td>
<td>0.58</td>
<td>0.0032</td>
</tr>
<tr>
<td>PDFDQNR</td>
<td>0.88</td>
<td>0.0032</td>
</tr>
<tr>
<td>QLPR</td>
<td>0.5768</td>
<td>0.0560</td>
</tr>
<tr>
<td>CKB</td>
<td>-</td>
<td>0.0209</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>$\Delta t = 0.1$</th>
<th>$\Delta t = 0.01$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\omega$</td>
<td>$L_\infty$</td>
</tr>
<tr>
<td>UNPNR</td>
<td>0.04</td>
<td>0.002</td>
</tr>
<tr>
<td>TDPuQNR</td>
<td>&lt; 0.002</td>
<td>0.002</td>
</tr>
<tr>
<td>PDPuQNR</td>
<td>1.2</td>
<td>0.002</td>
</tr>
<tr>
<td>TDFDQNR</td>
<td>0.62</td>
<td>0.002</td>
</tr>
<tr>
<td>PDFDQNR</td>
<td>0.5768</td>
<td>0.053</td>
</tr>
<tr>
<td>QLPR</td>
<td>-</td>
<td>0.013</td>
</tr>
<tr>
<td>CKB</td>
<td>-</td>
<td>0.013</td>
</tr>
</tbody>
</table>
(vii) CKB. This method is taken from Chan and Kerkhoven's [2] treatment of the Korteweg-de Vries equation. The scheme, which is explicit in Fourier space, is defined as follows. Let

$$\hat{U}_p^n = \mathcal{F}[U^n](p), \quad W_j^n = \frac{1}{3}(U_j^n)^2, \quad \hat{W}_p^n = \mathcal{F}[W^n](p);$$

then

$$\hat{U}_p^{n+1} - \hat{U}_p^n = -i2 \Delta t p \hat{W}_p^n - \nu \Delta t p^2 (\hat{U}_p^{n+1} + \hat{U}_p^{n-1}),$$

for \( p = -M, -M + 1, \ldots, M - 1. \)

Since the objective here is to compare the relative merits of various fully-implicit iterative methods, we do not include CKB for genuine comparison; rather, it is included to provide perspective; Burgers' equation is almost certainly more efficiently solved using a very good semi-implicit method such as CKB.

Table 2.1 provides numerical results obtained by applying the seven methods to solve (2.2.1) with \( \nu = 0.2 \) and initial condition given by (2.2.10) with \( \zeta = 0. \) Results comprise of the maximum pointwise error \( L_x \) at \( T = 0.4 \) and the CPU-time taken, \( E, \) normalised to the time of the fastest implicit method; these are listed for three different values of \( N \) and two values of \( \Delta t. \) The presented values of \( \omega \) were found to be optimal by trial and error. The implicit methods all use a fixed tolerance of \( 10^{-8} \) in the iteration.

Table 2.2 shows the relative efficiency of the methods for a given problem. In this case the CPU-time is minimised under the constraint \( L_x < 2.0 \cdot 10^{-3} \) by varying \( N, \Delta t \) and the tolerance.

The results show quite clearly that TDFDCNR is the best implicit method overall for solving the given problem. Although the pentadiagonal preconditioners force a faster convergence of the Richardson iterations, this is offset by the extra computation necessary to invert them compared with the tridiagonal preconditioner. The unpreconditioned method is competitive for small \( N, \) but the rate of convergence diminishes rapidly as \( N \) is increased.

3. Equations involving a third derivative

There has been considerable recent interest on the subject of preconditioning iterative methods, but there has been much less attention directed at spectral preconditioning, and to our knowledge nothing has been written on the problems inherent in finding a good precondi-
tioner for equations involving a third derivative term $u_{xxx}$. It is for this reason that the present study was undertaken, using Section 2 as a motivator for Section 3.

3.1. A steady linear equation

Since we can find no mention in the literature of preconditioners for it, we begin by considering the simple steady equation

$$-\frac{d^3 u}{dx^3} = f(x), \quad u(x + 2\pi) = u(x), \quad x \in \mathbb{R}. \quad (3.1.1)$$

Writing the Fourier pseudospectral approximation to $u(x)$ as

$$V_j = \frac{1}{\sqrt{N}} \sum_{p=-M}^{M-1} \hat{V}_p e^{i p x_j}, \quad \text{for } j = 0, 1, \ldots, N-1,$$

where $x_j = 2\pi j/N$, the discrete equations are

$$\sum_{p=-M}^{M-1} i p^3 \hat{V}_p e^{i p x_j} = f(x_j), \quad \text{for } j = 0, 1, \ldots, N-1,$$

or

$$LV = F. \quad (3.1.2)$$

The preconditioned version of a Richardson iteration is

$$V^{k+1} = V^k + \epsilon H^{-1}(F - LV^k),$$

where $H$ has the properties (2.1.8).

3.1.1. A second-order centred finite-difference preconditioner

To illustrate the extra complexity which a third derivative term introduces we begin by considering the preconditioner given by

$$[HV](j) = -\frac{1}{2(\Delta x)^3} (V_{j+2} - 2V_{j+1} + 2V_{j-1} - V_{j-2}). \quad (3.1.3)$$

The eigenvalues associated with the preconditioned system are then

$$\lambda_p^{(2)} = -\frac{(p \Delta x)^3}{\sin(2p \Delta x) - 2 \sin(p \Delta x)}, \quad \text{for } p = -M, -M+1, \ldots, M-1,$$

which is unbounded for $p = -M$ and is of order $N$ for $|p|$ close to $M$. Therefore, we can only incorporate (3.1.3) if we use a device, first proposed in [5] for a first-order problem, whereby we define

$$\tilde{V}_p = \begin{cases} -i p^3 \hat{V}_p, & |p| \leq \frac{1}{3}N \text{ (say)}, \\ 0, & \frac{1}{3}N < |p| \leq \frac{1}{2}N, \end{cases} \quad (3.1.4)$$
and use the truncated approximation
\[
\frac{d^3 u}{dx^3} \bigg|_{x} = \frac{1}{\sqrt{N}} \sum_{\nu = -M}^{M-1} \hat{V}_\nu e^{ip\pi x}.
\]
For this setup we have \(\Lambda_{\text{max}}^{(2)} = \frac{16}{81} \pi^3 / \sqrt{3}\) and \(\omega_{\text{opt}} = 0.44\), but there will be a loss of accuracy as a result of the truncation described by (3.1.4).

3.1.2. The mid-point collocation technique

We have shown that when a preconditioner uses a centred finite-difference approximation of an odd-order derivative, the resultant eigenvalues become unbounded. When dealing with systems involving odd-order derivatives, therefore, we must look for some general technique which will overcome or avoid this difficulty. One such technique is what we will term “mid-point collocation” (MPC), and what has previously been referred to (cf. [1]) as “using a staggered grid”.

The main idea behind MPC is to replace the usual pseudospectral system by an equally accurate system wherein \(u\) and all its higher derivatives are approximated pseudospectrally at the new collocation points
\[
\tilde{x}_j = x_{j+1/2} = \frac{(2j + 1)\pi}{N}, \quad \text{for} \quad j = 0, 1, \ldots, N-1.
\] (3.1.5)
The \(q\)th derivative, where \(q \geq 0\), of \(u\) at \(x = \tilde{x}_j\) will thus be approximated by
\[
\frac{d^q u}{dx^q} \bigg|_{x_{j+1/2}} = \frac{1}{\sqrt{N}} \sum_{\nu = -M}^{M-1} (i\nu)^q \hat{V}_\nu e^{ip\pi (x_{j+1/2})/N}.
\] (3.1.6)
Now, when constructing a preconditioner based on finite differences, we likewise approximate \(u\) and its derivatives at the new nodes \(\tilde{x}_j\). An important result of this is that a first-order, uncentred, finite-difference approximation of a derivative at \(x = x_j\) becomes, using MPC, a second-order, centred approximation at \(x = \tilde{x}_j\).

One important point to be noted when using MPC is that the right-hand side of the system being solved must also be based on approximations at the new collocation points \(\tilde{x}_j\); otherwise we would have a system inconsistent with the original problem.

Using MPC, a preconditioner based on second-order finite differences is given by
\[
[H V](j) = -\frac{V_{j+2} - 3V_{j+1} + 3V_j - V_{j-1}}{(\Delta x)^3},
\]
and this yields the associated eigenvalues
\[
\Lambda_{\text{MPC}} = \frac{(p \Delta x)^3}{8 \sin^3\left(\frac{1}{2} p \Delta x\right)}, \quad \text{for} \quad p = -M, -M + 1, \ldots, M - 1.
\]
Hence, the eigenvalues here have the properties
\[1 \leq \Lambda_{\text{MPC}} \leq \frac{1}{8} \pi^3 \quad \text{and} \quad \omega_{\text{opt}} = \frac{16}{8 + \pi^3}.
\]
If we attempt to base a preconditioner on a six-term finite-difference approximation, it turns
out that we can again only achieve second-order accuracy. We do, however, have one parameter to help reduce the range of eigenvalues further. For example, we may define $H$ by
\[
[H_{\psi}(j)] = \frac{V_{j+3} - 6V_{j+2} + 13V_{j+1} - 13V_j + 6V_{j-1} - V_{j-2}}{(\Delta x)^3}
\]
to obtain eigenvalues in the range $[0.4, 1.0]$ with corresponding $\omega_{\text{opt}} \approx 1.4$.

3.2. A simple linear dispersive equation

Consider the equation
\[
\begin{align*}
    u_t &= u_{xxx}, \\
    u(x + 2\pi, t) &= u(x, t), \quad \text{for } (x, t) \in \mathbb{R} \times [0, T], \\
    u(x, 0) &= u_0.
\end{align*}
\]
A Crank–Nicolson treatment leads to the fully-discrete approximation
\[
[C(0) - \frac{1}{2} \Delta t C(3)]U_n^{n+1} = [C(0) + \frac{1}{2} \Delta t C(3)]U_n^n,
\]
or
\[
AU_n^{n+1} = F_n,
\]
where $C(3)$ is such that $C(3)U = U_{xxx}$, with elements
\[
C_{i,j}^{(3)} = \begin{cases} 
\frac{1}{4}(-1)^{i-j} \cot \left( \frac{\pi(i-j)}{N} \right) \left( 3 \csc^2 \left( \frac{\pi(i-j)}{N} \right) - 2M^2 \right), & i \neq j, \\
0, & i = j.
\end{cases}
\]
The eigenvalues of $A$ are $\lambda_p = 1 + \frac{1}{2} \Delta t \mu^3$ when the derivatives are evaluated at the usual collocation points $x_j = 2\pi j/N$. If we use MPC, then $\lambda_p^{\text{MPC}} = \lambda_p e^{i\pi\Delta x/2}$.

3.2.1. Mid-point collocation preconditioners

It can be shown that preconditioners based on a finite-difference approximations, either centred or uncentred with respect to the usual collocation points, do not readily satisfy the properties (2.1.8). We therefore restrict our attention here to an examination of mid-point collocation preconditioners. That is, we will consider finite-difference approximations, centred with respect to the collocation points (3.1.5), to provide a preconditioner for the system to be solved using MPC.

A general form of such a such a preconditioner, incorporating an eight-term approximation, is given by
\[
[H_{\psi}(j)] = \frac{1}{2}(U_{j+1}^n + U_j^n) - \frac{\Delta t}{2(\Delta x)^3} \left( aU_{j+4}^n + bU_{j+3}^n + cU_{j+2}^n + dU_{j+1}^n - dU_j^n - cU_{j-1}^n - bU_{j-2}^n - aU_{j-3}^n \right),
\]
for
\[
[ H_{\psi}(j) ] = \frac{1}{2}(U_{j+1}^n + U_j^n) - \frac{\Delta t}{2(\Delta x)^3} \left( aU_{j+4}^n + bU_{j+3}^n + cU_{j+2}^n + dU_{j+1}^n - dU_j^n - cU_{j-1}^n - bU_{j-2}^n - aU_{j-3}^n \right),
\]
(3.2.3)
where \( a, b, c \) and \( d \) satisfy the second-order accuracy constraints
\[
7a + 5b + 3c + d = 0, \quad 91a + 35b + 9c + d = 6.
\] (3.2.4)
The eigenvalues associated with the preconditioned system are
\[
\Lambda_n^{\text{MPC}} = \frac{1 + \frac{1}{2} \Delta t \rho^3}{\cos \psi - i \Delta t/(\Delta x)^3 f(\psi)} \approx \frac{-4\psi^3}{f(\psi)}, \quad \text{for large } |\psi|,
\]
where \( \psi = \frac{1}{\Delta x} \Delta x \) and \( f(\psi) = a \sin 7\psi + b \sin 5\psi + c \sin 3\psi + d \sin \psi \). One possible strategy is to solve \( f(\psi) = -4\psi^3 \) for some large value of \( |\psi| \); for example, \( \psi = \frac{1}{2} \pi \) yields the condition \( a - b + c - d = \frac{1}{2} \pi^3 \), so we should ideally choose \( a, b, c \) and \( d \) to satisfy this condition. We will consider the three possible cases in turn.

(i) \( a = b = 0 \). In this case (3.2.4) completely determines the values of \( c \) and \( d \): \( c = 1 \) and \( d = -3 \). Now we have \( f(\psi) = -4 \sin^3 \psi \) and
\[
1 \leq \Re \left( \Lambda_n^{\text{MPC}} \right) \approx \frac{\psi^3}{\sin^3 \psi} \leq \frac{1}{8} \pi^3, \quad |\Im(\Lambda_n^{\text{MPC}})| \ll 1.
\]
Notice here that \( a - b + c - d = 4 \), which is not close to \( \frac{1}{2} \pi^3 \).

(ii) \( a = 0, b \neq 0 \). Writing \( c \) as a multiple of \( b \), \( c = bx \) (say), (3.2.4) gives us \( a - b + c - d = 4(1 - 4/(x + 5)) \). The ideal value of \( x \) is \(-5 - 32/(\pi^3 - 8)\), but we will choose the nearest integer, \( x = -6 \), for simplicity. This choice yields the values \( a = 0, b = -1, c = 6, d = -13 \), and for these we have
\[
0.437 \leq \Re \left( \Lambda_n^{\text{MPC}} \right) \approx \frac{\psi^3}{\sin^3 \psi(1 + 4 \sin^2 \psi)} \leq 1, \quad |\Im(\Lambda_n^{\text{MPC}})| \ll 1,
\]
and \( a - b + c - d = 20 \), which is fairly close to \( \frac{1}{2} \pi^3 \).

(iii) \( a \neq 0 \). In this case we have one extra parameter to help us restrict the range of eigenvalues. However, the addition computational cost of inverting \( H \) will outweigh any small improvement to be made on the rate of convergence.

We may conclude that (3.2.3), with values of \( a, b, c, d \) as given in case (ii) above, would seem to be the best choice of preconditioner for the system (3.2.2) when MPC is incorporated.

### 3.3. The linearised KdV equation

As a precursor to a full consideration of the KdV equation we look briefly at its linearised form
\[
\begin{align*}
u_t + \mu \nu_x + \epsilon \nu_{xxx} &= 0, \\
v(x + 2\pi, t) &= v(x, t), \\
v(x, 0) &= v_0.
\end{align*}
\]
(3.3.1)
where \( \mu, \epsilon \) are real constants. The analogous fully-discrete equations are
\[
\left[ \mathcal{E}^{(0)} + \frac{1}{2} \Delta t \mathcal{E}^{(1)} + \frac{1}{2} \epsilon \Delta t \mathcal{E}^{(3)} \right] U^{n+1} = \left[ \mathcal{E}^{(0)} - \frac{1}{2} \mu \Delta t \mathcal{E}^{(1)} - \frac{1}{2} \epsilon \Delta t \mathcal{E}^{(3)} \right] U^n,
\]
where \( \mathcal{E} \) refers to approximations using MPC rather than the standard \( C \). In circumstances where the dispersion term dominates the solutions of (3.3.1) we would expect the analysis of
Section 3.2 to lead us to a good preconditioner for the linearised KdV also. Indeed, if we define our preconditioning matrix \( H \) by

\[
[\mathbf{H} \mathbf{U}^n](j) = \frac{1}{2} (U_{j+1}^n + U_j^n) + \frac{\mu}{2\Delta x} (U_{j+1}^n - U_j^n) \\
+ \frac{\epsilon}{(\Delta x)^3} (U_{j+2}^n - 3U_{j+1}^n + 3U_j^n - U_{j-1}^n),
\]

then it is easy to show that, using MPC, the eigenvalues \( \lambda_p^{MPC} \) are such that

\[ 1 < \Re(\lambda_p^{MPC}) < \frac{1}{8} \pi^3 \quad \text{and} \quad \left| \Im(\lambda_p^{MPC}) \right| \ll 1, \]

provided \( \epsilon \) is not too small relative to \( \mu \). This choice of preconditioner approximates the third derivative term as in case (i) of Section 3.2.1.

3.4. The Korteweg–de Vries equation

The simplest equation combining both nonlinearity and dispersion is the KdV equation

\[
\begin{align*}
\frac{\partial u}{\partial t} + \mu \frac{\partial u}{\partial x} + \epsilon \frac{\partial^3 u}{\partial x^3} &= 0, \\
\frac{\partial u(x + 2\pi, t)}{\partial t} &= u(x, t), \\
u(0, t) &= u_0,
\end{align*}
\]

where \((x, t) \in \mathbb{R} \times [0, T] \) and \( \mu, \epsilon \) are real constants.

The analysis in this section follows very closely the analysis of Section 2.2 for Burgers' equation, with the obvious adjustments corresponding to the replacement of the diffusive term by the dispersive term.

3.4.1. The fully-discrete equations

For the remainder of Section 3.4 we will again use the notation \( V = \mathbf{U}^{n+1} \), \( U = \mathbf{U}^n \) and \( V^k = \text{th Newton approximation to } V \).

The \( \theta = 1 \) fully-discrete analogue of (3.4.1) is

\[
\begin{align*}
F_i = V_i - U_i + \frac{\mu}{2\Delta t} \left[ (U_i + V_i)^2 \right]_x + \frac{\epsilon}{2\Delta t} \left[ U_i + V_i \right]_{xxx} = 0, \\
\text{for } i = 0, 1, \ldots, N - 1.
\end{align*}
\]

The Jacobian \( \mathcal{J} \) associated with this scheme has elements

\[
\mathcal{J}_{ij} =\begin{cases}
\frac{\mu}{2\Delta t} (U_i + V_j) \mathcal{E}_i^{(1)} + \frac{\epsilon}{2\Delta t} \mathcal{E}_i^{(3)} + \mathcal{E}_i^{(0)}, & i \neq j, \\
\mathcal{E}_i^{(0)}, & i = j.
\end{cases}
\]

As before, \( \mathcal{J}^\xi \) may be evaluated using the FFT, and the algorithm (2.2.8), with adjusted Euler step, can be used to solve (3.4.2) by the QNR method.

3.4.2. Possible preconditioners

We will consider three preconditioning matrices: one which is the tridiagonal version of the form (2.2.9), and two which are based on finite-difference approximations centred at the collocation points \( \tilde{x}_j \) and incorporating MPC.
When the dispersive term is approximated by a four-term expression, as in Section 3.2.1, case (i), the preconditioning matrix $H^{(4)}$ has elements

$$
H_{i,i-1} = -\kappa,
H_{i,i} = \frac{1}{2} - \rho(U_i + V_i) + 3\kappa,
H_{i,i+1} = \frac{1}{2} + \rho(U_{i+1} + V_{i+1}) - 3\kappa,
H_{i,i+2} = \kappa,
$$

where

$$
\kappa = \frac{\epsilon \Delta t}{2(\Delta x)^3}
\text{ and } \rho = \frac{\mu \Delta t}{4(\Delta x)}.
$$

When the dispersive term is approximated by a six-term expression, as in Section 3.2.1, case (ii), the preconditioning matrix has elements

$$
H_{i,i-2} = \kappa,
H_{i,i-1} = -6\kappa,
H_{i,i} = \frac{1}{2} - \rho(U_i + V_i) + 13\kappa,
H_{i,i+1} = \frac{1}{2} + \rho(U_{i+1} + V_{i+1}) - 13\kappa,
H_{i,i+2} = 6\kappa,
H_{i,i+3} = -\kappa.
$$

3.4.3. Numerical results on using the three preconditioners

When $\mu = 6$ and $\epsilon = 1$, (3.4.1) has the exact 2-soliton solution

$$
u(x, t) = \frac{\partial^2}{\partial x^2} \left[ \ln f(x, t) \right],$$

$$f(x, t) = 1 + e^{\eta_1} + e^{\eta_2} + \left( \frac{b_1 - b_2}{b_1 + b_2} \right) e^{(\eta_1 + \eta_2)},$$

$$\eta_i = \eta_i(x, t) = b_i x - b_i^2 t + \eta^{(0)}_i, \text{ for } i = 1, 2,$$

where $b_i$ and $\eta^{(0)}_i$ are constants for $i = 1, 2$. Soliton $i$, $i = 1, 2$, has amplitude $\frac{1}{2}b_i^2$ and travels with velocity $b_i^2$ from an initial location $x = -\eta^{(0)}_i/b_i$.

Unfortunately, the solution (3.4.6) is not periodic. However, it is approximately periodic in the interval $x \in [-L, L]$ for $L \in \mathbb{R}$ sufficiently large. We therefore transform the spatial variable to $X = \pi(x + L)/L$ giving an approximately $2\pi$-periodic dependent variable $u(X, t)$ which satisfies

$$v_t + \mu v X + \hat{\epsilon} v_{XXX} = 0, \text{ for } (X, t) \in \mathbb{R} \times [0, T],$$

where $\mu = 6\pi/L$ and $\hat{\epsilon} = (\pi/L)^3$.

The test problem considered here is to find the 2-soliton solution of (3.4.7) at $t = 2.0$ given an initial condition obtained from (3.4.6) with $t = 0$ and parameters set as follows:

$$b_1 = 1.0, \quad \eta^{(0)}_1 = 0.0,$$
$$b_2 = \sqrt{2}, \quad \eta^{(0)}_2 = 2\sqrt{2}.$$
Table 3.1
Comparison of preconditioners for (3.4.7)

<table>
<thead>
<tr>
<th>Method</th>
<th>$N$</th>
<th>$\Delta t$</th>
<th>$\omega$</th>
<th>Tolerance</th>
<th>$L_\infty$</th>
<th>$E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNPKDV</td>
<td>64</td>
<td>0.01</td>
<td>1.0</td>
<td>0.001</td>
<td>1.73 $\cdot 10^{-3}$</td>
<td>2.44</td>
</tr>
<tr>
<td>TDPUKDV</td>
<td>64</td>
<td>0.01</td>
<td>1.0</td>
<td>0.001</td>
<td>1.83 $\cdot 10^{-3}$</td>
<td>2.96</td>
</tr>
<tr>
<td>SFDKDV4</td>
<td>64</td>
<td>0.083</td>
<td>0.3</td>
<td>0.00013</td>
<td>1.98 $\cdot 10^{-3}$</td>
<td>2.03</td>
</tr>
<tr>
<td>SFDKDV6</td>
<td>64</td>
<td>0.06</td>
<td>1.0</td>
<td>0.0019</td>
<td>1.81 $\cdot 10^{-3}$</td>
<td>1</td>
</tr>
</tbody>
</table>

The four methods for solving the test problem all use algorithm (2.2.8), though with differing preconditioning matrices. UNPKDV has $H = I$, the identity matrix; TDPUKDV has $H$ given by (2.2.9) with $l = 1$; SFDKDV4 has $H$ given by (3.4.4); and SFDKDV6 has $H$ given by (3.4.5). SFDKDV4 and SFDKDV6 each use mid-point collocation.

For each method the number of grid points, the time-step, the relaxation parameter $\omega$ and the tolerance are all set so as to minimise the computing time necessary under the given accuracy constraint (see Table 3.1). $E$ is a measure of the CPU-time taken by each method normalised to the most efficient method.

The main reasons why we consider the use of preconditioning techniques is that we hope to improve the convergence properties of the iterative method we are employing in those cases where the time-step restriction is due to problems of convergence rather than to accuracy constraints. That is, we would hope that preconditioning will improve convergence properties sufficiently to allow us to use a much larger time-step and still obtain the required accuracy for the given problem.

With this in mind we conclude that, for our given test problem, the preconditioner used in SFDKDV6, when applied in conjunction with the mid-point collocation technique, considerably improves the efficiency of the quasi-Newton–Richardson method studied in this paper.

It is interesting to note that, although the preconditioner employed in SFDKDV4 does allow us to increase the time-step, the rate of convergence is not sufficiently good to make a significant increase in efficiency of the QNR method. This contrast in performance between SFDKDV4 and SFDKDV6 is in close accord with the linear analysis of Section 3.2.1.

4. Conclusions

The solution of linear time-dependent partial differential equations by pseudospectral-iterative methods suffers from the difficulty posed by very large spectral condition numbers of the associated fully-discrete linear systems. We have shown that preconditioning techniques provide a useful tool for substantially reducing the condition number of such systems and thereby significantly increasing the rate of convergence of the iterates, particularly where a large number of grid points is used. It is shown, however, that the introduction of a third derivative term dramatically increases the complexity of analysis required to determine a good preconditioner. We noted that, in such cases, it was very beneficial to introduce the notion of “mid-point collocation” in order to obtain reasonable bounds for the eigenvalues of the particular system. We may conclude that preconditioning is effective in improving the conver...
gence properties of pseudospectral-iterative methods for solving linear partial differential equations.

The case for preconditioning, when solving nonlinear equations by a Newton–Richardson method is less clear-cut. This is because the relative performance of the NR method, employing any given preconditioner, is likely to be problem-dependent. However, we were able to show that, for two given test problems, preconditioners could be found which significantly improved the efficiency of the basic NR method. These results suggest that there is a case to be made for the use of preconditioning techniques when solving nonlinear partial differential equations by some iterative method such as an NR method. In particular, the case can be made when the basic iterative method suffers from time-step restrictions due to problems of convergence, since the idea of preconditioning is to alleviate these restrictions. The case cannot be made, however, when the accuracy constraint is so tight that it imposes a time-step restriction more severe than the convergence of the iterates.

This paper has merely touched upon a small part of the much larger topic of spectral conditioning. We considered only one type of iterative method, only two classes of preconditioners, and only two, one-dimensional, nonlinear partial differential equations. It is clear that much further research must be undertaken in order to establish the relative merits and demerits of preconditioned pseudospectral-iterative methods, when compared to other tried and tested methods, for solving a variety of problems. It is our hope that this work provides some justification for such research.

References