ON THE CYCLIC REDUCTION OF TRIDIAGONAL SYSTEMS

by

Ward J. Shaffer

Submitted in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in the

Mathematics

Program

YOUNGSTOWN STATE UNIVERSITY

August, 2005

ON THE CYCLIC REDUCTION OF TRIDIAGONAL SYSTEMS

Ward J. Shaffer

I hereby release this thesis to the public. I understand that this thesis will be made available from the OhioLINK ETD Center and the Maag Library Circulation Desk for public access. I also authorize the University or other individuals to make copies of the thesis as needed for scholarly research.

Signature:

Ward J. Shaffer, Student — 27 Jul 05 — Date

Approvals:

John J. Buoni, Thesis Advisor — 27 Jul 05 — Date

Richard L. Burden, Committee Member — 7/27/05 — Date

Eric J. Wingler, Committee Member — 7/27/05 — Date

Peter J. Kasvinsky, Dean of Graduate Studies — 8/4/05 — Date

ABSTRACT

Tridiagonal systems arise frequently in applied mathematics. This thesis will investigate the use of complete and truncated cyclic reduction algorithms to solve these systems. We will develop the complete cyclic reduction algorithm for tridiagonal and block tridiagonal systems. We will show the development of the Bondeli-Gander truncated algorithm for tridiagonal systems and will extend it to the case where the off-diagonal elements are not 1. Finally, we will use the work of Heller to create a truncated algorithm for block tridiagonal systems and will discuss the work necessary to extend the Bondeli-Gander algorithm to block tridiagonal systems.

ACKNOWLEDGEMENT

The author would like to acknowledge the assistance of Dr. John J. Buoni of Youngstown State University. Without his patience, assistance, and prodding, this thesis would not exist. Also, Drs. Richard Burden and Eric Wingler provided many useful comments and questions concerning drafts of this thesis, and their contributions are gratefully admitted. Dr. Doug Faires provided invaluable assistance with the inclusion of the Maple examples in this thesis, and the author is extremely grateful for his contribution. Finally, Drs. Rich Goldthwait and Jamal Tartir listened patiently to my ideas on this thesis, helping me to organize my thoughts, and I would like to thank them for this.

Chapter 1

Introduction to Poisson's Equation and Tridiagonal Matrices

One of the most important—and most studied—partial differential equations is Poisson's equation,

$$u_{xx} + u_{yy} = f(x,y). \tag{1.1}$$

It occurs frequently in mathematics, physics, and engineering; common applications include the study of electrostatic and gravitational potential, incompressible fluid flow, vibrating membranes, and steady-state heat distribution problems. Time t is not involved in the Poisson equation. Thus, these applications focus on states of equilibrium, and the problems are 'boundary-value' problems instead of 'initial-condition' problems.

As an example, let's look at the equilibrium distribution of heat in a region R with boundary S. We will assume that the heat flux is given by a constant times the gradient of the distribution. This, combined with the law of conservation of energy, gives $f(x,y) = 0$. Also, we will assume that the temperature distribution within R is governed by the temperatures on S. Thus, we have the Dirichlet boundary conditions, $u(x,y) = g(x,y)$ for all $(x,y) \in S$. Summarizing, our differential equation has become the boundary-value problem

$$\begin{aligned} u_{xx} + u_{yy} &= 0 \quad \text{for all } (x,y) \in R \\ u(x,y) &= g(x,y) \quad \text{for all } (x,y) \in S. \end{aligned} \tag{1.2}$$

Much of the material in the next two sections is borrowed from [2] and [6].

1.1 The Finite-Difference Approximation Method

Because computers are routinely used in the solution of the above boundary-value problem, it is rarely solved explicitly. Instead, approximations to the actual values in R are made [2]. To simplify our development of the approximations, we will assume R is a rectangle with $a < x < b$ and $c < y < d$. (In fact, R may be any shape, and numerous studies have been made on these non-rectangular regions. For examples, see [3], [4], and [13].)

The approximations will be generated by the Finite-Difference method. First, we need to choose integers m and n. We may now define step sizes $h = (b-a)/m$ and $k = (d-c)/n$. Next, divide the interval $[a,b]$ into m parts of width h and the interval $[c,d]$ into n parts of width k, and place a grid on R by drawing horizontal and vertical lines through the points (x_i, y_j), where

$$x_i = a + ih, \text{ and } y_j = c + jk$$

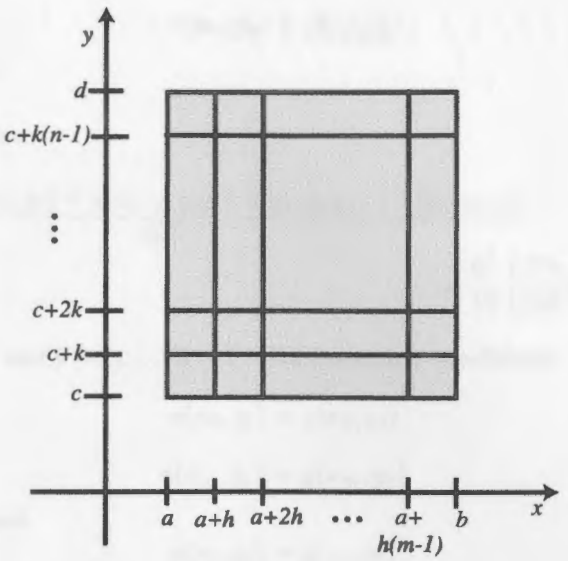

Figure 1.1: Illustration of Grid Lines and Mesh Points

for each $i = 0, 1, \ldots, m$, and $j = 0, 1, \ldots, n$.

The lines $x = x_i$ and $y = y_j$ are the **grid lines** and their intersections are the **mesh points** of the grid. (See Figure 1.1.) For the interior mesh points, we can apply Taylor's theorem in x about x_i. The result is the centered-difference formula

$$\frac{\partial^2 u}{\partial x^2}(x_i, y_j) = \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{h^2} - \frac{h^2}{12}\frac{\partial^4 u}{\partial x^4}(\xi_i, y_j),$$

where $\xi_i \in (x_{i-1}, x_{i+1})$. Similarly, we can apply Taylor's theorem in y about y_j, resulting in the centered-difference formula

$$\frac{\partial^2 u}{\partial y^2}(x_i, y_j) = \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{k^2} - \frac{k^2}{12}\frac{\partial^4 u}{\partial y^4}(x_i, \eta_j),$$

where $\eta_j \in (y_{j-1}, y_{j+1})$.

Substituting these formulae into (1.2), we can state the Poisson boundary-value problem at (x_i, y_j) as

$$\frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{h^2} + \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{k^2} = \frac{h^2}{12}\frac{\partial^4 u}{\partial x^4}(\xi_i, y_j) + \frac{k^2}{12}\frac{\partial^4 u}{\partial y^4}(x_i, \eta_j) \tag{1.3}$$

for each $i = 1, \ldots, m-1$ and $j = 1, \ldots, n-1$, with boundary conditions

$$u(x_0, y_j) = g(x_0, y_j)$$

$$u(x_m, y_j) = g(x_m, y_j)$$

for each $j = 0, 1, \ldots, n$, and

$$u(x_i, y_0) = g(x_i, y_0)$$

$$u(x_i, y_n) = g(x_i, y_n)$$

for each $i = 1, \ldots, m-1$.

When $h = k$, (1.3) becomes

$$\frac{u(x_{i+1}, y_j) - 4u(x_i, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1})}{h^2} = \frac{h^2}{12}\left[\frac{\partial^4 u}{\partial x^4}(\xi_i, y_j) + \frac{\partial^4 u}{\partial y^4}(x_i, \eta_j)\right], \tag{1.4}$$

for each $i = 1, \ldots, m-1$ and $j = 1, \ldots, n-1$, with boundary conditions

$$u(x_0, y_j) = g(x_0, y_j)$$

$$u(x_m, y_j) = g(x_m, y_j)$$

for each $j = 0, 1, \ldots, n$, and

$$u(x_i, y_0) = g(x_i, y_0)$$

$$u(x_i, y_n) = g(x_i, y_n)$$

for each $i = 1, \ldots, m-1$.

Allowing for a truncation error of order $O(h^2)$, our approximation becomes

$$u(x_{i+1}, y_j) - 4u(x_i, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}) = 0, \tag{1.5}$$

for each $i = 1, \ldots, m-1$ and $j = 1, \ldots, n-1$, with boundary conditions

$$u(x_0, y_j) = g(x_0, y_j)$$

$$u(x_m, y_j) = g(x_m, y_j)$$

for each $j = 0, 1, \ldots, n$, and

$$u(x_i, y_0) = g(x_i, y_0)$$

$$u(x_i, y_n) = g(x_i, y_n)$$

for each $i = 1, \ldots, m-1$.

If we relabel each point in the grid by

$$P_l = (x_i, y_j),$$

where $l = i + (n - 1 - j)(m - 1)$, we obtain a system of equations

$$T\vec{u} = \vec{d},$$

where T is a matrix of order $(m-1)(n-1) \times (m-1)(n-1)$. To illustrate this, we will look at the following example.

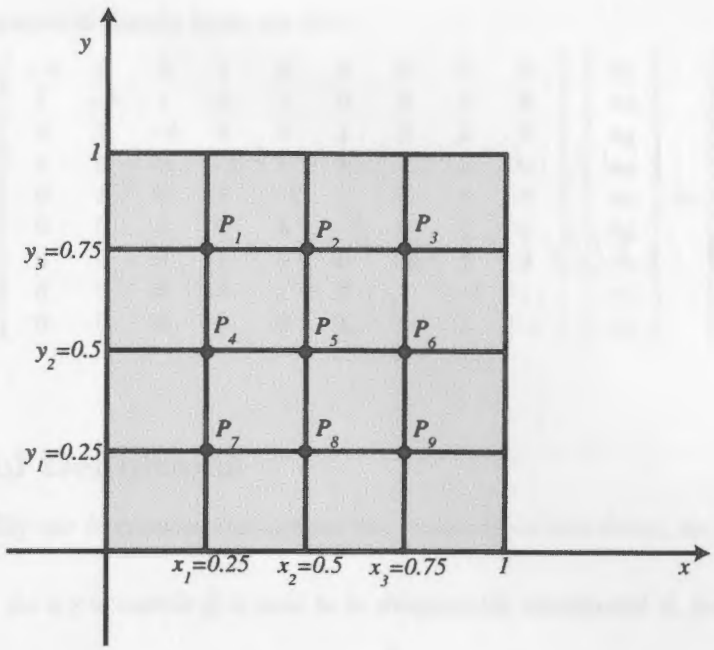

Figure 1.2: Labelling of Interior Mesh Points

1.2 Example

Consider the region R defined by

$$R = \{(x,y) \in \mathbb{R}^2 : 0 < x < 1,\ 0 < y < 1\}.$$

If $h = 0.25$, then $m = n = 4$. Labelling the interior mesh points in the manner proposed above, our region would appear as in Figure 1.2.

Applying (1.5) to each P_l and utilizing the boundary conditions, we develop the following system of equations:

$$\begin{aligned}
-4u_1 + u_2 + u_4 &= d_1 \\
u_1 - 4u_2 + u_3 + u_5 &= d_2 \\
u_2 - 4u_3 + u_6 &= d_3 \\
u_1 - 4u_4 + u_5 + u_7 &= d_4 \\
u_2 + u_4 - 4u_5 + u_6 + u_8 &= 0 \\
u_3 + u_5 - 4u_6 + u_9 &= d_6 \\
u_4 - 4u_7 + u_8 &= d_7 \\
u_5 + u_7 - 4u_8 + u_9 &= d_8 \\
u_6 + u_8 - 4u_9 &= d_9,
\end{aligned}$$

where u_l represents the approximate value of the distribution function at P_l and d_l represents the accumulation of boundary values adjacent to P_l.

Rewriting this system in matrix form, we have

$$(1.6) \qquad T\vec{u} = \begin{bmatrix} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ 0 \\ d_6 \\ d_7 \\ d_8 \\ d_9 \end{bmatrix} = \vec{d}.$$

1.3 Useful Definitions

In order to simplify our discussions throughout the remainder of this thesis, we need to define several terms.

Definition 1.1 *An $n \times n$ matrix A is said to be **diagonally dominant** if, for all $i = 1, \ldots, n$, and $j = 1, \ldots, n$,*

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|.$$

*If the equality is removed, then A is said to be **strictly diagonally dominant.***

Definition 1.2 *An $n \times n$ matrix A is **symmetric** if $A = A^T$.*

Definition 1.3 *An $n \times n$ matrix A is a **diagonal** matrix if $a_{ij} = 0$ for all $i \neq j$ and $a_{ii} \neq 0$ for some $i = 1, \ldots, n$.*

Definition 1.4 *An $n \times n$ matrix A is a **band matrix** if there are integers p and q, with $1 < p, q < n$, such that $a_{ij} = 0$ whenever $i + p \leq j$ or $j + q \leq i$. The **bandwidth** of a band matrix is $w = p + q - 1$.*

Definition 1.5 *A band matrix is **tridiagonal** if $w = 3$.*

Definition 1.6 *A band matrix is **pentadiagonal** if $w = 5$.*

Definition 1.7 *Let $\{m_1, ..., m_p\}$ and $\{n_1, \ldots, n_q\}$ be sets of positive integers. Let $A_{ij} \in \mathbb{R}^{m_i \times n_j}$ for $i = 1, \ldots, p$, and $j = 1, \ldots, q$. Then matrix A is a $p \times q$ **block matrix** if A can written as*

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1q} \\ \vdots & \ddots & \vdots \\ A_{p1} & \cdots & A_{pq}. \end{bmatrix}$$

Definition 1.8 *An $n \times n$ block matrix A with square invertible diagonal blocks is **block diagonally dominant** if for all $i = 1, \ldots, n$,*

$$\|A_{ii}^{-1}\| \sum_{\substack{j=1 \\ j \neq i}}^{n} \|A_{ij}\| < 1$$

for some norm.

We may also talk about *block symmetric*, *block tridiagonal*, and *block pentadiagonal* matrices throughout this thesis. The definitions of these terms are analogous to definitions of the standard terms.

1.4 Some Observations on Example 1.2

There are several interesting things about the matrix T from Example 1.2. First, notice that T is diagonally dominant, symmetric, and pentadiagonal. Also, it has a certain pattern to its entries. If we let

$$A = \begin{bmatrix} -4 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & -4 \end{bmatrix}$$

and B be the 3×3 identity, we are able to rewrite T as the block matrix

$$T = \begin{bmatrix} A & B & 0 \\ B & A & B \\ 0 & B & A \end{bmatrix}.$$

Notice that this substitution makes T block symmetric and block tridiagonal. Also, T is now block diagonally dominant, which may be verified with a few elementary calculations. All of these properties will be useful in the later chapters of this thesis. Finally, notice that A is also symmetric, diagonally dominant, and tridiagonal.

1.5 Summary

In this chapter, we have developed block tridiagonal matrix systems for approximations to the solution of Poisson's equation. The remaining chapters will focus on one method of solving these systems—cyclic odd-even reduction. The next two chapters will discuss the cyclic reduction algorithm as it applies to tridiagonal systems, with the remaining chapters devoted to block tridiagonal systems.

Several numerical examples are used in this thesis to illustrate procedures and algorithms. The values in these examples were computed using Maple 9.5 with ten-digit arithmetic. The Maple routines vary from efficient to labor-intensive, depending on the desired illustration. Those designed to demonstrate the processes in a step-by-step fashion are more labor intensive than those designed to test the algorithms.

Chapter 2

Cyclic Reduction for Tridiagonal Matrices ($n = 1$)

As noted in the first chapter, we would like to solve the system of equations

$$T\vec{x} = \begin{bmatrix} A & B & & 0 \\ B & \ddots & \ddots & \\ & \ddots & \ddots & B \\ 0 & & B & A \end{bmatrix} \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_m \end{bmatrix} = \begin{bmatrix} \vec{d}_1 \\ \vdots \\ \vec{d}_m \end{bmatrix} = \vec{d}, \tag{2.1}$$

where T is a diagonally dominant matrix of $m \times m$ blocks, A and B are $n \times n$ matrices, A is invertible, and $AB = BA$. For simplicity, we will assume that $m = 2^k - 1$. Note that the overall dimension of T is $mn \times mn$. Also, $\vec{x}_j$ and $\vec{d}_j$ are n-vectors.

When $n = 1$, (2.1) takes on the form

$$T\vec{x} = \begin{bmatrix} a & b & & 0 \\ b & \ddots & \ddots & \\ & \ddots & \ddots & b \\ 0 & & b & a \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} d_1 \\ \vdots \\ d_m \end{bmatrix} = \vec{d}. \tag{2.2}$$

The cyclic reduction algorithm is based on two operations—the decoupling of the even-indexed variables from the odd-indexed ones (which is the *reduction* phase) and the eventual recovery of the odd-indexed variables through back-substitution. In this chapter, we will demonstrate the ideas behind both phases of cyclic reduction. Based on observations about these phases, we will provide an algorithm for the cyclic reduction of tridiagonal matrices. Our approach to the reduction phase will be based on the approach of Bondeli and Gander [1] instead of the approach of Hockney [10].

2.1 The Reduction Phase

Consider the three consecutive equations of (2.2), where j is even,

$$\begin{aligned} bx_{j-2} + ax_{j-1} + bx_j &= d_{j-1} \\ bx_{j-1} + ax_j + bx_{j+1} &= d_j \\ bx_j + ax_{j+1} + bx_{j+2} &= d_{j+1} \end{aligned} \tag{2.3}$$

To eliminate the odd-indexed variables, we should multiply the first and third equations in (2.3) by $-b/a$ and add them to the second equation. This gives us

$$(-b^2/a)x_{j-2} + (a - 2b^2/a)x_j + (-b^2/a)x_{j+2} = d_j - (b/a)(d_{j-1} + d_{j+1}). \tag{2.4}$$

By continuing this process throughout T, we can decouple the even-indexed variables from the odd-indexed ones; we then obtain the new system

$$T^{(1)}\vec{x}^{(1)} = \begin{bmatrix} a^{(1)} & b^{(1)} & & \\ b^{(1)} & \ddots & \ddots & \\ & \ddots & \ddots & b^{(1)} \\ & & b^{(1)} & a^{(1)} \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ \vdots \\ x_{m-1} \end{bmatrix} = \begin{bmatrix} d_2^{(1)} \\ d_4^{(1)} \\ \vdots \\ d_{m-1}^{(1)} \end{bmatrix} = \vec{d}^{(1)}, \tag{2.5}$$

where $a^{(1)} = a - 2b^2/a$, $b^{(1)} = -b^2/a$, and $d_j^{(1)} = d_j - (b/a)(d_{j-1} + d_{j+1})$.

Notice that (2.5) is similar to (2.2), except that it has dimension $m^{(1)} = 2^{k-1} - 1$. So, we can multiply the odd equations in (2.5) by $-b^{(1)}/a^{(1)}$. Adding these new odd equations to the adjacent even equations gives us

$$T^{(2)}\vec{x}^{(2)} = \begin{bmatrix} a^{(2)} & b^{(2)} & & \\ b^{(2)} & \ddots & \ddots & \\ & \ddots & \ddots & b^{(2)} \\ & & b^{(2)} & a^{(2)} \end{bmatrix} \begin{bmatrix} x_4 \\ x_8 \\ \vdots \\ x_{m-3} \end{bmatrix} = \begin{bmatrix} d_4^{(2)} \\ d_8^{(2)} \\ \vdots \\ d_{m-3}^{(2)} \end{bmatrix} = \vec{d}^{(2)}, \tag{2.6}$$

where

$$a^{(2)} = a^{(1)} - 2[b^{(1)}]^2/a^{(1)},$$
$$b^{(2)} = -[b^{(1)}]^2/a^{(1)},$$

and

$$d_j^{(2)} = d_j^{(1)} - [b^{(1)}/a^{(1)}][d_{j-2}^{(1)} + d_{j+2}^{(1)}].$$

Repeatedly applying this process to (2.6) eventually gives the single equation

$$a^{(k-1)}x_{2^{k-1}} = d_{2^{k-1}}^{(k-1)}. \tag{2.7}$$

Once we solve (2.7) for $x_{2^{k-1}}$, we can then back-substitute to determine the values of the remaining variables.

2.1.1 Example (reduction phase only)

Let's look at this process in a system where $m = 7$. We will take the initial system

$$T\vec{x} = \begin{bmatrix} -4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} -3 \\ -2 \\ -2 \\ -2 \\ -2 \\ -2 \\ -3 \end{bmatrix} = \vec{d}. \tag{2.8}$$

Using the previous reduction process, the first decoupled system is

$$T^{(1)}\vec{x}^{(1)} = \begin{bmatrix} -3.5 & 0.25 & 0 \\ 0.25 & -3.5 & 0.25 \\ 0 & 0.25 & -3.5 \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \end{bmatrix} = \begin{bmatrix} -3.25 \\ -3 \\ -3.25 \end{bmatrix} = \vec{d}^{(1)}. \tag{2.9}$$

A second application of the reduction routine gives the equation

$$-3.464285714x_4 = -3.464285714. \tag{2.10}$$

2.1.2 Some observations about the reduction phase

Let r be the depth of the reduction of the system. That is, for (2.2), $r = 0$; $r = 1$ for (2.5); and so on. Then, we can make the following observations.

1 At each step of the reduction, the variables indexed by the even multiples of 2^{r-1} are decoupled from those indexed by the odd multiples.

2 After the rth step of the reduction, the dimension of the decoupled system is $m^{(r)} = 2^{k-r} - 1$.

3 The last reduction step occurs when $r = k - 1$.

4 After the rth step of the reduction, the entries in the new system are

$$a^{(r)} = a^{(r-1)} - 2[b^{(r-1)}]^2/a^{(r-1)},$$

$$b^{(r)} = -[b^{(r-1)}]^2/a^{(r-1)},$$

and

$$d_j^{(r)} = d_j^{(r-1)} - [b^{(r-1)}/a^{(r-1)}][d_{j-2^{r-1}}^{(r-1)} + d_{j+2^{r-1}}^{(r-1)}],$$

where j ranges over the multiples of 2^r that are less than m.

Reviewing these observations, it appears possible for the reduction to break down if $a^{(r-1)}$ ever becomes zero. However, that is not the case, as will be shown in Section 2.4.

2.2 The Back-Substitution Phase

To illustrate the back-substitution process, let's begin by looking at the general system for $m = 7$.

$$T\vec{x} = \begin{bmatrix} a & b & 0 & 0 & 0 & 0 & 0 \\ b & a & b & 0 & 0 & 0 & 0 \\ 0 & b & a & b & 0 & 0 & 0 \\ 0 & 0 & b & a & b & 0 & 0 \\ 0 & 0 & 0 & b & a & b & 0 \\ 0 & 0 & 0 & 0 & b & a & b \\ 0 & 0 & 0 & 0 & 0 & b & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \end{bmatrix} = \vec{d}. \tag{2.11}$$

Applying the reduction process to (2.11), we obtain the decoupled system

$$T^{(1)}\vec{x}^{(1)} = \begin{bmatrix} a^{(1)} & b^{(1)} & 0 \\ b^{(1)} & a^{(1)} & b^{(1)} \\ 0 & b^{(1)} & a^{(1)} \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \end{bmatrix} = \begin{bmatrix} d_2^{(1)} \\ d_4^{(1)} \\ d_6^{(1)} \end{bmatrix} = \vec{d}^{(1)}, \tag{2.12}$$

where

$$a^{(1)} = a - 2b^2/a,$$
$$b^{(1)} = -b^2/a,$$

and

$$d_j^{(1)} = d_j - b(d_{j-1} + d_{j+1})/a.$$

Performing the reduction on (2.12), we find that

$$a^{(2)}x_4 = d_4^{(2)}, \tag{2.13}$$

where

$$a^{(2)} = a^{(1)} - 2[b^{(1)}]^2/a^{(1)}$$

and

$$d_4^{(2)} = d_4^{(1)} - b^{(1)}(d_2^{(2)} + d_6^{(1)})/a^{(1)}.$$

Solving (2.13), we determine the value of x_4 to be

$$x_4 = d_4^{(2)}/a^{(2)}. \tag{2.14}$$

Substituting this value of x_4 into (2.12), we can find the values of x_2 and x_6, which are

$$\begin{aligned} x_2 &= (d_2^{(1)} - b^{(1)}x_4)/a^{(1)} \\ x_6 &= (d_6^{(1)} - b^{(1)}x_4)/a^{(1)}. \end{aligned} \tag{2.15}$$

Finally, we can solve for x_1, x_3, x_5, and x_7 through the substitution of the even-indexed variables into (2.11). Doing this, we obtain

$$\begin{aligned} x_1 &= (d_1 - bx_2)/a \\ x_3 &= [d_3 - b(x_2 - x_4)]/a \\ x_5 &= [d_5 - b(x_4 - x_6)]/a \\ x_7 &= (d_7 - bx_6)/a. \end{aligned} \tag{2.16}$$

2.2.1 Example (back-substitution phase only)

Let's return to our previous example. The equation that remained at the end of the reduction was

$$-3.464285714x_4 = -3.464285714. \tag{2.17}$$

Solving this equation for x_4, we determine that

$$x_4 = 1.$$

Using this value of x_4 and using the formulae in (2.15), we see

$$\begin{aligned} x_2 &= (-3.25 - 0.25 \times 1)\,/\,(-3.5) \\ &= (-3.25 - 0.25)\,/\,(-3.5) \\ &= 1, \end{aligned}$$

and

$$x_6 = 1.$$

Finally, using the formulae in (2.16), we obtain

$$\begin{aligned} x_1 &= (-3-1)/(-4) \\ &= 1, \\ \\ x_3 &= (-2-1-1)/(-4) \\ &= 1, \\ \\ x_5 &= 1, \\ \\ \text{and } x_7 &= 1. \end{aligned}$$

2.2.2 Some observations about the back-substitution

Let r be the depth of the *reduction* of the system. That is, (2.11) has a depth of $r = 0$, (2.12) has a depth of $r = 1$, and (2.13) has a depth of $r = 2$. From the above development, we can make the following observations.

1 At each stage of the back-substitution, we are finding the value of the variables x_j, where $j = 2^r, 2^r + 2^{r+1}, \ldots, m + 1 - 2^r$. That is, j ranges over the odd multiples of 2^r that are less than or equal to m.

2 For each x_j, $x_j = [d_j^{(r)} - b^{(r)}(x_{j-2^r} + x_{j+2^r})]/a^{(r)}$.

We should note that the second item requires dummy variables x_0 and x_{m+1}, both of which must be set equal to 0.

2.3 The Cyclic Reduction Algorithm

The observations we made in the two previous sections can be combined into an algorithm for cyclic reduction. Specifically, when $n = 1$, we have the following algorithm. (In an effort to bridge the gap between the above development and the writing of computer programs, the superscripts on a and b have become subscripts. Also, the superscripts and subscripts on d have been combined into a two-element subscript such that $d_{j,r}$ is the jth element of $\vec{d}^{(r)}$. Finally, the back-substitution relies on the fact that $x_0 = x_{m+1} = 0$.)

THE REDUCTION PHASE
$s = 1$
$a_0 = a$
$b_0 = b$
for $r = 1, \ldots, k-1$
  $b_r = -b_{r-1}^2/a_{r-1}$
  $a_r = a_{r-1} + 2b_r$
  $t = s$
  $s = 2s$
  for $j = s, 2s, \ldots, m + 1 - s$
    $d_{j,r} = d_{j,r-1} - b_{r-1}(d_{j-t,r-1} + d_{j+t,r-1})/a_{r-1}$
  end
end

THE BACK-SUBSTITUTION PHASE
$s = (n+1)/2$
$x_s = d_{s,k-1}/a_{k-1}$
for $i = k-2, k-3, \ldots, 0$
  $t = s$
  $s = s/2$
  for $j = s, s+t, \ldots, m+1-s$
    $x_j = [d_{j,i} - b_i(x_{j-s} + x_{j+s})]/a_i$
  end
end

2.3.1 Example

Let's demonstrate the algorithm on a system with $m = 127$. Notice that this makes $k = 7$. For this example, let $a = -4$, $b = 1$, and $d_j = 1$ for $j = 1, \ldots, 127$.

Initially, we set $a_0 = a$, $b_0 = b$, and $s = 1$. So,

$$a_0 = -4,$$

$$b_0 = 1,$$

and

$$s = 1.$$

Continuing with the algorithm, when $r = 1$, we have

$$\begin{aligned} b_1 &= 0.2500000000, \\ \text{and } a_1 &= -3.500000000. \end{aligned}$$

Because all $d_j = 1$ to begin this example, all d_j will be equal throughout the example. Finishing this step of the algorithm, we find

$$d_{j,1} = 1.5000000000$$

for $j = 2, 4, \ldots, 126$.

For $r = 2$, we find

$$\begin{aligned} b_2 &= 0.01785714286, \\ a_2 &= -3.464285714, \\ \text{and } d_{j,2} &= 1.714285714 \end{aligned}$$

for $j = 4, 8, \ldots, 124$.

When $r = 3$, our values become

$$\begin{aligned} b_3 &= 0.00009204712816, \\ a_3 &= -3.464101620, \\ \text{and } d_{j,3} &= 1.731958763 \end{aligned}$$

for $j = 8, 16, \ldots, 120$.

Next, when $r = 4$,

$$\begin{aligned} b_4 &= 2.445850247 \times 10^{-9}, \\ a_4 &= -3.464101615, \\ \text{and } d_{j,4} &= 1.732050805 \end{aligned}$$

for $j = 16, 32, \ldots, 112$.

When $r = 5$, we have

$$\begin{aligned} b_5 &= 1.726907607 \times 10^{-18}, \\ a_5 &= -3.464101615, \\ \text{and } d_{j,5} &= 1.732050807 \end{aligned}$$

for $j = 32, 64, 96$.

Our final reduction occurs when $r = 6$. At this point,

$$\begin{aligned} a_6 &= -3.464101615, \\ \text{and } d_{64,6} &= 1.732050807. \end{aligned}$$

So, at the end of the reduction phase, we have the equation

$$-3.464101615\, x_{64} = 1.732050807. \tag{2.18}$$

To begin the back-substitution phase, we solve (2.18) for x_{64}. We then begin the back-substitutions.

In the first back-substitution, we find values for x_{32} and x_{96}. The second round of back-substitution determines the values of x_{16}, x_{48}, x_{80}, and x_{112}. The third step of the back-substitution gives the values of $x_8, x_{24}, \ldots, x_{120}$. The values of $x_4, x_{12}, \ldots, x_{124}$ are found in the fourth recovery of variables. Back-substitution for a fifth time determines the values of $x_2, x_6, \ldots, x_{126}$. The final step in the back-substitution sequence will give the values of the odd-indexed variables. These values are shown in the Maple 9.5 output in Appendix C.

Once the back-substitution was done, the routine checked the solution. The error was measured by taking the infinity norm of $T\vec{x} - \vec{d}$. This error was found to be 1×10^{-9}. It took Maple 9.5 a total of 2.75 seconds to run the routine, including the check.

2.4 The Stability of the Algorithm

In studying the above algorithm, we notice that every step in the reduction involves division by a_{r-1}. Thus, the algorithm will fail should $a_{r-1} = 0$ at any time in the routine. However, this is not possible, as proven below.

Theorem 2.1 *Let A be a diagonally dominant, tridiagonal, symmetric, Toeplitz matrix, and let $|a| \geq 2|b|$. The sequence $\{a_i\}$ produced by the above algorithm converges quadratically to*

$$\text{sign}(a)\sqrt{a^2 - 4b^2}.$$

Also, $a_i \neq 0$ for all $i \in \mathbb{N}$.

Proof: By Algorithm 2.3, we know

$$a_i = a_{i-1} + 2b_i. \tag{2.19}$$

This implies

$$\frac{a_i - a_{i-1}}{2} = b_i. \tag{2.20}$$

Substituting (2.20) into the computation of the off-diagonal elements, we obtain

$$\begin{aligned} b_i &= -\frac{b_{i-1}^2}{a_{i-1}} \qquad (2.21) \\ \frac{a_i - a_{i-1}}{2} &= -\left(\frac{a_{i-1} - a_{i-2}}{2}\right)^2 \Big/ a_{i-1} \\ &= -\frac{(a_{i-1} - a_{i-2})^2}{4a_{i-1}} \\ a_i - a_{i-1} &= -\frac{(a_{i-1} - a_{i-2})^2}{2a_{i-1}} \\ 2a_i a_{i-1} - 2a_{i-1}^2 &= -a_{i-1}^2 + 2a_{i-1}a_{i-2} - a_{i-2}^2 \\ 2a_i a_{i-1} - a_{i-1}^2 - 2a_{i-1}a_{i-2} + a_{i-2}^2 &= 0. \qquad (2.22) \end{aligned}$$

We recognize that (2.22) is a nonlinear homogeneous difference equation of order two with initial conditions

$$a_0 = a$$

and

$$a_1 = \frac{a^2 - 2b^2}{a}.$$

For $i = 1, 2, \ldots$, define

$$r_i = 2a_i a_{i-1} - a_{i-1}^2. \tag{2.23}$$

Substituting this into (2.22) gives

$$r_i - r_{i-1} = 0.$$

This linear homogeneous difference equation has constant solutions

$$r_i = r \tag{2.24}$$

for $i = 1, 2, \ldots$.

Using the initial conditions, we find that

$$\begin{aligned} r &= r_1 \\ &= 2a_0 a_1 - a_0^2 \\ &= 2a\frac{a^2 - 2b^2}{a} - a^2 \\ &= 2a^2 - 4b^2 - a^2 \\ &= a^2 - 4b^2. \end{aligned}$$

Substituting (2.24) into (2.23) yields

$$\begin{aligned} r &= 2a_i a_{i-1} - a_{i-1}^2 \\ \Rightarrow \quad 2a_i a_{i-1} &= a_{i-1}^2 + r \\ \Rightarrow \quad a_i &= \frac{a_{i-1}}{2} + \frac{r}{2a_{i-1}}. \end{aligned} \tag{2.25}$$

The recursion defined by (2.25) is nothing more than Newton's method for the solution of the equation $f(x) = x^2 - r = 0$. Because $f'(\sqrt{r}) = 2\sqrt{r} = 2\sqrt{a^2 - 4b^2} \neq 0$, the sequence $\{a_i\}$ converges quadratically to $\sqrt{r} = \sqrt{a^2 - 4b^2}$.

$$\begin{aligned} |a| &\geq 2|b| \\ f'(\sqrt{r}) &= 2\sqrt{a^2 - 4b^2} \\ &\geq 0. \end{aligned}$$

Thus, the sequence $\{a_i\}$ is monotone, which implies

$$\lim_{i \to \infty} a_i = \text{sign}(a)\sqrt{a^2 - 4b^2}. \tag{2.26}$$

Because $|a| \geq 2|b|$, the sequence $\{a_r\}$ is monotone, and $\lim_{i \to \infty} a_i = \text{sign}(a)\sqrt{a^2 - 4b^2}$, $a_r \neq 0$ for all $r \in \mathbb{N}$. Thus, the algorithm is stable. **QED.**

2.5 Summary

In Example 2.3.1, notice that the sequence $\{b_i\}$ converged to 0 quickly. This means that the changes in a_i and $\vec{d}_i$ become much smaller as the depth of the reduction increases. In fact, the value of a_r became constant in ten-digit arithmetic after the fourth reduction step, and the value of $d_{j,r}$ became constant after the fifth reduction step. Thus, we should be able terminate the reduction phase earlier than step $k - 1$, saving costs (as measured in floating-point operations) when solving the system numerically.

In this chapter, we have developed a cyclic reduction algorithm for those systems where the dimension of the block A is $n = 1$. As noted by Bondeli and Gander, the algorithm requires $O(8m)$ floating-point operations [1]. However, we can save overhead by using a truncated algorithm based on the rapid convergence (noted above) of the sequence $\{a_r\}$. This truncated cyclic reduction algorithm will be developed in the next chapter.

Chapter 3

A Truncated Cyclic Reduction for Tridiagonal Matrices ($n = 1$)

As proven at the end of the previous chapter, the sequence $\{a_r\}$ converges rapidly. This rapid convergence of the sequence allows us to save costs (as measured by floating-point operations) by ending the algorithm early. In this chapter, we will use this convergence as a basis for the development of the Bondeli-Gander truncated algorithm for cyclic reduction [1].

3.1 Revisiting the Reduction Phase

As in Chapter 2, we are interested in solving the system of equations

$$T\vec{x} = \begin{bmatrix} a & b & & 0 \\ b & \ddots & \ddots & \\ & \ddots & \ddots & b \\ 0 & & b & a \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} d_1 \\ \vdots \\ d_m \end{bmatrix} = \vec{d}. \tag{3.1}$$

The next two lemmas will use the quadratic convergence of the sequence $\{a_i\}$ to give us values for that sequence.

Lemma 3.1 *Suppose $|a| = 2|b|$. Then the off-diagonal and diagonal elements are given by*

$$b_k = -\text{sign}(a)(b/2^k), \tag{3.2}$$

$$a_k = \text{sign}(a)(b/2^{k-1}), \tag{3.3}$$

for $k = 1, 2, \ldots$.

Proof by induction: Let $n = 1$. From Algorithm 2.3,

$$\begin{aligned} b_1 &= -b_0^2/a_0 \\ &= -b^2/a \\ &= -b^2/(\pm 2b) \\ &= \mp b/2 \\ &= -\text{sign}(a)(b/2^n). \end{aligned}$$

Also,

$$\begin{aligned} a_1 &= a_0 - 2b_0^2/a_0 \\ &= \pm 2b - 2b^2/(\pm 2b) \\ &= \pm 2b \mp b \\ &= \pm b \\ &= \text{sign}(a)(b/2^{n-1}). \end{aligned}$$

Now, assume (3.2) and (3.3) hold for $n = k$. Then, by Algorithm 2.3,

$$b_{k+1} = -b_k^2/a_k.$$

Using (3.2) and (3.3), we obtain

$$\begin{aligned} b_{k+1} &= \frac{-[\mp b/2^k]^2}{\pm b/2^{k-1}} \\ &= \frac{-b^2/2^{2k}}{\pm b/2^{k-1}} \\ &= \mp b/2^{k+1} \\ &= -\text{sign}(a)(b/2^{k+1}). \end{aligned}$$

Also,

$$\begin{aligned} a_{k+1} &= a_k - 2b_k^2/a_k \\ &= \frac{\pm b}{2^{k-1}} - \frac{2[\mp b^2/2^k]^2}{\pm b/2^{k-1}} \\ &= \frac{\pm b}{2^{k-1}} - \frac{2b^2/2^{2k}}{\pm b/2^{k-1}} \\ &= \frac{\pm 2b}{2^k} - \frac{\pm b}{2^k} \\ &= \pm b/2^k \\ &= \text{sign}(a)(b/2^{(k+1)-1}). \end{aligned}$$

Thus, (3.2) and (3.3) hold for $n = k + 1$. **QED**.

Lemma 3.2 *Suppose $|a| > 2|b|$. Then the diagonal elements produced by the reduction phase of Algorithm 2.3 are given by*

$$a_i = \sqrt{a^2 - 4b^2}\coth(y_0 2^i), \tag{3.4}$$

where

$$y_0 = \text{sign}(a)[\ln(\sqrt{a^2 - 4b^2} + |a|) - \ln 2|b|].$$

Proof: Let $a_i = \sqrt{r}\coth(y_i)$, and substitute into (2.25). Then,

$$\begin{aligned} \sqrt{r}\coth(y_i) &= \frac{\sqrt{r}\coth(y_{i-1})}{2} + \frac{\sqrt{r}}{2\coth(y_{i-1})} \\ \coth(y_i) &= \frac{\coth(y_{i-1})}{2} + \frac{1}{2\coth(y_{i-1})} \\ &= \frac{\coth^2(y_{i-1}) + 1}{2\coth(y_{i-1})}. \end{aligned} \tag{3.5}$$

Using the identity

$$\coth(2\alpha) = \frac{\coth^2(\alpha)+1}{2\coth(\alpha)},$$

(3.5) becomes

$$\coth(y_i) = \coth(2y_{i-1}). \tag{3.6}$$

Because $\coth(x)$ is injective for all $x \in \mathbb{R} - \{0\}$, (3.6) implies

$$y_i = 2y_{i-1},$$

a first-order linear homogeneous difference equation with solution

$$y_i = 2^i y_0. \tag{3.7}$$

The initial condition for (3.7) is

$$a_0 = \sqrt{r}\coth(y_0).$$

This implies

$$\begin{aligned} \frac{a_0}{\sqrt{r}} &= \coth(y_0) \\ \tanh(y_0) &= \frac{\sqrt{r}}{a_0} \\ y_0 &= \operatorname{arctanh}\left(\frac{\sqrt{a^2-4b^2}}{a}\right). \end{aligned} \tag{3.8}$$

We should note that

$$\left|\frac{\sqrt{a^2-4b^2}}{a}\right| < 1.$$

Using the identity

$$\operatorname{arctanh}(\alpha) = \frac{1}{2}\ln\left(\frac{1+\alpha}{1-\alpha}\right)$$

when $|\alpha| < 1$, (3.8) becomes

$$\begin{aligned} y_0 &= \frac{1}{2}\ln\left(\frac{1+\frac{\sqrt{a^2-4b^2}}{a}}{1-\frac{\sqrt{a^2-4b^2}}{a}}\right) \\ &= \frac{1}{2}\ln\left(\frac{a+\sqrt{a^2-4b^2}}{a-\sqrt{a^2-4b^2}}\right) \\ &= \frac{1}{2}\ln\left[\frac{(a+\sqrt{a^2-4b^2})^2}{a^2-(a^2-4b^2)}\right] \\ &= \frac{1}{2}\ln\left[\frac{(a+\sqrt{a^2-4b^2})^2}{4b^2}\right] \\ &= \ln\left|\frac{a+\sqrt{a^2-4b^2}}{2|b|}\right|. \end{aligned} \tag{3.9}$$

When $a > 2|b|$, (3.9) becomes

$$y_0 = \ln(a + \sqrt{a^2 - 4b^2}) - \ln 2|b|. \tag{3.10}$$

If $a < -2|b|$, then (3.9) becomes

$$\begin{aligned} y_0 &= \ln\left|\frac{a + \sqrt{a^2 - 4b^2}}{2|b|}\right| \\ &= \ln\left(\frac{-a - \sqrt{a^2 - 4b^2}}{2|b|}\right) \\ &= \ln\left[\frac{-a - \sqrt{a^2 - 4b^2}}{2|b|} \cdot \frac{-a + \sqrt{a^2 - 4b^2}}{-a + \sqrt{a^2 - 4b^2}}\right] \\ &= \ln\left[\frac{a^2 - (a^2 - 4b^2)}{2|b|(-a + \sqrt{a^2 - 4b^2})}\right] \\ &= \ln\left(\frac{2|b|}{-a + \sqrt{a^2 - 4b^2}}\right) \end{aligned}$$

$$y_0 = -\ln(-a + \sqrt{a^2 - 4b^2}) + \ln 2|b|. \tag{3.11}$$

Combining (3.10) and (3.11) into a single statement, we find

$$y_0 = \text{sign}(a)[\ln(|a| + \sqrt{a^2 - 4b^2}) - \ln 2|b|]. \tag{3.12}$$

QED.

As we saw with Example 2.3.1, there is a point in the sequence $\{a_i\}$ where the values become constant. This occurs when the values for the elements of the sequence $\{a_i\}$ are within the machine precision of the limit of the sequence. It is at this point that we can terminate the reduction phase of the algorithm and begin the back-substitution.

Lemma 3.3 *Suppose $|a| > 2|b|$. On a computer with precision ϵ, the sequence $\{a_i\}$ becomes constant for $i \geq k_a$, where*

$$k_a = \left\lceil \frac{\log\left[\log\left(\frac{2}{\epsilon} + 1\right)\right] - \log[\log(|a| + \sqrt{a^2 - 4b^2}) - \log 2|b|]}{\log 2} \right\rceil.$$

Proof: Assume $a > 2|b|$. (If $\{a_i\}$ is the sequence corresponding to the initial value a, then $\{-a_i\}$ is the sequence corresponding to the initial value $-a$.) We know $\{a_i\}$ is monotone and converges quadratically to

$$\lim_{i \to \infty} a_i = \sqrt{a^2 - 4b^2}.$$

Therefore, the sequence converges numerically if

$$\frac{a_i - \lim_{i \to \infty} a_i}{\lim_{i \to \infty} a_i} \leq \epsilon. \tag{3.13}$$

Using the identity

$$\coth(\alpha) = \frac{e^{2\alpha}+1}{e^{2\alpha}-1},$$

and (3.4), we find

$$\begin{aligned} a_i &= \sqrt{a^2-4b^2}\,\coth(y_0 2^i) \\ &= \sqrt{a^2-4b^2}\left(\frac{e^{2^i y_0}+1}{e^{2^i y_0}-1}\right). \end{aligned} \tag{3.14}$$

Notice that

$$\begin{aligned} e^{2^i y_0} &= \exp\left[2^i \ln\left(\frac{a+\sqrt{a^2-4b^2}}{2|b|}\right)\right] \\ &= \exp\left[\ln\left(\frac{a+\sqrt{a^2-4b^2}}{2|b|}\right)^{2^i}\right] \\ &= \left(\frac{a+\sqrt{a^2-4b^2}}{2|b|}\right)^{2^i}. \end{aligned} \tag{3.15}$$

Let $z_i = e^{2^i y_0}$. Then (3.14) becomes

$$a_i = \sqrt{a^2-4b^2}\left(\frac{z_i+1}{z_i-1}\right). \tag{3.16}$$

Using (2.26) and (3.16), we obtain

$$\begin{aligned} \frac{a_i - \lim_{i\to\infty} a_i}{\lim_{i\to\infty} a_i} &= \frac{\sqrt{a^2-4b^2}\left(\frac{z_i+1}{z_i-1}\right) - \sqrt{a^2-4b^2}}{\sqrt{a^2-4b^2}} \\ &= \frac{\sqrt{a^2-4b^2}\left(\frac{z_i+1}{z_i-1}-1\right)}{\sqrt{a^2-4b^2}} \\ &= \frac{z_i+1}{z_i-1}-1 \\ \frac{a_i - \lim_{i\to\infty} a_i}{\lim_{i\to\infty} a_i} &= \frac{2}{z_i-1}. \end{aligned} \tag{3.17}$$

Substituting this into (3.13), we find

$$\begin{aligned} \frac{2}{z_i-1} &\le \varepsilon \\ 2 &\le \varepsilon z_i - \varepsilon \\ 2+\varepsilon &\le \varepsilon z_i \\ z_i &\ge \frac{2+\varepsilon}{\varepsilon} \\ z_i &\ge \frac{2}{\varepsilon}+1. \end{aligned} \tag{3.18}$$

(3.19)

Substituting the definition of z_i into (3.19), we see

$$\begin{aligned}
\left(\frac{a+\sqrt{a^2-4b^2}}{2|b|}\right)^{2^i} &\geq \frac{2}{\varepsilon}+1 \\
2^i \log\left(\frac{a+\sqrt{a^2-4b^2}}{2|b|}\right) &\geq \log\left(\frac{2}{\varepsilon}+1\right) \\
2^i &\geq \frac{\log\left(\frac{2}{\varepsilon}+1\right)}{\log\left(\frac{a+\sqrt{a^2-4b^2}}{2|b|}\right)} \\
&= \frac{\log\left(\frac{2}{\varepsilon}+1\right)}{\log(a+\sqrt{a^2-4b^2})-\log 2|b|}
\end{aligned}$$

This implies

$$\begin{aligned}
i\log 2 &\geq \log\left(\frac{\log\left(\frac{2}{\varepsilon}+1\right)}{\log(a+\sqrt{a^2-4b^2})-\log 2|b|}\right) \\
&\geq \log\left[\log\left(\frac{2}{\varepsilon}+1\right)\right]-\log[\log(a+\sqrt{a^2-4b^2})-\log 2|b|] \\
i &\geq \frac{\log\left[\log\left(\frac{2}{\varepsilon}+1\right)\right]-\log[\log(a+\sqrt{a^2-4b^2})-\log 2|b|]}{\log 2}.
\end{aligned}$$

QED.

3.2 Revisiting the Back-Substitution Phase

From the three lemmas in Section 3.1, we can modify the cyclic reduction algorithm to terminate the reduction phase early. However, terminating the reduction phase after some intermediate step $r < k-1$ gives the equation

$$T^{(r)}\vec{x}^{\,(r)} = \begin{bmatrix} a^{(r)} & b^{(r)} & & \\ b^{(r)} & \ddots & \ddots & \\ & \ddots & \ddots & b^{(r)} \\ & & b^{(r)} & a^{(r)} \end{bmatrix} \begin{bmatrix} x_{2^r} \\ \vdots \\ x_{m+1-2^r} \end{bmatrix} = \begin{bmatrix} d_{2^r}^{(r)} \\ \vdots \\ d_{m+1-2^r}^{(r)} \end{bmatrix} = \vec{d}^{\,(r)}, \tag{3.20}$$

where the indices of $\vec{x}^{\,(r)}$ and $\vec{d}^{\,(r)}$ increase by 2^r throughout the vectors.

Notice in (3.20) that we have not decoupled all of the other variables from $x_{2^{k-1}}$. This means we will need to solve the system for all remaining variables. However, as we noted in Example 2.3.1, the sequence $\{b_i\}$ converged quickly to 0. Also, if we have continued the reduction through step k_a, the sequence $\{a_i\}$ is constant; it is nearly constant if r is close to k_a. So, we can approximate the solution to this remaining system by solving the equation

$$a^{(r)}x_i = d_i^{(r)}$$

for $i = 1(2^r), 2(2^r), 3(2^r), \ldots, (m+1-2^r)$. That is, $x_i = d_i^{(r)}/a^{(r)}$.

Once we have found these variables, we can continue with the back-substitution as before.

3.3 The Bondeli-Gander Algorithm

Using our development and comments from the previous two sections, we modify the cyclic reduction algorithm from Section 2.3 slightly. Changes include calculating the earliest point at which we may terminate the reduction, terminating the reduction at this earliest point, and beginning the back-substitution routine by calculating approximations for the variables remaining from the truncated reduction. All other parts of the algorithm remain the same.

THE REDUCTION PHASE

$z = \log[\log(2/\varepsilon)] - \log[\log(|a| + \sqrt{a^2 - 4b^2}) - \log 2|b|]$
$k_a = \min(k-1, \lceil z/\log 2 \rceil)$
$s = 1$
$a_0 = a$
$b_0 = b$
for $r = 1, 2, \ldots, k_a$
    $b_r = -b_{r-1}^2 / a_{r-1}$
    $a_r = a_{r-1} + 2b_r$
    $t = s$
    $s = 2s$
    for $j = s, 2s, \ldots, m+1-s$
        $d_{j,r} = d_{j,r-1} - b_{r-1}(d_{j-t,r-1} + d_{j+t,r-1})/a_{r-1}$
    end
end

THE BACK-SUBSTITUTION PHASE

for $k = s, 2s, \ldots, m+1-s$
    $x_k = d_{s,k_a}/a_{k_a}$
end
for $i = k_a - 1, k_a - 2, \ldots, 0$
    $t = s$
    $s = s/2$
    for $j = s, s+t, \ldots, m+1-s$
        $x_j = [d_{j,i} - b_i(x_{j-s} + x_{j+s})]/a_i$
    end
end

3.3.1 Example

As in the previous chapter, let's demonstrate the algorithm on a system with $m = 127$. Notice that this makes $k = 7$. For this example, let $a = -4$, $b = 1$, and $d_j = 1$ for $j = 1, \ldots, 127$. Also, we set $\varepsilon = 1 \times 10^{-10}$ because we are operating in ten-digit arithmetic with Maple 9.5.

Initially, we set $a_0 = a$, $b_0 = b$, and $s = 1$. So, $a_0 = -4$, $b_0 = 1$, and $s = 1$. Calculating, we find

$$z = 1.255524450,$$

which makes

$$k_a = 5.$$

Continuing with the algorithm, when $r = 1$, we have

$$\begin{aligned} b_1 &= 0.2500000000, \\ \text{and } a_1 &= -3.500000000. \end{aligned}$$

Because all $d_j = 1$ to begin this example, all d_j will be equal throughout the example. Finishing this step of the algorithm, we find

$$d_{j,1} = 1.5000000000$$

for $j = 2, 4, \ldots, 126$.

For $r = 2$, we find

$$\begin{aligned} b_2 &= 0.01785714286, \\ a_2 &= -3.464285714, \\ \text{and } d_{j,2} &= 1.714285714 \end{aligned}$$

for $j = 4, 8, \ldots, 124$.

When $r = 3$, our values become

$$\begin{aligned} b_3 &= 0.00009204712815, \\ a_3 &= -3.464101620, \\ \text{and } d_{j,3} &= 1.731958763 \end{aligned}$$

for $j = 8, 16, \ldots, 120$.

Our final reduction occurs when $r = 4$. At this point,

$$\begin{aligned} b_4 &= 2.445850247 \times 10^{-9}, \\ a_4 &= -3.464101615, \\ \text{and } d_{j,4} &= 1.732050805 \end{aligned}$$

for $j = 16, 32, \ldots, 112$.

To begin the back-substitution phase, we solve the approximation

$$x_j = d_{j,5}/a_5,$$

where $j = 16, 32, \ldots, 112$. We then back-substitute these values to find the remaining variables.

In the first back-substitution, we find values for $x_8, x_{24}, \ldots, x_{120}$. The second round of back-substitution determines the values of $x_4, x_{12}, \ldots, x_{124}$. The third step of the back-substitution gives the values of $x_2, x_6, \ldots, x_{126}$ The values of the odd-indexed variables are found in the fourth recovery of variables. These values are shown in the Maple 9.5 output in Appendix D.

Once the back-substitution was done, the routine checked the solution. The error was measured by taking the infinity norm of $T\vec{x} - \vec{d}$. This error was found to be 2.8×10^{-9}. It took Maple 9.5 a total of 2.66 seconds to run the routine, including the check.

Considering that we only eliminated the calculations for the final two reduction steps of Example 2.3.1, a time savings of 0.09 seconds, or about 3.4 percent, is not surprising. This time savings would be larger if we were dealing with a larger system of equations, which would require more reduction steps.

3.4 Summary

In this chapter, we developed the Bondeli-Gander truncated algorithm for the tridiagonal case. The floating-point operations count for this process is $O((8-4/2^{k_a})m)$ [1]. The savings that are obtained by the use of this algorithm are significant only when $|a|$ is large enough that k_a is small. This would allow the reduction to terminate after a few steps.

We have developed the complete cyclic reduction algorithm, as well as a truncated algorithm, for the tridiagonal case. In the next two chapters, we will develop analogous algorithms for the block tridiagonal case.

Chapter 4

Cyclic Reduction for Block Tridiagonal Matrices ($n > 1$)

In the two previous chapters, we developed the complete cyclic reduction algorithm and a truncated cyclic reduction algorithm for tridiagonal matrices. In this chapter, we will develop the complete cyclic reduction algorithm for block tridiagonal matrices.

Let's look again at (2.1), which was

$$T\vec{x} = \begin{bmatrix} A & B & & 0 \\ B & \ddots & \ddots & \\ & \ddots & \ddots & B \\ 0 & & B & A \end{bmatrix} \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_m \end{bmatrix} = \begin{bmatrix} \vec{d}_1 \\ \vdots \\ \vec{d}_m \end{bmatrix} = \vec{d}, \tag{4.1}$$

where T is a diagonally dominant matrix of $m \times m$ blocks, A and B are $n \times n$ matrices, A is invertible, and $\vec{x}_j$ and $\vec{d}_j$ are n-vectors. Again, we are assuming that $m = 2^k - 1$ for simplicity. Also, note that the overall dimension of T is $mn \times mn$, and the overall lengths of $\vec{x}$ and $\vec{d}$ are mn.

The development of the complete cyclic reduction algorithm for block tridiagonal systems is very similar to the development in Chapter 2 for tridiagonal systems. As such, we will abbreviate parts of the development. Also, we will continue using the approach of Heller [9] for the reduction phase of the algorithm.

4.1 The Reduction Phase

Again, we are interested in the elimination of the odd-indexed variables and their eventual recovery through back-substitution. So, let's look at three consecutive equations in the system, centering on Equation j, where j is even. These equations are

$$\begin{aligned} B\vec{x}_{j-2} + A\vec{x}_{j-1} + B\vec{x}_j &= \vec{d}_{j-1} \\ B\vec{x}_{j-1} + A\vec{x}_j + B\vec{x}_{j+1} &= \vec{d}_j \\ B\vec{x}_j + A\vec{x}_{j+1} + B\vec{x}_{j+2} &= \vec{d}_{j+1}. \end{aligned} \tag{4.2}$$

Multiplying the first and third equations of (4.2) on the left by $-BA^{-1}$ and then adding the three equations, we obtain

$$-BA^{-1}B\vec{x}_{j-2} + (A - 2BA^{-1}B)\vec{x}_j - BA^{-1}B\vec{x}_{j+2} = \vec{d}_j - BA^{-1}(\vec{d}_{j-1} + \vec{d}_{j+1}). \tag{4.3}$$

Continuing this process throughout T and rewriting the decoupled equations as a new system, we have

$$T^{(1)}\vec{x}^{(1)} = \begin{bmatrix} A^{(1)} & B^{(1)} & & \\ B^{(1)} & \ddots & \ddots & \\ & \ddots & \ddots & B^{(1)} \\ & & B^{(1)} & A^{(1)} \end{bmatrix} \begin{bmatrix} \vec{x}_2 \\ \vec{x}_4 \\ \vdots \\ \vec{x}_{m-1} \end{bmatrix} = \begin{bmatrix} \vec{d}_2^{(1)} \\ \vec{d}_4^{(1)} \\ \vdots \\ \vec{d}_{m-1}^{(1)} \end{bmatrix} = \vec{d}^{(1)}, \tag{4.4}$$

where $A^{(1)} = A - 2BA^{-1}B$, $B^{(1)} = -BA^{-1}B$, and $\vec{d}_j^{(1)} = \vec{d}_j - BA^{-1}(\vec{d}_{j-1} + \vec{d}_{j+1})$.

This process may be applied repeatedly, and before the rth reduction step, the system analogous to (4.2) is

$$\begin{aligned} B^{(r-1)}\vec{x}_{(j-2)2^{r-1}} + A^{(r-1)}\vec{x}_{(j-1)2^{r-1}} + B^{(r-1)}\vec{x}_{j2^{r-1}} &= \vec{d}^{(r-1)}_{(j-1)2^{r-1}} \\ B^{(r-1)}\vec{x}_{(j-1)2^{r-1}} + A^{(r-1)}\vec{x}_{j2^{r-1}} + B^{(r-1)}\vec{x}_{(j+1)2^{r-1}} &= \vec{d}^{(r-1)}_{j2^{r-1}} \\ B^{(r-1)}\vec{x}_{j2^{r-1}} + A^{(r-1)}\vec{x}_{(j+1)2^{r-1}} + B^{(r-1)}\vec{x}_{(j+2)2^{r-1}} &= \vec{d}^{(r-1)}_{(j+1)2^{r-1}}. \end{aligned} \tag{4.5}$$

Applying the same process as above, we obtain

$$B^{(r)}\vec{x}_{(j-2)2^{r-1}} + A^{(r)}\vec{x}_{j2^{r-1}} + B^{(r)}\vec{x}_{(j+2)2^{r-1}} = \vec{d}^{(r)}_{j2^{r-1}}, \tag{4.6}$$

where

$$\begin{aligned} B^{(r)} &= -B^{(r-1)}[A^{(r-1)}]^{-1}B^{(r-1)}, \\ A^{(r)} &= A^{(r-1)} - 2B^{(r-1)}[A^{(r-1)}]^{-1}B^{(r-1)}, \\ \text{and } \vec{d}^{(r)}_{j2^{r-1}} &= \vec{d}^{(r-1)}_{j2^{r-1}} - B^{(r-1)}[A^{(r-1)}]^{-1}[\vec{d}^{(r-1)}_{(j-1)2^{r-1}} + \vec{d}^{(r-1)}_{(j+1)2^{r-1}}]. \end{aligned}$$

Continuing to apply this process $k-1$ times, we obtain the single equation

$$A^{(k-1)}\vec{x}_{2^{k-1}} = \vec{d}^{(k-1)}_{2^{k-1}}. \tag{4.7}$$

Once we obtain this equation, we would solve for $\vec{x}_{2^{k-1}}$ by some method; back-substitution would then allow us to recover the eliminated variables.

4.1.1 Example (reduction phase only)

Let's look at this process through an example. In this example, let T be a 7×7 block matrix, A be the 3×3 matrix

$$\begin{bmatrix} -4 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & -4 \end{bmatrix},$$

B be the 3×3 identity, and $\vec{d}_i$ be the vector

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

for $i = 1, \ldots, 7$. Notice that this makes T a 21×21 matrix and that $k = 3$.

Going through the first reduction step, we find that

$$B^{(1)} = \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix},$$

$$A^{(1)} = \begin{bmatrix} -3.464285714 & 1.142857143 & 0.03571428572 \\ 1.142857143 & -3.428571429 & 1.142857143 \\ 0.03571428572 & 1.142857143 & -3.464285714 \end{bmatrix},$$

and

$$\vec{d}_i^{(1)} = \begin{bmatrix} 1.535714286 \\ 0.1428571429 \\ 0.03571428571 \end{bmatrix},$$

for $i = 2, 4, 6$.

The second reduction sequence yields

$$B^{(2)} = \begin{bmatrix} 0.03125000002 & 0.02678571430 & 0.01339285715 \\ 0.02678571430 & 0.04464285716 & 0.02678571429 \\ 0.01339285715 & 0.02678571430 & 0.03125000002 \end{bmatrix},$$

$$A^{(2)} = \begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000002 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix},$$

and

$$\vec{d}_4^{(2)} = \begin{bmatrix} 1.857142858 \\ 0.3750000001 \\ 0.1428571429 \end{bmatrix}.$$

Now, we may solve the equation

$$\begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000002 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix} \vec{x}_4 = \begin{bmatrix} 1.857142858 \\ 0.3750000001 \\ 0.1428571429 \end{bmatrix} \tag{4.8}$$

by some other method. If the block size is small (as in this case), we could use some form of Gaussian elimination.

4.1.2 Some observations about the reduction phase

Let r be the depth of the reduction of the system. That is, for (4.1), $r = 0$; $r = 1$ for (4.4); and so on. Then, we can make the following observations.

1 At the rth step of the reduction, the variables indexed by the even multiples of 2^{r-1} are decoupled from those indexed by the odd multiples.

2 After the rth step of the reduction, the dimension of the decoupled system is $m^{(r)} = 2^{k-r} - 1$.

3 The last reduction step occurs when $r = k - 1$.

4 After the rth step of the reduction, the entries in the new system are

$$\begin{aligned} A^{(r)} &= A^{(r-1)} - 2B^{(r-1)}[A^{(r-1)}]^{-1}B^{(r-1)}, \\ B^{(r)} &= -B^{(r-1)}[A^{(r-1)}]^{-1}B^{(r-1)}, \\ \text{and } \vec{d}_j^{(r)} &= \vec{d}_j^{(r-1)} - B^{(r-1)}[A^{(r-1)}]^{-1}[\vec{d}_{j-2^{r-1}}^{(r-1)} + \vec{d}_{j+2^{r-1}}^{(r-1)}], \end{aligned}$$

where j ranges over the multiples of 2^r that are less than m.

4.2 The Back-substitution Phase

The development of the back-substitution for block tridiagonal systems is completely analogous to the development presented in Chapter 2 for tridiagonal systems. So, let's look at the back-substitution through the case where $m = 7$.

When $m = 7$, the reduction is finished in two steps, and the final equation is

$$A^{(2)}\vec{x}_4 = \vec{d}_4^{(2)}. \tag{4.9}$$

The method chosen to solve this system will depend on the dimension of $A^{(2)}$, which is n. For this discussion, we will assume that a direct solution will be a reasonable choice.

Solving this system directly, we would find the inverse of $A^{(2)}$ and multiply both sides of (4.9) on the left by this inverse. The lone variable then becomes

$$\vec{x}_4 = [A^{(2)}]^{-1}\vec{d}_4^{(2)}. \tag{4.10}$$

This value of $\vec{x}_4$ will then form the basis of the back-substitution routine.

Moving to the first reduction step, we find that the original system was transformed into

$$T^{(1)}\vec{x}^{(1)} = \begin{bmatrix} A^{(1)} & B^{(1)} & 0 \\ B^{(1)} & A^{(1)} & B^{(1)} \\ 0 & B^{(1)} & A^{(1)} \end{bmatrix} \begin{bmatrix} \vec{x}_2 \\ \vec{x}_4 \\ \vec{x}_6 \end{bmatrix} = \begin{bmatrix} \vec{d}_2^{(1)} \\ \vec{d}_4^{(1)} \\ \vec{d}_6^{(1)} \end{bmatrix} = \vec{d}^{(1)}. \tag{4.11}$$

So we can solve this system for $\vec{x}_2$ and $\vec{x}_6$ by solving the equations

$$A^{(1)}\vec{x}_2 + B^{(1)}\vec{x}_4 = \vec{d}_2^{(1)} \tag{4.12}$$

$$B^{(1)}\vec{x}_4 + A^{(1)}\vec{x}_6 = \vec{d}_6^{(1)}. \tag{4.13}$$

Doing this, we find

$$\vec{x}_2 = [A^{(1)}]^{-1}[\vec{d}_2^{(1)} - B^{(1)}\vec{x}_4], \tag{4.14}$$

$$\vec{x}_6 = [A^{(1)}]^{-1}[\vec{d}_6^{(1)} - B^{(1)}\vec{x}_4]. \tag{4.15}$$

Finally, we may use the original system,

$$T\vec{x} = \begin{bmatrix} A & B & 0 & 0 & 0 & 0 & 0 \\ B & A & B & 0 & 0 & 0 & 0 \\ 0 & B & A & B & 0 & 0 & 0 \\ 0 & 0 & B & A & B & 0 & 0 \\ 0 & 0 & 0 & B & A & B & 0 \\ 0 & 0 & 0 & 0 & B & A & B \\ 0 & 0 & 0 & 0 & 0 & B & A \end{bmatrix} \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \\ \vec{x}_5 \\ \vec{x}_6 \\ \vec{x}_7 \end{bmatrix} = \begin{bmatrix} \vec{d}_1 \\ \vec{d}_2 \\ \vec{d}_3 \\ \vec{d}_4 \\ \vec{d}_5 \\ \vec{d}_6 \\ \vec{d}_7 \end{bmatrix} = \vec{d}, \tag{4.16}$$

to solve for the odd-indexed variables. The equations of interest are

$$\begin{aligned} A\vec{x}_1 + B\vec{x}_2 &= \vec{d}_1 \\ B\vec{x}_2 + A\vec{x}_3 + B\vec{x}_4 &= \vec{d}_3 \\ B\vec{x}_4 + A\vec{x}_5 + B\vec{x}_6 &= \vec{d}_5 \\ B\vec{x}_6 + A\vec{x}_7 &= \vec{d}_7. \end{aligned}$$

Solving these equations for the unknown variables, we find

$$\begin{aligned} \vec{x}_1 &= A^{-1}(\vec{d}_1 - B\vec{x}_2) \\ \vec{x}_3 &= A^{-1}[\vec{d}_3 - B(\vec{x}_2 + \vec{x}_4)] \\ \vec{x}_5 &= A^{-1}[\vec{d}_5 - B(\vec{x}_4 + \vec{x}_6)] \\ \vec{x}_7 &= A^{-1}(\vec{d}_7 - B\vec{x}_6). \end{aligned}$$

4.2.1 Example (back-substitution phase only)

Let's return to Example 4.1.1. The final equation after the reduction was

$$\begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000002 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix} \vec{x}_4 = \begin{bmatrix} 1.857142858 \\ 0.3750000001 \\ 0.1428571429 \end{bmatrix} \tag{4.17}$$

Solving this equation for $\vec{x}_4$, we find

$$\vec{x}_4 = [\ -0.7044655033 \quad -0.4398060869 \quad -0.2096201423\ ]^T.$$

We can now substitute this value into the system defined by $A^{(1)}$, $B^{(1)}$, and $\vec{d}_j^{(1)}$. This allows us to find $\vec{x}_2$ and $\vec{x}_6$, which are

$$\begin{aligned} \vec{x}_2 &= [\ -0.6290769962 \quad -0.3621061140 \quad -0.1651594705\ ]^T \\ \text{and} \quad \vec{x}_6 &= [\ -0.6290769962 \quad -0.3621061140 \quad -0.1651594705\ ]^T. \end{aligned}$$

Finally, we can substitute the three known values into the original system to solve for the odd-indexed variables. These variables are

$$\begin{aligned} \vec{x}_1 &= [\ -0.4651739083 \quad -0.2316186373 \quad -0.09919452695\ ]^T, \\ \vec{x}_3 &= [\ -0.6890279628 \quad -0.4225693511 \quad -0.1993372410\ ]^T, \\ \vec{x}_5 &= [\ -0.6890279628 \quad -0.4225693511 \quad -0.1993372410\ ]^T, \\ \text{and } \vec{x}_7 &= [\ -0.4651739083 \quad -0.2316186373 \quad -0.09919452695\ ]^T. \end{aligned}$$

4.2.2 Some observations about the back-substitution

As we have done before, let r be the depth of the reduction of the system. That is, (4.9) has a depth of $r = 2$, (4.11) has a depth of $r = 1$, and (4.16) has a depth of $r = 0$. Similar to Chapter 2, we can make the following notes about back-substitution in the block case.

1 At the rth stage of the back-substitution, we are finding the value of the vectors $\vec{x}_j$, where j ranges over the odd multiples of 2^r that are less than or equal to m.

2 For each $\vec{x}_j$, $\vec{x}_j = [A^{(r)}]^{-1}[\vec{d}_j^{(r)} - B^{(r)}(\vec{x}_{j-2^r} + \vec{x}_{j+2^r})]$.

We should note that the second item requires dummy variables $\vec{x}_0$ and $\vec{x}_{m+1}$, both of which must be set equal to $\vec{0}$.

4.3 The Complete Cyclic Reduction Algorithm

The development and observations we made in the two previous sections can be combined into the following algorithm for complete cyclic reduction. (In an effort to bridge the gap between the above development and the writing of computer programs, the superscripts on A and B have become subscripts. Also, the superscripts and subscripts on $\vec{d}$ have been combined into a two-element subscript such that $\vec{d}_{j,r}$ is the jth component vector of $\vec{d}^{(r)}$. Finally, the back substitution relies on the fact that $\vec{x}_0 = \vec{x}_{m+1} = \vec{0}$).

THE REDUCTION PHASE

$s = 1$
$A_0 = A$
$B_0 = B$
for $r = 1, \ldots, k-1$
  $M_r = -B_{r-1}A_{r-1}^{-1}$
  $B_r = M_r B_{r-1}$
  $A_r = A_{r-1} + 2B_r$
  $t = s$
  $s = 2s$
  for $j = s, 2s, \ldots, m+1-s$
    $\vec{d}_{j,r} = \vec{d}_{j,r-1} + M_r(\vec{d}_{j-t,r-1} + \vec{d}_{j+t,r-1})$
  end
end

THE BACK-SUBSTITUTION PHASE

$s = (n+1)/2$
$\vec{x}_s = A_{k-1}^{-1}\vec{d}_{s,k-1}$
for $i = k-2, k-3, \ldots, 0$
  $t = s$
  $s = s/2$
  for $j = s, s+t, \ldots, m+1-s$
    $\vec{x}_j = A_i^{-1}[\vec{d}_{k,i} - B_i(\vec{x}_{j-s} + \vec{x}_{j+s})]$
  end
end

4.3.1 Example

Let's demonstrate the algorithm on a system with $m = 1023$ and $n = 3$. Notice that $k = 10$ and the overall dimension of T is 3069×3069. For this example, let

$$A = \begin{bmatrix} -4 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & -4 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\text{and } \vec{d}_j = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix},$$

for $j = 1, 2, \ldots, 1023$.

Initially, we set

$$\begin{aligned} A_0 &= A, \\ B_0 &= B, \\ \text{and } s &= 1. \end{aligned}$$

When $r = 1$, the algorithm produces the updates

$$\begin{aligned} M_1 &= \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}, \\ B_1 &= \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}, \\ A_1 &= \begin{bmatrix} -3.464285714 & 1.142857143 & 0.03571428572 \\ 1.142857143 & -3.423571429 & 1.142857143 \\ 0.03571428572 & 1.142857143 & -3.464285714 \end{bmatrix}, \\ \text{and } \vec{d}_{j,1} &= \begin{bmatrix} 1.714285714 & 1.857142857 & 1.714285714 \end{bmatrix}^T, \end{aligned}$$

for $j = 2, 4, \ldots, 1022$.

For $r = 2$, we find

$$\begin{aligned} M_2 &= \begin{bmatrix} 0.09821428575 & 0.06250000004 & 0.02678571431 \\ 0.06250000003 & 0.1250000000 & 0.06250000003 \\ 0.02678571430 & 0.06250000004 & 0.09821428574 \end{bmatrix}, \\ B_2 &= \begin{bmatrix} 0.03125000002 & 0.02678571431 & 0.01339285716 \\ 0.02678571430 & 0.04464285715 & 0.02678571430 \\ 0.01339285715 & 0.02678571430 & 0.03125000001 \end{bmatrix}, \\ A_2 &= \begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000004 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix}, \\ \text{and } \vec{d}_{j,2} &= \begin{bmatrix} 2.375000000 & 2.750000000 & 2.375000000 \end{bmatrix}^T, \end{aligned}$$

for $j = 4, 8, \ldots, 1020$.

Continuing when $r = 3$, our updates become

$$\begin{aligned} M_3 &= \begin{bmatrix} 0.01543754045 & 0.01723673580 & 0.01028290127 \\ 0.01723673580 & 0.02572044169 & 0.01723673580 \\ 0.01028290126 & 0.01723673580 & 0.01543754044 \end{bmatrix}, \\ B_3 &= \begin{bmatrix} 0.001081838848 & 0.001458437538 & 0.0009897917195 \\ 0.001458437537 & 0.002071630566 & 0.001458437537 \\ 0.0009897917191 & 0.001458437538 & 0.001081838847 \end{bmatrix}, \\ A_3 &= \begin{bmatrix} -3.399622036 & 1.199345447 & 0.06447958348 \\ 1.199345447 & -3.335142454 & 1.199345447 \\ 0.06447958348 & 1.199345447 & -3.399622036 \end{bmatrix}, \\ \text{and } \vec{d}_{j,3} &= \begin{bmatrix} 2.591974145 & 3.055211419 & 2.591974145 \end{bmatrix}^T, \end{aligned}$$

for $j = 8, 16, \ldots, 1016$.

When $r = 4$, we have

$$M_4 = \begin{bmatrix} 0.0006443266570 & 0.0008911491475 & 0.0006177549398 \\ 0.0008911491471 & 0.001262081595 & 0.0008911491471 \\ 0.0006177549397 & 0.0008911491472 & 0.0006443266566 \end{bmatrix},$$

$$B_4 = \begin{bmatrix} 2.608191700 & 3.686798990 & 2.605745850 \\ 3.686798986 & 5.213937545 & 3.686798986 \\ 2.605745850 & 3.686798988 & 2.608191698 \end{bmatrix} \times 10^{-6},$$

$$A_4 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479497 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479495 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,4} = \begin{bmatrix} 2.603962009 & 3.072162613 & 2.603962009 \end{bmatrix}^T,$$

for $j = 16, 32, \ldots, 1008$.

Next, when $r = 5$, the updates are

$$M_5 = \begin{bmatrix} 1.590945050 & 2.249434502 & 1.590238994 \\ 2.249434500 & 3.181184040 & 2.249434500 \\ 1.590238992 & 2.249434500 & 1.590945049 \end{bmatrix} \times 10^{-6},$$

$$B_5 = \begin{bmatrix} 1.658646118 & 2.345679711 & 1.658645944 \\ 2.345679709 & 3.317292058 & 2.345679709 \\ 1.658645943 & 2.345679710 & 1.658646116 \end{bmatrix} \times 10^{-11},$$

$$A_5 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,5} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 32, 64, \ldots, 992$.

For the sixth reduction, $r = 6$, and

$$M_6 = \begin{bmatrix} 1.011992178 & 1.431173027 & 1.011992128 \\ 1.431173026 & 2.023984303 & 1.431173026 \\ 1.011992127 & 1.431173027 & 1.011992177 \end{bmatrix} \times 10^{-11},$$

$$B_6 = \begin{bmatrix} 6.714147064 & 9.495237837 & 6.714147064 \\ 9.495237828 & 13.42829411 & 9.495237827 \\ 6.714147062 & 9.495237832 & 6.714147061 \end{bmatrix} \times 10^{-22},$$

$$A_6 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,6} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 64, 128, \ldots, 960$.

When $r = 7$,

$$M_7 = \begin{bmatrix} 4.096512468 & 5.793343488 & 4.096512468 \\ 5.793343483 & 8.193024921 & 5.793343483 \\ 4.096512466 & 5.793343484 & 4.096512466 \end{bmatrix} \times 10^{-22},$$

$$B_7 = \begin{bmatrix} 1.100183486 & 1.555894406 & 1.100183485 \\ 1.555894404 & 2.200366967 & 1.555894404 \\ 1.100183485 & 1.555894406 & 1.100183485 \end{bmatrix} \times 10^{-42},$$

$$A_7 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,7} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 128, 256, \ldots, 896$.

For $r = 8$,

$$M_8 = \begin{bmatrix} 6.712565755 & 9.493001523 & 6.712565752 \\ 9.493001513 & 13.42513148 & 9.493001513 \\ 6.712565752 & 9.493001522 & 6.712565752 \end{bmatrix} \times 10^{-43},$$

$$B_8 = \begin{bmatrix} 2.954021592 & 4.177617398 & 2.954021592 \\ 4.177617393 & 5.908043174 & 4.177617392 \\ 2.954021592 & 4.177617397 & 2.954021591 \end{bmatrix} \times 10^{-84},$$

$$A_8 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,8} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 256, 512, 768$.

Finally, when $r = 9$,

$$A_9 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{512,8} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T.$$

So, our original system has been reduced to the equation

$$\text{(4.18)} \qquad \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix} \vec{x}_{512} = \begin{bmatrix} 2.603992398 \\ 3.072205589 \\ 2.603992398 \end{bmatrix}.$$

To begin the back-substitution phase of the algorithm, we solve (4.18) for $\vec{x}_{512}$. This value then becomes the starting point for the back-substitution routine.

In the first back-substitution, we find values for $\vec{x}_{256}$ and $\vec{x}_{768}$. The three known values are then used in a second round of back-substitution to find $\vec{x}_{128}$, $\vec{x}_{384}$, $\vec{x}_{640}$, and $\vec{x}_{896}$. This process continues

until the eighth set of back-substitutions yields the values for the odd-indexed vectors. The final solution is shown in the Maple 9.5 output in Appendix G.

Once the back-substitution was done, the routine checked the solution. The error, which was measured by taking the infinity norm of $T\vec{x} - \vec{d}$, was found to be 6×10^{-9}. It took Maple 9.5 a total of 79.7 seconds to run the routine, including the check.

4.4 The Stability of the Algorithm

At every step of Algorithm 4.3, we multiply by A_r^{-1}. So, the algorithm will break down if A_r is singular at any point in the reduction. In order to show that A_r^{-1} exists, we first need to define the vector and matrix infinity norms

$$\begin{aligned} \|\vec{v}\|_\infty &= \max_{j=1,\ldots,p} |v_j| \\ \text{and } \|M\|_\infty &= \max_{i=1,\ldots,p} \sum_{j=1}^{q} |m_{ij}| \end{aligned}$$

and the matrix one norm

$$\|M\|_1 = \max_{j=1,\ldots,q} \sum_{i=1}^{p} |m_{ij}|,$$

where $\vec{v}$ is a p-vector and M is a $p \times q$ matrix. It should be noted that the matrix infinity-, one-, and two-norms are natural (or induced) norms and have the property $\|AB\| \leq \|A\| \cdot \|B\|$ [8]. For convenience, we will also define

$$\beta_i(M) = \sum_{j=1}^{q} |m_{ij}|,$$

for $i = 1, \ldots, p$. Notice that this makes the definition of the matrix infinity norm

$$\|M\|_\infty = \max_{i=1,\ldots,p} [\beta_i(M)].$$

To show that A_r is not singular, we will need to consider the norms of the block tridiagonal matrices

$$\begin{aligned} C_r &= (-A_r^{-1}B_r, 0, -A_r^{-1}B_r) \\ \text{and } D_r &= (-B_rA_r^{-1}, 0, -B_rA_r^{-1}). \end{aligned}$$

The following theorem was initially stated by Heller [9].

Theorem 4.1 *Suppose A^{-1} exists, $\|C_0\|_\infty < 1$, and $\|D_0\|_1 < 1$. Then,*

1. *A_r^{-1} exists for all $r \leq k$,*
2. *$\|C_{r+1}\|_\infty \leq \|C_r^2\|_\infty < 1$ for all $r < k$, and*
3. *$\|D_{r+1}\|_1 \leq \|D_r^2\|_1 < 1$ for all $r < k$.*

Proof by induction: Define

$$\begin{aligned} \sigma_{r,1} &= A_r^{-1}B_rA_r^{-1}B_r, \\ \sigma_{r,j} &= 2A_r^{-1}B_rA_r^{-1}B_r, \text{ for } 1 < j < m_r, \\ \sigma_{r,m_r} &= A_r^{-1}B_rA_r^{-1}B_r, \\ \tau_{r,j} &= A_r\sigma_{r,j}A_r^{-1}. \end{aligned}$$

Notice that $\sigma_{0,j}$ is the diagonal block of the jth block row of C_0^2. So,

$$\begin{aligned} \|\sigma_{0,j}\|_\infty &\leq \|C_0^2\|_\infty \\ &\leq \|C_0\|_\infty^2 \\ &< 1. \end{aligned}$$

Recall that

$$(I - \sigma_{0,j})^{-1} = I + \sigma_{0,j} + \sigma_{0,j}^2 + \cdots,$$

which converges when the spectral radius of $\sigma_{0,j} < 1$. However, the spectral radius is less than any of the compatible norms; therefore, $(I - \sigma_{0,j})^{-1}$ exists.

Also, $\tau_{0,j}$ is the diagonal block of the jth block column of D_0^2. So,

$$\begin{aligned} \|\tau_{0,j}\|_1 &\leq \|D_0^2\|_1 \\ &\leq \|D_0\|_1^2 \\ &< 1. \end{aligned}$$

So, by a similar argument as above, $(I - \tau_{0,j})^{-1}$ exists. Because $(I - \sigma_{0,j})^{-1}$, $(I - \tau_{0,j})^{-1}$, and A^{-1} all exist and are all $n \times n$ matrices, $(I - \sigma_{0,j})^{-1}A^{-1}$ and $A^{-1}(I - \tau_{0,j})^{-1}$ exist. However, for $1 < j < m_r$,

$$\begin{aligned} (I - \sigma_{0,j})^{-1}A^{-1} &= [A(I - \sigma_{0,j})]^{-1} \\ &= [A(I - 2A^{-1}BA^{-1}B)]^{-1} \\ &= (A - 2BA^{-1}B)^{-1} \\ &= A_1^{-1}. \end{aligned}$$

Note also,

$$\begin{aligned} A^{-1}(I - \tau_{0,j})^{-1} &= [(I - \tau_{0,j})A]^{-1} \\ &= [(I - A\sigma_{0,j}A^{-1})A]^{-1} \\ &= (A - A\sigma_{0,j})^{-1} \\ &= A_1^{-1}. \end{aligned}$$

Therefore, by either condition, A_1^{-1} exists, and statement 1 holds for $n = 1$.

Now, define

$$\begin{aligned} E_{r,1} &= 0, \\ E_{r,j} &= -B_r A_r^{-1} B_r, \text{ for } 1 < j \leq m_r, \\ F_{r,j} &= A_r(I - \sigma_{r,j}) = (I - \tau_{r,j})A_r, \text{ for } 1 \leq j \leq m_r, \\ G_{r,j} &= -B_r A_r^{-1} B_r, \text{ for } 1 \leq j < m_r, \\ G_{r,m_r} &= 0, \\ \text{and} \quad S_r &= \text{diag}(\sigma_{r,j}), \text{where } 1 \leq j \leq m_r. \end{aligned}$$

Also, define the block pentadiagonal matrix

$$J_r = (-F_r^{-1}E_r, 0, 0, 0, -F_r^{-1}G_r).$$

From the above definitions, we have

$$\begin{aligned} \|S_0\|_\infty &\leq \max_{j=1,\ldots,m} \|\sigma_{0,j}\|_\infty \\ &\leq \|C_0^2\|_\infty \\ &< 1. \end{aligned}$$

Also,

$$\begin{aligned} E_{0,1} &= 0, \\ E_{0,j} &= -BA^{-1}B \\ &= B_1 \qquad \text{for } j = 2,\ldots,m, \\ F_{0,1} &= A(I - \sigma_{0,1}) \\ &= A - BA^{-1}B \\ &= A_1 - B_1 \\ F_{0,j} &= A(I - \sigma_{0,j}) \\ &= A - 2BA^{-1}B \\ &= A_1 \qquad \text{for } j = 2,\ldots,m-1, \\ F_{0,m} &= A_1 - B_1 \\ G_{0,j} &= -BA^{-1}B \\ &= B_1 \qquad \text{for } j = 1,\ldots,m-1, \end{aligned}$$

$$\text{and} \quad G_{0,m} = 0.$$

Substituting the above matrices into the definition of J_0, we find

$$\begin{aligned} \|J_0\|_\infty &= \max_{j=1,\ldots,m_r} (\|F_{0,j}^{-1}(E_{0,j} + G_{0,j})\|_\infty) \\ &= \max(\|(A_1 - B_1)^{-1}B_1\|_\infty, 2\|A_1^{-1}B_1\|_\infty) \\ &\geq 2\|A_1^{-1}B_1\|_\infty \\ &= \|C_1\|_\infty. \end{aligned}$$

Now, note that the multiplication $(I - S_0)J_0$ gives the block pentadiagonal matrix

$$(-(I - \sigma_{0,j})F_{0,j}^{-1}E_{0,j}, 0, 0, 0, -(I - \sigma_{0,j})F_{0,j}^{-1}G_{0,j}).$$

When $j = 2,\ldots,m-1$, we see that

$$\begin{aligned} -(I - \sigma_{0,j})F_{0,j}^{-1}E_{0,j} &= -(I - 2A^{-1}BA^{-1}B)(A_1^{-1})(B_1) \\ &= -(A^{-1}A_1)(A_1^{-1})(B_1) \\ &= -A^{-1}B_1 \\ &= A^{-1}BA^{-1}B. \end{aligned}$$

Also, because $E_{0,j} = G_{0,j}$ for $j = 2, \ldots, m-1$,

$$-(I - \sigma_{0,j})F_{0,j}^{-1}G_{0,j} = A^{-1}BA^{-1}B.$$

These are exactly the non-zero off-diagonal blocks of C_0^2 in rows 2 through $m-1$.

For $j = 1$, we have

$$\begin{aligned} -(I - \sigma_{0,1})F_{0,1}^{-1}G_{0,1} &= -(I - A^{-1}BA^{-1}B)(A - BA^{-1}B)^{-1}(B_1) \\ &= -(I - A^{-1}BA^{-1}B)[A(I - A^{-1}BA^{-1}B)]^{-1}(B_1) \\ &= -(I - A^{-1}BA^{-1}B)(I - A^{-1}BA^{-1}B)^{-1}(A^{-1})(B_1) \\ &= A^{-1}B_1 \\ &= A^{-1}BA^{-1}B, \end{aligned}$$

which is the non-zero off-diagonal block of row 1 of C_0^2.

Finally, notice that $E_{0,m} = G_{0,1}$, $F_{0,m} = F_{0,1}$, and $\sigma_{0,m} = \sigma_{0,1}$, all by definition. So,

$$-(I - \sigma_{0,m})F_{0,m}^{-1}E_{0,m} = A^{-1}BA^{-1}B.$$

Again, this the non-zero off-diagonal block of row m in C_0^2.

We know from before that S_0 is the block diagonal part of C_0^2, and we have seen that the block diagonal portion of $(I - S_0)J_0$ is null. Thus,

$$C_0^2 = S_0 + (I - S_0)J_0,$$

and we can separate $\beta_l(C_0^2)$ into two mutually exclusive parts. Hence, for each l,

$$\begin{aligned} \|C_0^2\|_\infty &\geq \beta_l(C_0^2) \\ &= \beta_l(S_0) + \beta_l[(I - S_0)J_0] \\ &\geq \beta_l(S_0) + \beta_l(J_0) - \beta_l(S_0)\|J_0\|_\infty \\ &= \beta_l(J_0) + \beta_l(S_0)(1 - \|J_0\|_\infty). \end{aligned}$$

Now, suppose $\|J_0\|_\infty = 1$. Then, for some l, $\|J_0\|_\infty = \beta_l(J_0) = 1$, and

$$\begin{aligned} 1 &> \|C_0^2\|_\infty \\ &\geq \beta_l(J_0) + \beta_l(S_0)(1 - 1) \\ &= \beta_l(J_0) \\ &= 1, \end{aligned}$$

a contradiction.

If $\|J_0\|_\infty > 1$, then for some l, $\|J_0\|_\infty = \beta_l > 1$, and

$$\begin{aligned} 1 &> \|C_0^2\|_\infty \\ &\geq \beta_l(J_0) + \beta_l(S_0)(1 - \|J_0\|_\infty). \end{aligned}$$

This implies

$$\begin{aligned} 1 - \beta_l(J_0) &> \beta_l(S_0)(1 - \|J_0\|_\infty) \\ \beta_l(S_0) &> \frac{1 - \beta_l(J_0)}{1 - \|J_0\|_\infty} \\ &\geq 1. \end{aligned}$$

However, we showed earlier that $\|S_0\|_\infty < 1$. Thus,

$$\begin{aligned} 1 &> \|S_0\|_\infty \\ &\geq \beta_l(S_0) \quad \text{for } l = 1, \ldots, m \\ &\geq 1, \end{aligned}$$

a contradiction. Therefore, $\|J_0\|_\infty < 1$, and thus, for all $l = 1, \ldots, m$,

$$\begin{aligned} \|C_0^2\|_\infty &\geq \beta_l(J_0) + \beta_l(S_0)(1 - \|J_0\|_\infty) \\ &> \beta_l(J_0). \end{aligned}$$

Hence, $1 > \|C_0^2\|_\infty > \|J_0\|_\infty \geq \|C_1\|_\infty$, and statement 2 holds for $n = 1$.

Finally, define the block diagonal matrix

$$T_r = \text{diag}(\tau_r)$$

and the block pentadiagonal matrix

$$K_r = (-E_{r,j}F_{r,j-2}^{-1}, 0, 0, 0, -G_{r,j}F_{r,j+2}^{-1}).$$

From the above definition, we know

$$\begin{aligned} \|T_0\|_1 &= \max_{j=1,\ldots,m} \|\tau_{0,j}\|_1 \\ &\leq \|D_0^2\|_1 \\ &< 1. \end{aligned}$$

Substituting the values of $E_{0,j}$, $F_{0,j}$, and $G_{0,j}$ into the definition of K_0, we find

$$\begin{aligned} \|K_0\|_1 &\geq 2\|B_1 A_1^{-1}\|_1 \\ &= \|D_1\|_1. \end{aligned}$$

The multiplication $K_0(I - T_0)$ gives the block pentadiagonal matrix

$$(-E_{0,j}F_{0,j-2}^{-1}(I - \tau_{0,j-2}), 0, 0, 0, -G_{0,j}F_{0,j+2}^{-1}(I - \tau_{0,j+2})).$$

For $j = 3$,

$$\begin{aligned} -E_{0,3}F_{0,1}(I - \tau_{0,1}) &= (BA^{-1}B)(A - BA^{-1}B)(I - A\sigma_{0,1}A^{-1}) \\ &= (BA^{-1}B)[(I - BA^{-1}BA^{-1})A]^{-1}(I - AA^{-1}BA^{-1}BA^{-1}) \\ &= BA^{-1}BA^{-1}. \end{aligned}$$

When $j = 4, \ldots, m$,

$$\begin{aligned} -E_{0,j}F_{0,j-2}^{-1}(I - \tau_{0,j-2}) &= (BA^{-1}B)(A - 2BA^{-1}B)^{-1}(I - 2AA^{-1}BA^{-1}BA^{-1}) \\ &= (BA^{-1}B)[(I - 2BA^{-1}BA^{-1})A]^{-1}(I - 2BA^{-1}BA^{-1}) \\ &= BA^{-1}BA^{-1}. \end{aligned}$$

For $j = 1, \ldots, m - 3$,

$$\begin{aligned} -G_{0,j}F_{0,j+2}^{-1}(I - \tau_{0,j+2}) &= (BA^{-1}B)(A - 2BA^{-1}B)^{-1}(I - 2AA^{-1}BA^{-1}BA^{-1}) \\ &= (BA^{-1}B)[(I - 2BA^{-1}BA^{-1})A]^{-1}(I - 2BA^{-1}BA^{-1}) \\ &= BA^{-1}BA^{-1}. \end{aligned}$$

Finally, for $j = m-2$,

$$\begin{aligned} -G_{0,m-2}F_{0,m}^{-1}(I-\tau_{0,m}) &= (BA^{-1}B)(A-BA^{-1}B)(I-A\sigma_{0,m}A^{-1}) \\ &= (BA^{-1}B)[(I-BA^{-1}BA^{-1})A]^{-1}(I-AA^{-1}BA^{-1}BA^{-1}) \\ &= BA^{-1}BA^{-1}. \end{aligned}$$

Notice that these are the non-zero off-diagonal blocks of D_0^2. Also, the block diagonal part of $K_0(I-T_0)$ is null, and T_0 is the block diagonal part of D_0^2. So,

$$D_0^2 = K_0(I-T_0)+T_0.$$

Notice from the above exposition that D_0^2 and K_0 are symmetric. Thus,

$$\|D_0^2\|_1 = \|D_0^2\|_\infty,$$

and

$$\|K_0\|_1 = \|K_0\|_\infty.$$

So, by an argument similar to the one for statement 2, we know that statement 3 is true, and the theorem holds for the case $n = 1$.

Now suppose that statements 1 through 3 hold through $n = r$. Then, $\sigma_{r,j}$ is the diagonal block of the jth block row of C_r^2. So,

$$\begin{aligned} \|\sigma_{r,j}\|_\infty &\leq \|C_r^2\|_\infty \\ &\leq \|C_r\|_\infty^2 \\ &< 1. \end{aligned}$$

Thus, $(I-\sigma_{r,j})^{-1}$ exists.

Also, $\tau_{r,j}$ is the diagonal block of the jth block column of D_r^2. So,

$$\begin{aligned} \|\tau_r\|_1 &\leq \|D_r^2\|_1 \\ &\leq \|D_r\|_1^2 \\ &< 1. \end{aligned}$$

So, $(I-\tau_{r,j})^{-1}$ exists. Because $(I-\sigma_{r,j})^{-1}$, $(I-\tau_{r,j})^{-1}$, and A_r^{-1} exist, $(I-\sigma_{r,j})^{-1}A_r^{-1}$ and $A_r^{-1}(I-\tau_{r,j})^{-1}$ exist. However,

$$\begin{aligned} (I-\sigma_{r,j})^{-1}A_r^{-1} &= [A_r(I-\sigma_{r,j})]^{-1} \\ &= [A_r(I-2A_r^{-1}B_rA_r^{-1}B_r)]^{-1} \\ &= (A_r-2B_rA_r^{-1}B_r)^{-1} \\ &= A_{r+1}^{-1}. \end{aligned}$$

Similarly,

$$\begin{aligned} A_r^{-1}(I-\tau_{r,j})^{-1} &= [(I-\tau_{r,j})A_r]^{-1} \\ &= [(I-A_r\sigma_{r,j}A_r^{-1})A_r]^{-1} \\ &= (A_r-A_r\sigma_{r,j})^{-1} \\ &= A_{r+1}^{-1}. \end{aligned}$$

Thus, by either condition, A_{r+1}^{-1} exists.

By definition, we have

$$\begin{aligned} \|S_r\|_\infty &= \max_{j=1,\ldots,m_r} \|\sigma_{r,j}\|_\infty \\ &\leq \|C_r^2\|_\infty \\ &< 1. \end{aligned}$$

Also,

$$\begin{aligned} E_{r,1} &= 0 \\ E_{r,j} &= -B_r A_r^{-1} B_r \\ &= B_{r+1} \qquad \text{for } j = 2,\ldots,m_r, \end{aligned}$$

$$\begin{aligned} F_{r,1} &= A_r(I - \sigma_{r,1}) \\ &= A_r - B_r A_r^{-1} B_r, \\ &= A_{r+1} - B_{r+1} \end{aligned}$$

$$\begin{aligned} F_{r,j} &= A_r(I - \sigma_{r,j}) \\ &= A_r - 2B_r A_r^{-1} B_r \\ &= A_{r+1} \qquad \text{for } j = 2,\ldots,m_r - 1, \end{aligned}$$

$$\begin{aligned} F_{r,m_r} &= A_r - B_r A_r^{-1} B_r, \\ &= A_{r+1} - B_{r+1} \end{aligned}$$

$$\begin{aligned} G_{r,j} &= -B_r A_r^{-1} B_r \\ &= B_{r+1} \qquad \text{for } j = 1,\ldots,m_r - 1, \end{aligned}$$

$$\text{and} \quad G_{r,m_r} = 0.$$

Substituting these matrices into the definition of J_r, we find

$$\begin{aligned} \|J_r\|_\infty &\geq 2\|A_{r+1}^{-1} B_{r+1}\|_\infty \\ &= \|C_{r+1}\|_\infty. \end{aligned}$$

Now, the multiplication $(I - S_r)J_r$ gives the block pentadiagonal matrix

$$(-(I - \sigma_{r,j})F_{r,j}^{-1} E_{r,j}, 0, 0, 0, -(I - \sigma_{r,j})F_{r,j}^{-1} G_{r,j}).$$

When $j = 2,\ldots,m_r - 1$, we find that

$$\begin{aligned} -(I - \sigma_{r,j})F_{r,j}^{-1} E_{r,j} &= -(I - 2A_r^{-1} B_r A_r^{-1} B_r)(A_{r+1}^{-1})(B_{r+1}) \\ &= (A_r^{-1} A_{r+1})(A_{r+1})(-B_r A_r^{-1} B_r) \\ &= A_r^{-1} B_r A_r^{-1} B_r. \end{aligned}$$

Also, because $E_{r,j} = G_{r,j}$ for $j = 2,\ldots,m_r - 1$,

$$-(I - \sigma_{r,j})F_{r,j}^{-1} G_{r,j} = A_r^{-1} B_r A_r^{-1} B_r.$$

For $j = 1$, we see that

$$\begin{aligned} -(I - \sigma_{r,1})F_{r,1}^{-1}G_{r,1} &= -(I - A_r^{-1}B_rA_r^{-1}B_r)(A_r - B_rA_r^{-1}B_r)^{-1}(B_{r+1}) \\ &= -(I - A_r^{-1}B_rA_r^{-1}B_r)(I - A_r^{-1}B_rA_r^{-1}B_r)^{-1}(A_r^{-1})(B_{r+1}) \\ &= A_r^{-1}B_rA_r^{-1}B_r. \end{aligned}$$

Lastly, we note that $E_{r,m_r} = G_{r,1}$, $F_{r,m_r} = F_{r,1}$, and $\sigma_{r,m_r} = \sigma_{r,1}$ by definition. Thus,

$$-(I - \sigma_{r,j})F_{r,m_r}^{-1}E_{r,m_r} = A_r^{-1}B_rA_r^{-1}B_r.$$

These are the non-zero off-diagonal blocks of C_r^2 in rows 1 through m_r.

We can see that S_r is the block diagonal part of C_r^2 and that the block diagonal portion of $(I - S_r)J_r$ is null. Therefore,

$$C_r^2 = S_r + (I - S_r)J_r,$$

and we can separate $\beta_l(C_r^2)$ into two mutually exclusive parts. So, for each l,

$$\begin{aligned} \|C_r^2\|_\infty &\geq \beta_l(C_r^2) \\ &= \beta_l(S_r) + \beta_l[(I - S_r)J_r] \\ &\geq \beta_l(S_r) + \beta_l(J_r) - \beta_l(S_r)\|J_r\|_\infty \\ &= \beta_l(J_r) + \beta_l(S_r)(1 - \|J_r\|_\infty). \end{aligned}$$

By an argument similar to the $n = 1$ case, we know that $\|J_r\|_\infty < 1$. Hence, for all $l = 1, \ldots, m_r$, we know that $\|C_r^2\|_\infty > \beta_l(J_r)$, which implies $1 > \|C_r^2\|_\infty > \|J_r\|_\infty \geq \|C_{r+1}\|_\infty$. Thus, statement 2 holds for $n = r + 1$.

Finally, we know

$$\begin{aligned} \|T_r\|_1 &= \max_{j=1,\ldots,m_r} \|\tau_{r,j}\|_1 \\ &\leq \|D_r^2\|_1 \\ &< 1. \end{aligned}$$

Substituting the values of $E_{r,j}$, $F_{r,j}$, and $G_{r,j}$ into the definition of K_r, we find

$$\begin{aligned} \|K_r\|_1 &\geq 2\|B_rA_r^{-1}\|_1 \\ &= \|D_1\|_1. \end{aligned}$$

The multiplication $K_r(I - T_r)$ gives the block pentadiagonal matrix

$$(-E_{r,j}F_{r,j-2}^{-1}(I - \tau_{r,j-2}), 0, 0, 0, -G_{r,j}F_{r,j+2}^{-1}(I - \tau_{r,j+2})).$$

For $j = 3$,

$$\begin{aligned} -E_{r,3}F_{r,1}(I - \tau_{r,1}) &= (B_rA_r^{-1}B_r)(A_r - B_rA_r^{-1}B_r)(I - A_r\sigma_{r,1}A_r^{-1}) \\ &= (B_rA_r^{-1}B_r)[(I - B_rA_r^{-1}B_rA_r^{-1})A_r]^{-1}(I - A_rA_r^{-1}B_rA_r^{-1}B_rA_r^{-1}) \\ &= B_rA_r^{-1}B_rA_r^{-1}. \end{aligned}$$

When $j = 4, \ldots, m_r$,

$$\begin{aligned} -E_{r,j}F_{r,j-2}^{-1}(I - \tau_{r,j-2}) &= (B_rA_r^{-1}B_r)(A_r - 2B_rA_r^{-1}B_r)^{-1}(I - 2A_rA_r^{-1}B_rA_r^{-1}B_rA_r^{-1}) \\ &= (B_rA_r^{-1}B_r)[(I - 2B_rA_r^{-1}B_rA_r^{-1})A_r]^{-1}(I - 2B_rA_r^{-1}B_rA_r^{-1}) \\ &= B_rA_r^{-1}B_rA_r^{-1}. \end{aligned}$$

For $j = 1, \ldots, m_r - 3$,

$$\begin{aligned} -G_{r,j}F_{r,j+2}^{-1}(I - \tau_{r,j+2}) &= (B_rA_r^{-1}B_r)(A_r - 2B_rA_r^{-1}B_r)^{-1}(I - 2A_rA_r^{-1}B_rA_r^{-1}B_rA_r^{-1}) \\ &= (B_rA_r^{-1}B_r)[(I - 2B_rA_r^{-1}B_rA_r^{-1})A_r]^{-1}(I - 2B_rA_r^{-1}B_rA_r^{-1}) \\ &= B_rA_r^{-1}B_rA_r^{-1}. \end{aligned}$$

Finally, for $j = m_r - 2$,

$$\begin{aligned} -G_{r,m-2}F_{r,m}^{-1}(I - \tau_{r,m}) &= (B_rA_r^{-1}B_r)(A_r - B_rA_r^{-1}B_r)(I - A_r\sigma_{r,m_r}A_r^{-1}) \\ &= (B_rA_r^{-1}B_r)[(I - B_rA_r^{-1}B_rA_r^{-1})A_r]^{-1}(I - A_rA_r^{-1}B_rA_r^{-1}B_rA_r^{-1}) \\ &= B_rA_r^{-1}B_rA_r^{-1}. \end{aligned}$$

Notice that these are the non-zero off-diagonal blocks of D_r^2. Also, the block diagonal part of $K_r(I - T_r)$ is null, and T_r is the block diagonal part of D_r^2. So,

$$D_r^2 = K_r(I - T_r) + T_r.$$

Notice from the above exposition that D_r^2 and K_r are symmetric. Thus,

$$\|D_r^2\|_1 = \|D_r^2\|_\infty,$$

and

$$\|K_r\|_1 = \|K_r\|_\infty.$$

So, by an argument similar to the one for statement 2, we know that statement 3 is true, and the theorem holds for the case $n = r + 1$. **QED.**

4.5 Comments about the algorithm

One of the results of the cyclic reduction algorithm is that A_r and B_r will commute if A and B commute [8].

Theorem 4.2 *Suppose A and B are $n \times n$ matrices such that A^{-1} exists and $AB = BA$. Then, for matrices A_r and B_r produced by the cyclic reduction algorithm, $A_rB_r = B_rA_r$.*

Proof by induction: Let $n = 1$. Then by the algorithm, $B_1 = -B^2A^{-1}$, and $A_1 = A - 2B^2A^{-1}$. (It is easy to show that $AB = BA \Rightarrow A^{-1}B = BA^{-1}$.) This implies

$$\begin{aligned} A_1B_1 &= (A - 2B^2A^{-1})(-B^2A^{-1}) \\ &= -AB^2A^{-1} + 2B^2A^{-1}B^2A^{-1} \\ &= -B^2 - 2B_1^2, \end{aligned}$$

and

$$\begin{aligned} B_1A_1 &= (-B^2A^{-1})(A - 2B^2A^{-1}) \\ &= -B^2A^{-1}A + 2B^2A^{-1}B^2A^{-1} \\ &= -B^2 - 2B_1^2. \end{aligned}$$

Thus, the theorem holds for $n = 1$.

Now assume that the theorem holds for $n = k$. Then, we have

$$\begin{aligned} A_{k+1}B_{k+1} &= (A_k - 2B_kA_k^{-1}B_k)(-B_k)A_k^{-1}B_k \\ &= -A_kB_kA_k^{-1}B_k + 2B_kA_k^{-1}B_kB_kA_k^{-1}B_k \\ &= -B_k^2 - 2B_{k+1}^2, \end{aligned}$$

and

$$\begin{aligned} B_{k+1}A_{k+1} &= (-B_kA_k^{-1}B_k)(A_k - 2B_kA_k^{-1}B_k) \\ &= -B_kA_k^{-1}B_kA_k + 2B_kA_k^{-1}B_kB_kA_k^{-1}B_k \\ &= -B_k^2 - 2B_{k+1}^2. \end{aligned}$$

Therefore, $A_rB_r = B_rA_r$ for all r. **QED.**

4.6 Summary

In Example 4.3.1, notice that the sequence $\{\|B_r\|_\infty\}$ converged to 0 quickly. This means that the changes in A_r and $\vec{d}_{j,r}$ become much smaller as the depth of the reduction increases. In fact, A_r and $\vec{d}_{j,r}$ become constant after the fifth reduction step. Therefore, we should be able to stop the algorithm early and still maintain reasonable accuracy when solving the system numerically.

In this chapter, we developed the cyclic reduction algorithm for block matrices of size $m = 2^k - 1$. However, we also noted that the algorithm might be terminated early due to the rapid convergence of $\{\|B_r\|_\infty\}$ to 0. This truncated algorithm will be developed in the next chapter.

Chapter 5

A Truncated Cyclic Reduction Algorithm for Block Tridiagonal Matrices ($n > 1$)

As shown in the previous chapter, $\|C_r\|_\infty$ converges to 0 quadratically. This convergence should allow us to save costs (as measured by floating point operations) by ending the algorithm early. In this chapter, we will use this convergence to develop a truncated algorithm for block cyclic reduction.

5.1 Revisiting the Reduction Phase

As in Chapter 4, we are interested in solving the system of equations

$$T\vec{x} = \begin{bmatrix} A & B & & 0 \\ B & \ddots & \ddots & \\ & \ddots & \ddots & B \\ 0 & & B & A \end{bmatrix} \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_m \end{bmatrix} = \begin{bmatrix} \vec{d}_1 \\ \vdots \\ \vec{d}_m \end{bmatrix} = \vec{d}. \tag{5.1}$$

After r reductions, we have

$$T^{(r)}\vec{x}^{(r)} = \begin{bmatrix} A_r & B_r & & 0 \\ B_r & \ddots & \ddots & \\ & \ddots & \ddots & B_r \\ 0 & & B_r & A_r \end{bmatrix} \begin{bmatrix} \vec{x}_{2^r} \\ \vec{x}_{2(2^r)} \\ \vdots \\ \vec{x}_{m_r} \end{bmatrix} = \begin{bmatrix} \vec{d}_{2^r} \\ \vec{d}_{2(2^r)} \\ \vdots \\ \vec{d}_{m_r} \end{bmatrix} = \vec{d}^{(r)}, \tag{5.2}$$

where $m_r = m - 2^r + 1$. If we terminate the reduction phase when $r < k - 1$, we must solve (5.2) for $\vec{x}^{(r)}$ and begin our back-substitution. So, we need to determine the point at which we should terminate the reduction.

5.2 Revisiting the Back-Substitution

We will begin our discussion of the back-substitution by defining two vectors. First, let $\vec{x}^{(r)}$ be the exact solution to (5.2). Then $\vec{x}$, the exact solution to (5.1), can be found through back-substitution

based on $\vec{x}^{(r)}$. Also, define $\vec{y}^{(r)}$ to be an approximation of $\vec{x}^{(r)}$ found by solving the equation

$$A_r \vec{y}^{(r)} = \vec{d}^{(r)}. \tag{5.3}$$

So, if $\vec{y}$ is the solution to (5.1) found through back-substitution based on $\vec{y}^{(r)}$, then $\vec{y}$ is an approximation of $\vec{x}$. Two questions arise from this approximation:

1 Will the error of the initial approximation, measured by $\|\vec{x}^{(r)} - \vec{y}^{(r)}\|_\infty$, be compounded as we perform the back-substitution?

2 What is the maximum value of the relative error?

The answers were provided by Heller [9].

Theorem 5.1 *Let A^{-1} exist and $\|C_0\|_\infty < 1$. Then $\|\vec{x} - \vec{y}\|_\infty = \|\vec{x}^{(r)} - \vec{y}^{(r)}\|_\infty$.*

Proof by induction: Define z_r to be the vector

$$z_r = (0, \vec{x}_{2^r} - \vec{y}_{2^r}, 0, \vec{x}_{2(2^r)} - \vec{y}_{2(2^r)}, \ldots, 0, \vec{x}_{m_r} - \vec{y}_{m_r}, 0).$$

Notice that the non-zero components of $\vec{z}_r$ are the components of $(\vec{x}^{(r)} - \vec{y}^{(r)})$.

Let $n = 1$, and consider the vector $(\vec{x} - \vec{y})$. From Algorithm 4.3, we know that the odd-indexed component vectors are

$$\vec{x}_j = A^{-1}\vec{d}_j - A^{-1}B(\vec{x}_{j-1} + \vec{x}_{j+1}),$$

and

$$\vec{y}_j = A^{-1}\vec{d}_j - A^{-1}B(\vec{y}_{j-1} + \vec{y}_{j+1}),$$

for $j = 1, 3, \ldots, m$. The even-indexed component vectors do not change and are the components of $\vec{x}^{(1)}$ and $\vec{y}^{(1)}$.

By construction,

$$\vec{z}_1 = (0, \vec{x}_2 - \vec{y}_2, 0, \vec{x}_4 - \vec{y}_4, \ldots, 0, \vec{x}_{m-1} - \vec{y}_{m-1}, 0).$$

Thus,

$$\|\vec{z}_1\|_\infty = \|\vec{x}^{(1)} - \vec{y}^{(1)}\|_\infty.$$

We know that the odd-indexed components of $(\vec{x} - \vec{y})$ are

$$\vec{x}_j - \vec{y}_j = -A^{-1}B[(\vec{x}_{j-1} - \vec{y}_{j-1}) + (\vec{x}_{j+1} - \vec{y}_{j+1})],$$

and multiplication shows that these odd-indexed components are the non-zero components of $C_0\vec{z}_1$. The even-indexed components of $(\vec{x} - \vec{y})$ are the components of $(\vec{x}^{(1)} - \vec{y}^{(1)})$. So,

$$\|\vec{x} - \vec{y}\|_\infty = \max(\|\vec{x}^{(1)} - \vec{y}^{(1)}\|_\infty, \|C_0\vec{z}_1\|_\infty).$$

However,

$$\begin{aligned} \|C_0\vec{z}_1\|_\infty &\leq \|C_0\|_\infty \|\vec{z}_1\|_\infty \\ &< \|\vec{z}_1\|_\infty \\ &= \|\vec{x}^{(1)} - \vec{y}^{(1)}\|_\infty. \end{aligned}$$

Thus,

$$\|\vec{x} - \vec{y}\|_\infty = \|\vec{x}^{(1)} - \vec{y}^{(1)}\|_\infty.$$

So, the theorem holds when $n = 1$.

Now, suppose $\|\vec{x} - \vec{y}\|_\infty = \|\vec{x}^{(k)} - \vec{y}^{(k)}\|_\infty$. Then, for $\vec{x}^{(k)}$,

$$\vec{x}_j = A_k^{-1}\vec{d}_j^{(k)} - A_k^{-1}B_k(\vec{x}_{j-2^{k-1}} + \vec{x}_{j+2^{k-1}}),$$

and for $\vec{y}^{(k)}$,

$$\vec{y}_j = A_k^{-1}\vec{d}_j^{(k)} - A_k^{-1}B_k(\vec{y}_{j-2^{k-1}} + \vec{y}_{j+2^{k-1}}),$$

for $j = 2^k, 2(2^k), \ldots, m_k$. Also,

$$\vec{z}_{k+1} = (0, \vec{x}_{2^k-2^{k-1}} - \vec{y}_{2^k-2^{k-1}}, \ldots, 0, \vec{x}_{m_k} - \vec{y}_{m_k}).$$

By construction,

$$\|\vec{z}_{k+1}\|_\infty = \|\vec{x}^{(k+1)} - \vec{y}^{(k+1)}\|_\infty,$$

and

$$\vec{x}^{(k)} - \vec{y}^{(k)} = C_k\vec{z}_{k+1}.$$

So,

$$\begin{aligned}\|\vec{x} - \vec{y}\|_\infty &= \|\vec{x}^{(k)} - \vec{y}^{(k)}\|_\infty \\ &= \max(\|\vec{x}^{(k+1)} - \vec{y}^{(k+1)}\|_\infty, \|C_k\vec{z}_{k+1}\|_\infty).\end{aligned}$$

However,

$$\begin{aligned}\|C_k\vec{z}_{k+1}\|_\infty &\leq \|C_k\|_\infty\|\vec{z}_{k+1}\|_\infty \\ &< \|\vec{z}_{k+1}\|_\infty \\ &= \|\vec{x}^{(k+1)} - \vec{y}^{(k+1)}\|_\infty.\end{aligned}$$

Thus, $\|\vec{x} - \vec{y}\|_\infty = \|\vec{x}^{(k+1)} - \vec{y}^{(k+1)}\|_\infty$, and the theorem holds for all $n \in \mathbf{N}$. **QED.**

So, we know that the absolute error of the approximation will be $\|\vec{x}^{(r)} - \vec{y}^{(r)}\|_\infty$, where r is the last reduction step performed.

Theorem 5.2 *Suppose A^{-1} exists and $\|C_0\|_\infty < 1$. Then,*

$$\frac{\|\vec{x}^{(k)} - \vec{y}^{(k)}\|_\infty}{\|\vec{x}^{(k)}\|_\infty} \leq \|C_k\|_\infty.$$

Proof: By definition, $\vec{y}^{(k)} = A_k^{-1}\vec{d}^{(k)}$. For $\vec{x}^{(k)}$,

$$\vec{x}_j = A_k^{-1}\vec{d}_j^{(k)} - A_k^{-1}B_k(\vec{x}_{j-2^k} + \vec{x}_{j+2^k}).$$

So,

$$\begin{aligned}\vec{x}^{(k)} - \vec{y}^{(k)} &= -A_k^{-1}B_k(\vec{x}_{j-2^k} + \vec{x}_{j+2^k}) \\ &= C_k\vec{x}^{(k)}.\end{aligned}$$

This implies

$$\begin{aligned}\|\vec{x}^{(k)} - \vec{y}^{(k)}\|_\infty &= \|C_k\vec{x}^{(k)}\|_\infty \\ &\leq \|C_k\|_\infty\|\vec{x}^{(k)}\|_\infty,\end{aligned}$$

and

$$\frac{\|\vec{x}^{(k)} - \vec{y}^{(k)}\|_\infty}{\|\vec{x}^{(k)}\|_\infty} \leq \|C_k\|_\infty.$$

QED.

Thus, we know that the relative error of the approximation has an upper bound that is $\|C_r\|_\infty$. Now, we need to determine the step at which we should stop the reduction of the system. To do this, we will define k_a to be the point at which the reduction will be stopped based on the relative error of the approximation. Heller provided a termination point based on machine precision and $\|C_0\|_\infty$ [9].

Theorem 5.3 *Suppose A^{-1} exists and $\|C_0\|_\infty < 1$. Then, on a computer with precision ε, the relative error*

$$\frac{\|\vec{x}^{(r)} - \vec{y}^{(r)}\|_\infty}{\|\vec{x}^{(r)}\|_\infty} < \varepsilon$$

for all $i \geq k_a$, where

$$k_a = \left\lceil \frac{\log\left(\frac{\log \varepsilon}{\log \|C_0\|_\infty}\right)}{\log 2} \right\rceil.$$

Proof: From Theorem 4.1, we know $\|C_{k+1}\|_\infty \leq \|C_k\|_\infty^2$. So,

$$\|C_i\|_\infty \leq \|C_0\|_\infty^{2^i}.$$

Thus, to ensure $\|C_i\|_\infty < \varepsilon$, we have

$$\begin{aligned}
\|C_0\|_\infty^{2^i} &< \varepsilon \\
2^i \log(\|C_0\|_\infty) &< \log \varepsilon \\
2^i &> \frac{\log \varepsilon}{\log \|C_0\|_\infty} \\
i \log 2 &> \log\left(\frac{\log \varepsilon}{\log \|C_0\|_\infty}\right) \\
i &> \frac{\log\left(\frac{\log \varepsilon}{\log \|C_0\|_\infty}\right)}{\log 2}.
\end{aligned}$$

QED.

5.3 A Truncated Block Cyclic Reduction Algorithm

Using our development from the two previous sections, we can modify Algorithm 4.3. Changes will include calculating the earliest point at which we may terminate the reduction, terminating the reduction at this point, and beginning the back-substitution routine by calculating approximations for the variables remaining from the truncated reduction. All other parts of the algorithm remain the same.

THE REDUCTION PHASE
$\gamma = 2\|A^{-1}B\|_\infty$

$z = \log\left(\frac{\log \varepsilon}{\log \gamma}\right)$
$k_a = \min(k-1, \lceil z/\log 2 \rceil)$
$s = 1$
$A_0 = A$
$B_0 = B$
for $r = 1, \ldots, k_a$
  $M_r = -B_{r-1}A_{r-1}^{-1}$
  $B_r = M_r B_{r-1}$
  $A_r = A_{r-1} + 2B_r$
  $t = s$
  $s = 2s$
  for $j = s, 2s, \ldots, m+1-s$
    $\vec{d}_{j,r} = \vec{d}_{j,r-1} + M_r(\vec{d}_{j-t,r-1} + \vec{d}_{j+t,r-1})$
  end
end

THE BACK-SUBSTITUTION PHASE
for $k = s, 2s, \ldots, m+1-s$
  $\vec{x}_s = A_{k_a}^{-1}\vec{d}_{s,k_a}$
end
for $i = k_a - 1, k_a - 2, \ldots, 0$
  $t = s$
  $s = s/2$
  for $j = s, s+t, \ldots, m+1-s$
    $\vec{x}_j = A_i^{-1}[\vec{d}_{k,i} - B_i(\vec{x}_{j-s} + \vec{x}_{j+s})]$
  end
end

5.3.1 Example

As we did in the last chapter, let's demonstrate the algorithm on a system with $m = 1023$ and $n = 3$. Notice that this makes $k = 10$ and the overall dimension of T is 3069×3069. For this example, let

$$A = \begin{bmatrix} -4 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & -4 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\text{and } \vec{d}_j = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix},$$

for $j = 1, 2, \ldots, 1023$. Also, because we are working in ten-digit arithmetic with Maple 9.5, we set $\varepsilon = 1 \times 10^{-10}$.

Initially, we set

$$\begin{aligned} A_0 &= A, \\ B_0 &= B, \\ \text{and } s &= 1. \end{aligned}$$

Calculating, we find $\gamma_0 = 0.8571428570$, making $z = 2.174270244$ and $k_a = 8$.

Continuing with the algorithm, when $r = 1$, the algorithm produces the updates

$$M_1 = \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} -3.464285714 & 1.142857143 & 0.03571428572 \\ 1.142857143 & -3.423571429 & 1.142857143 \\ 0.03571428572 & 1.142857143 & -3.464285714 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,1} = \begin{bmatrix} 1.714285714 & 1.857142857 & 1.714285714 \end{bmatrix}^T,$$

for $j = 2, 4, \ldots, 1022$.

For $r = 2$, we find

$$M_2 = \begin{bmatrix} 0.09821428575 & 0.06250000004 & 0.02678571431 \\ 0.06250000003 & 0.1250000000 & 0.06250000003 \\ 0.02678571430 & 0.06250000004 & 0.09821428574 \end{bmatrix},$$

$$B_2 = \begin{bmatrix} 0.03125000002 & 0.02678571431 & 0.01339285716 \\ 0.02678571430 & 0.04464285715 & 0.02678571430 \\ 0.01339285715 & 0.02678571430 & 0.03125000001 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000004 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,2} = \begin{bmatrix} 2.375000000 & 2.750000000 & 2.375000000 \end{bmatrix}^T,$$

for $j = 4, 8, \ldots, 1020$.

Continuing when $r = 3$, our updates become

$$M_3 = \begin{bmatrix} 0.01543754045 & 0.01723673580 & 0.01028290127 \\ 0.01723673580 & 0.02572044169 & 0.01723673580 \\ 0.01028290126 & 0.01723673580 & 0.01543754044 \end{bmatrix},$$

$$B_3 = \begin{bmatrix} 0.001081838848 & 0.001458437538 & 0.0009897917195 \\ 0.001458437537 & 0.002071630566 & 0.001458437537 \\ 0.0009897917191 & 0.001458437538 & 0.001081838847 \end{bmatrix},$$

$$A_3 = \begin{bmatrix} -3.399622036 & 1.199345447 & 0.06447958348 \\ 1.199345447 & -3.335142454 & 1.199345447 \\ 0.06447958348 & 1.199345447 & -3.399622036 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,3} = \begin{bmatrix} 2.591974145 & 3.055211419 & 2.591974145 \end{bmatrix}^T,$$

for $j = 8, 16, \ldots, 1016$.

When $r = 4$, we have

$$M_4 = \begin{bmatrix} 0.0006443266570 & 0.0008911491475 & 0.0006177549398 \\ 0.0008911491471 & 0.001262081595 & 0.0008911491471 \\ 0.0006177549397 & 0.0008911491472 & 0.0006443266566 \end{bmatrix},$$

$$B_4 = \begin{bmatrix} 2.608191700 & 3.686798990 & 2.605745850 \\ 3.686798986 & 5.213937545 & 3.686798986 \\ 2.605745850 & 3.686798988 & 2.608191698 \end{bmatrix} \times 10^{-6},$$

$$A_4 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479497 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479495 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,4} = \begin{bmatrix} 2.603962009 & 3.072162613 & 2.603962009 \end{bmatrix}^T,$$

for $j = 16, 32, \ldots, 1008$.

Next, when $r = 5$, the updates are

$$M_5 = \begin{bmatrix} 1.590945050 & 2.249434502 & 1.590238994 \\ 2.249434500 & 3.181184040 & 2.249434500 \\ 1.590238992 & 2.249434500 & 1.590945049 \end{bmatrix} \times 10^{-6},$$

$$B_5 = \begin{bmatrix} 1.658646118 & 2.345679711 & 1.658645944 \\ 2.345679709 & 3.317292058 & 2.345679709 \\ 1.658645943 & 2.345679710 & 1.658646116 \end{bmatrix} \times 10^{-11},$$

$$A_5 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,5} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 32, 64, \ldots, 992$.

For the sixth reduction, $r = 6$, and

$$M_6 = \begin{bmatrix} 1.011992178 & 1.431173027 & 1.011992128 \\ 1.431173026 & 2.023984303 & 1.431173026 \\ 1.011992127 & 1.431173027 & 1.011992177 \end{bmatrix} \times 10^{-11},$$

$$B_6 = \begin{bmatrix} 6.714147064 & 9.495237837 & 6.714147064 \\ 9.495237828 & 13.42829411 & 9.495237827 \\ 6.714147062 & 9.495237832 & 6.714147061 \end{bmatrix} \times 10^{-22},$$

$$A_6 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,6} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 64, 128, \ldots, 960$.

When $r = 7$,

$$M_7 = \begin{bmatrix} 4.096512468 & 5.793343488 & 4.096512468 \\ 5.793343483 & 8.193024921 & 5.793343483 \\ 4.096512466 & 5.793343484 & 4.096512466 \end{bmatrix} \times 10^{-22},$$

$$B_7 = \begin{bmatrix} 1.100183486 & 1.555894406 & 1.100183485 \\ 1.555894404 & 2.200366967 & 1.555894404 \\ 1.100183485 & 1.555894406 & 1.100183485 \end{bmatrix} \times 10^{-42},$$

$$A_7 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,7} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 128, 256, \ldots, 896$.

Finally, when $r = 8$,

$$A_8 = \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix},$$

$$\text{and } \vec{d}_{j,8} = \begin{bmatrix} 2.603992398 & 3.072205589 & 2.603992398 \end{bmatrix}^T,$$

for $j = 256, 512, 768$.

So, our original system has been reduced to the equation

$$\begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix} \vec{x}_{512} = \begin{bmatrix} 2.603992398 \\ 3.072205589 \\ 2.603992398 \end{bmatrix}. \tag{5.4}$$

To begin the back-substitution phase of the algorithm, we solve (5.4) for $\vec{x}_{256}$, $\vec{x}_{512}$, and $\vec{x}_{768}$ using the approximation

$$\vec{x}_j = A_8^{-1} \vec{d}_{j,8}$$

, where $j = 256, 512, 768$. These values then become the starting point for the back-substitution routine.

In the first back-substitution, we find values for $\vec{x}_{256}$ and $\vec{x}_{768}$. The three known values are then used in a second round of back-substitution to find $\vec{x}_{128}$, $\vec{x}_{384}$, $\vec{x}_{640}$, and $\vec{x}_{896}$. This process continues six more times to recover all of the vectors. The final solution is shown in the Maple 9.5 output in Appendix H.

Once the back-substitution was done, the routine checked the solution. The error, which was measured by taking the infinity norm of $T\vec{x} - \vec{d}$, was found to be 6×10^{-9}.

5.4 Summary

In this chapter, we developed a truncated algorithm for the cyclic reduction of block tridiagonal matrices. The time savings will only be significant only when $\|A\|_\infty$ is large. This will allow the routine to complete the reduction in only a few steps.

5.5 Ideas for Future Work

In this thesis, we developed the complete cyclic reduction algorithms for tridiagonal and block tridiagonal matrices, and we showed that these algorithms are stable. We also developed the Bondeli-Gander algorithm for tridiagonal systems and extended it to cases where $b \neq 1$. Finally, we developed a truncated algorithm for block tridiagonal matrices based on the work of Heller.

Future work based on this thesis would include extending the Bondeli-Gander algorithm to block tridiagonal matrices of the form (I, A, I). Ultimately, this would be further extended to systems of the form (B, A, B), where $B \neq I$.

The work for these extensions will rely on the facts that $\{A_i\} \to \sqrt{A^2 - 4I}$ when $T = (I, A, I)$ and $\{A_i\} \to \sqrt{A^2 - 4B^2}$ when $T = (B, A, B)$. Proofs of these statements will require the definition of an order relation on matrices to show monotonicity of the sequences. Also, the radicands must be shown to be positive definite. The convergence theorem then appears similar to Theorem 2.1. Continuing with lemmas similar to Lemmas 3.1 through 3.3 should establish a better termination point for the reduction than is currently found using Heller's criterion.

Appendix A

A Brief Discussion of Hyperbolic Functions

In mathematics, certain types of functions arise frequently enough to warrant their own classifications. Among these are the familiar quadratic, exponential, and circular (trigonometric) functions. The hyperbolic functions, so called because their relationships to the hyperbola $x^2 - y^2 = a^2$ are similar to the relationships of the circular functions to the circle $x^2 + y^2 = a^2$, are another such group. The function names are *hyperbolic sine* (sinh), *hyperbolic cosine* (cosh), and so on.

Two approaches may be taken in a discussion of the hyperbolic functions—geometric and analytic. The geometric approach may be found in the CRC Standard Mathematical Tables. In this appendix, we will discuss hyperbolic functions using an analytic approach.

To begin our discussion, we define the hyperbolic functions using the exponential function.

$$\begin{aligned} \sinh u &= \frac{e^u - e^{-u}}{2} \\ \cosh u &= \frac{e^u + e^{-u}}{2} \\ \tanh u &= \frac{\sinh u}{\cosh u} = \frac{e^u - e^{-u}}{e^u + e^{-u}} \end{aligned}$$

Similar to the circular functions, there is a basic identity for the hyperbolic functions.

Identity A.1 $\cosh^2 u - \sinh^2 u = 1$.

Proof: Starting with the definitions, we see

$$\begin{aligned} \cosh^2 u - \sinh^2 u &= \left(\frac{e^u + e^{-u}}{2}\right)^2 - \left(\frac{e^u - e^{-u}}{2}\right)^2 \\ &= \frac{e^{2u} + 2 + e^{-2u}}{4} - \frac{e^{2u} - 2 + e^{-2u}}{4} \\ &= \frac{2+2}{4} \\ &= 1. \end{aligned}$$

QED.

The reciprocals of the hyperbolic functions are defined in a manner analogous to the reciprocals of the circular functions. However, we will be concerned with only one of these,

$$\coth u = \frac{1}{\tanh u}.$$

The remainder of this appendix will focus on developing the formulae used in this thesis—namely, $(\coth 2u)$, a variant of $(\coth u)$, and $(\operatorname{arctanh} x)$. We will start with a determination of the angle-sum formulae.

Identity A.2 $\sinh(u+v) = (\sinh u)(\cosh v) + (\cosh u)(\sinh v)$

Proof:

$$\begin{aligned}
\sinh(u+v) &= \frac{e^{u+v} - e^{-u-v}}{2} \\
&= \frac{e^{u+v} - e^{-u-v}}{4} + \frac{e^{u+v} - e^{-u-v}}{4} \\
&= \frac{e^{u+v} + e^{u-v} - e^{-u+v} - e^{-u-v}}{4} + \frac{e^{u+v} - e^{u-v} + e^{-u+v} - e^{-u-v}}{4} \\
&= \frac{(e^{u} - e^{-u})(e^{v} + e^{-v})}{4} + \frac{(e^{u} + e^{-u})(e^{v} - e^{-v})}{4} \\
&= (\sinh u)(\cosh v) + (\cosh u)(\sinh v).
\end{aligned}$$

QED.

Identity A.3 $\cosh(u+v) = (\cosh u)(\cosh v) + (\sinh u)(\sinh v).$

Proof:

$$\begin{aligned}
\cosh(u+v) &= \frac{e^{u+v} + e^{-u-v}}{2} \\
&= \frac{e^{u+v} + e^{-u-v}}{4} + \frac{e^{u+v} + e^{-u-v}}{4} \\
&= \frac{e^{u+v} + e^{u-v} + e^{-u+v} + e^{-u-v}}{4} + \frac{e^{u+v} - e^{u-v} - e^{-u+v} + e^{-u-v}}{4} \\
&= \frac{(e^{u} + e^{-u})(e^{v} + e-v)}{4} + \frac{(e^{u} - e^{-u})(e^{v} - e^{-v})}{4} \\
&= (\cosh u)(\cosh v) + (\sinh u)(\sinh v).
\end{aligned}$$

QED.

Identity A.4 $\coth(u+v) = \frac{(\coth u)(\coth v)+1}{\coth u + \coth v}$

Proof:

$$\begin{aligned}
\coth(u+v) &= \frac{\cosh(u+v)}{\sinh(u+v)} \\
&= \frac{(\cosh u)(\cosh v) + (\sinh u)(\sinh v)}{(\sinh u)(\cosh v) + (\cosh u)(\sinh v)}
\end{aligned}$$

Because $\cosh u = (\coth u)(\sinh u)$,

$$\begin{aligned}
\coth(u+v) &= \frac{(\coth u)(\sinh u)(\coth v)(\sinh v) + (\sinh u)(\sinh v)}{(\sinh u)(\coth v)(\sinh v) + (\coth u)(\sinh u)(\sinh v)} \\
&= \frac{(\sinh u)(\sinh v)[(\coth u)(\coth v) + 1]}{(\sinh u)(\sinh v)(\coth u + \coth v)} \\
&= \frac{(\coth u)(\coth v) + 1}{\coth u + \coth v}
\end{aligned}$$

QED.

From here, we can find the double-angle formula for $(\coth 2u)$.

Identity A.5 $\coth(2u) = \frac{\coth^2 u + 1}{2 \coth u}$

Proof:

$$\begin{aligned}
\coth(2u) &= \coth(u+u) \\
&= \frac{(\coth u)(\coth u) + 1}{\coth u + \coth u} \\
&= \frac{\coth^2 u + 1}{2 \coth u}
\end{aligned}$$

QED.

We also use a variant of $(\coth u)$ in our proofs.

Identity A.6 $\coth u = \frac{e^{2u}+1}{e^{2u}-1}$

Proof:

$$\begin{aligned}
\coth u &= \frac{e^u + e^{-u}}{e^u - e^{-u}} \\
&= \frac{e^u(e^u + e^{-u})}{e^u(e^u - e^{-u})} \\
&= \frac{e^{2u} + 1}{e^{2u} - 1}
\end{aligned}$$

QED.

Finally, we need a formula for $(\operatorname{arctanh} x)$.

Identity A.7 $\operatorname{arctanh} x = \frac{1}{2} \ln\left(\frac{1+x}{1-x}\right)$ *when* $|x| < 1$

Proof:

$$\begin{aligned}
\operatorname{arctanh} x &= u \\
x &= \tanh u \\
&= \frac{e^u - e^{-u}}{e^u + e^{-u}}
\end{aligned}$$

So,

$$\begin{aligned} x(e^u + e^{-u}) &= e^u - e^{-u} \\ xe^u + xe^{-u} &= e^u - e^{-u} \\ e^{-u} + xe^{-u} &= e^u - xe^u \\ e^{-u}(1+x) &= e^u(1-x) \\ \frac{1+x}{1-x} &= e^{2u} \end{aligned}$$

Notice that $|x|$ must be less than 1 for this to be valid. If $|x| > 1$, then the left-hand side is negative, violating the property that the exponential function is always positive. If $|x| = 1$, then the left-hand side is either undefined (0 in the denominator) or 0. However, the exponential function is always positive. Thus, $|x|$ must always be less than 1.

$$\begin{aligned} \ln\left(\frac{1+x}{1-x}\right) &= 2u \\ u &= \frac{1}{2}\ln\left(\frac{1+x}{1-x}\right) \\ \operatorname{arctanh} x &= \frac{1}{2}\ln\left(\frac{1+x}{1-x}\right) \end{aligned}$$

QED.

Appendix B

Example 2.1.1

Example 2.1.1

```
> restart: with(linalg): Digits:=10:
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

This example will demonstrate the individual steps of the reduction phase of the cyclic reduction algorithm.

```
> m:=7: a:=-4: b:=1: d:=vector(7,[-3,-2,-2,-2,-2,-2,-3]):
> k:=ilog2(m+1);
```

$$k := 3$$

The First Reduction

```
> a1:=evalf(a-2/a); b1:=evalf(-1/a);
```

$$a1 := -3.500000000$$

$$b1 := 0.2500000000$$

```
> for j from 2 to 6 by 2 do
> d1[j]:=evalf(d[j]-(1/a)*(d[j-1]+d[j+1]));
> od;
```

$$d1_2 := -3.250000000$$

$$d1_4 := -3.$$

$$d1_6 := -3.250000000$$

The Second Reduction

```
> a2:=evalf(-3.5-2*0.25^2/(-3.5)); b2:=evalf(-0.25^2/(-3.5));
> d2[4]:=evalf(-3-(0.25/(-3.5))*(-3.25-3.25));
```

$$a2 := -3.464285714$$

$$b2 := 0.01785714286$$

$$d2_4 := -3.464285714$$

Appendix C

Example 2.3.1

Example 2.3.1

This program is designed to solve systems of linear equations of the form Tx=d. We are assuming that T is symmeteric, tridiagonal, and diagonally dominant. All diagonal elements of T are equal to a, and all sub- and super-diagonal entries are equal to b. Also, m (the dimension of T) is assumed to be 2^k-1.

```
> restart: with(linalg):
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

The Set-Up Phase

In this phase of the routine, we are defining and determining the parameters of the system. Because we will be modifying the values of a, b, and d[i] at each step of the reduction phase, we will use vectors to store the values of a and b and a matrix to store the values of d[i].

```
> m:=127: k:=ilog2(m+1):
> a:=vector(k,0): b:=vector(k,0): d:=matrix(m,k,0):
> X:=vector(m+2,0): x:=vector(m,0):
> a[1]:=-4: b[1]:=1:
> for i from 1 to m do
> d[i,1]:=1:
> od:
```

The Reduction Phase

In this phase of the routine, we will be decoupling the variables from each other. This will allow us to solve a single equation for a single unknown for eventual back-substitution.

```
> printlevel:=2: s:=1;
```

$$s := 1$$

```
> for r from 2 to k do
> b[r]:=evalf(-b[r-1]^2/a[r-1]);
> a[r]:=evalf(a[r-1]+2*b[r]);
> t:=s;
> s:=2*s;
> for j from s to m+1-s by s do
> d[j,r]:=evalf(d[j,r-1]-b[r-1]*(d[j-t,r-1]+d[j+t,r-1])/a[r-1]);
> od;
> od;
```

$$b_2 := 0.2500000000$$
$$a_2 := -3.500000000$$
$$t := 1$$
$$s := 2$$
$$d_{2,2} := 1.500000000$$
$$d_{4,2} := 1.500000000$$
$$d_{6,2} := 1.500000000$$

$$d_{8,2} := 1.500000000$$
$$d_{10,2} := 1.500000000$$
$$d_{12,2} := 1.500000000$$
$$d_{14,2} := 1.500000000$$
$$d_{16,2} := 1.500000000$$
$$d_{18,2} := 1.500000000$$
$$d_{20,2} := 1.500000000$$
$$d_{22,2} := 1.500000000$$
$$d_{24,2} := 1.500000000$$
$$d_{26,2} := 1.500000000$$
$$d_{28,2} := 1.500000000$$
$$d_{30,2} := 1.500000000$$
$$d_{32,2} := 1.500000000$$
$$d_{34,2} := 1.500000000$$
$$d_{36,2} := 1.500000000$$
$$d_{38,2} := 1.500000000$$
$$d_{40,2} := 1.500000000$$
$$d_{42,2} := 1.500000000$$
$$d_{44,2} := 1.500000000$$
$$d_{46,2} := 1.500000000$$
$$d_{48,2} := 1.500000000$$
$$d_{50,2} := 1.500000000$$
$$d_{52,2} := 1.500000000$$
$$d_{54,2} := 1.500000000$$
$$d_{56,2} := 1.500000000$$
$$d_{58,2} := 1.500000000$$
$$d_{60,2} := 1.500000000$$
$$d_{62,2} := 1.500000000$$
$$d_{64,2} := 1.500000000$$
$$d_{66,2} := 1.500000000$$
$$d_{68,2} := 1.500000000$$
$$d_{70,2} := 1.500000000$$
$$d_{72,2} := 1.500000000$$
$$d_{74,2} := 1.500000000$$
$$d_{76,2} := 1.500000000$$
$$d_{78,2} := 1.500000000$$
$$d_{80,2} := 1.500000000$$
$$d_{82,2} := 1.500000000$$

$$d_{84,2} := 1.500000000$$
$$d_{86,2} := 1.500000000$$
$$d_{88,2} := 1.500000000$$
$$d_{90,2} := 1.500000000$$
$$d_{92,2} := 1.500000000$$
$$d_{94,2} := 1.500000000$$
$$d_{96,2} := 1.500000000$$
$$d_{98,2} := 1.500000000$$
$$d_{100,2} := 1.500000000$$
$$d_{102,2} := 1.500000000$$
$$d_{104,2} := 1.500000000$$
$$d_{106,2} := 1.500000000$$
$$d_{108,2} := 1.500000000$$
$$d_{110,2} := 1.500000000$$
$$d_{112,2} := 1.500000000$$
$$d_{114,2} := 1.500000000$$
$$d_{116,2} := 1.500000000$$
$$d_{118,2} := 1.500000000$$
$$d_{120,2} := 1.500000000$$
$$d_{122,2} := 1.500000000$$
$$d_{124,2} := 1.500000000$$
$$d_{126,2} := 1.500000000$$
$$b_3 := 0.01785714286$$
$$a_3 := -3.464285714$$
$$t := 2$$
$$s := 4$$
$$d_{4,3} := 1.714285714$$
$$d_{8,3} := 1.714285714$$
$$d_{12,3} := 1.714285714$$
$$d_{16,3} := 1.714285714$$
$$d_{20,3} := 1.714285714$$
$$d_{24,3} := 1.714285714$$
$$d_{28,3} := 1.714285714$$
$$d_{32,3} := 1.714285714$$
$$d_{36,3} := 1.714285714$$
$$d_{40,3} := 1.714285714$$
$$d_{44,3} := 1.714285714$$
$$d_{48,3} := 1.714285714$$
$$d_{52,3} := 1.714285714$$

$$d_{56,3} := 1.714285714$$
$$d_{60,3} := 1.714285714$$
$$d_{64,3} := 1.714285714$$
$$d_{68,3} := 1.714285714$$
$$d_{72,3} := 1.714285714$$
$$d_{76,3} := 1.714285714$$
$$d_{80,3} := 1.714285714$$
$$d_{84,3} := 1.714285714$$
$$d_{88,3} := 1.714285714$$
$$d_{92,3} := 1.714285714$$
$$d_{96,3} := 1.714285714$$
$$d_{100,3} := 1.714285714$$
$$d_{104,3} := 1.714285714$$
$$d_{108,3} := 1.714285714$$
$$d_{112,3} := 1.714285714$$
$$d_{116,3} := 1.714285714$$
$$d_{120,3} := 1.714285714$$
$$d_{124,3} := 1.714285714$$
$$b_4 := 0.00009204712816$$
$$a_4 := -3.464101620$$
$$t := 4$$
$$s := 8$$
$$d_{8,4} := 1.731958763$$
$$d_{16,4} := 1.731958763$$
$$d_{24,4} := 1.731958763$$
$$d_{32,4} := 1.731958763$$
$$d_{40,4} := 1.731958763$$
$$d_{48,4} := 1.731958763$$
$$d_{56,4} := 1.731958763$$
$$d_{64,4} := 1.731958763$$
$$d_{72,4} := 1.731958763$$
$$d_{80,4} := 1.731958763$$
$$d_{88,4} := 1.731958763$$
$$d_{96,4} := 1.731958763$$
$$d_{104,4} := 1.731958763$$
$$d_{112,4} := 1.731958763$$
$$d_{120,4} := 1.731958763$$
$$b_5 := 0.2445850247\,10^{-8}$$
$$a_5 := -3.464101615$$

$$t := 8$$
$$s := 16$$
$$d_{16,5} := 1.732050805$$
$$d_{32,5} := 1.732050805$$
$$d_{48,5} := 1.732050805$$
$$d_{64,5} := 1.732050805$$
$$d_{80,5} := 1.732050805$$
$$d_{96,5} := 1.732050805$$
$$d_{112,5} := 1.732050805$$
$$b_6 := 0.1726907607\,10^{-17}$$
$$a_6 := -3.464101615$$
$$t := 16$$
$$s := 32$$
$$d_{32,6} := 1.732050807$$
$$d_{64,6} := 1.732050807$$
$$d_{96,6} := 1.732050807$$
$$b_7 := 0.8608898394\,10^{-36}$$
$$a_7 := -3.464101615$$
$$t := 32$$
$$s := 64$$
$$d_{64,7} := 1.732050807$$

The Back-Substitution Phase

In this phase of the algorithm, we will use the dummy vector X to determine the solution. This will allow us to define x[0]=x[n+1]=0 (X[1] and X[m+2]) for use in the j-loop of the back-substitution routine. We will then eliminate these artificial entries to obtain the our true solution x.

```
> s:=(m+1)/2:
```
$$s := 64$$
```
> X[s+1]:=d[s,k]/a[k]:
```
$$X_{65} := -0.4999999999$$
```
> for r from k-1 to 1 by -1 do
> t:=s;
> s:=s/2;
> for j from s to m+2-s by t do
> X[j+1]:=(d[j,r]-b[r]*(X[j-s+1]+X[j+s+1]))/a[r];
> od;
> od:
```
$$t := 64$$
$$s := 32$$
$$X_{33} := -0.4999999999$$
$$X_{97} := -0.4999999999$$
$$t := 32$$
$$s := 16$$

$$X_{17} := -0.4999999996$$
$$X_{49} := -0.4999999999$$
$$X_{81} := -0.4999999999$$
$$X_{113} := -0.4999999996$$
$$t := 16$$
$$s := 8$$
$$X_{9} := -0.4999867143$$
$$X_{25} := -0.5000000000$$
$$X_{41} := -0.5000000000$$
$$X_{57} := -0.5000000000$$
$$X_{73} := -0.5000000000$$
$$X_{89} := -0.5000000000$$
$$X_{105} := -0.5000000000$$
$$X_{121} := -0.4999867143$$
$$t := 8$$
$$s := 4$$
$$X_{5} := -0.4974226118$$
$$X_{13} := -0.4999999316$$
$$X_{21} := -0.5000000000$$
$$X_{29} := -0.5000000000$$
$$X_{37} := -0.5000000000$$
$$X_{45} := -0.5000000000$$
$$X_{53} := -0.5000000000$$
$$X_{61} := -0.5000000000$$
$$X_{69} := -0.5000000000$$
$$X_{77} := -0.5000000000$$
$$X_{85} := -0.5000000000$$
$$X_{93} := -0.5000000000$$
$$X_{101} := -0.5000000000$$
$$X_{109} := -0.5000000000$$
$$X_{117} := -0.4999999316$$
$$X_{125} := -0.4974226118$$
$$t := 4$$
$$s := 2$$
$$X_{3} := -0.4641016151$$
$$X_{7} := -0.4998149520$$
$$X_{11} := -0.4999990463$$
$$X_{15} := -0.4999999951$$
$$X_{19} := -0.5000000000$$
$$X_{23} := -0.5000000000$$
$$X_{27} := -0.5000000000$$
$$X_{31} := -0.5000000000$$
$$X_{35} := -0.5000000000$$

$$X_{39} := -0.5000000000$$
$$X_{43} := -0.5000000000$$
$$X_{47} := -0.5000000000$$
$$X_{51} := -0.5000000000$$
$$X_{55} := -0.5000000000$$
$$X_{59} := -0.5000000000$$
$$X_{63} := -0.5000000000$$
$$X_{67} := -0.5000000000$$
$$X_{71} := -0.5000000000$$
$$X_{75} := -0.5000000000$$
$$X_{79} := -0.5000000000$$
$$X_{83} := -0.5000000000$$
$$X_{87} := -0.5000000000$$
$$X_{91} := -0.5000000000$$
$$X_{95} := -0.5000000000$$
$$X_{99} := -0.5000000000$$
$$X_{103} := -0.5000000000$$
$$X_{107} := -0.5000000000$$
$$X_{111} := -0.5000000000$$
$$X_{115} := -0.4999999951$$
$$X_{119} := -0.4999990463$$
$$X_{123} := -0.4998149520$$
$$X_{127} := -0.4641016151$$
$$t := 2$$
$$s := 1$$
$$X_{2} := -0.3660254038$$
$$X_{4} := -0.4903810568$$
$$X_{6} := -0.4993093910$$
$$X_{8} := -0.4999504165$$
$$X_{10} := -0.4999964402$$
$$X_{12} := -0.4999997445$$
$$X_{14} := -0.4999999818$$
$$X_{16} := -0.4999999988$$
$$X_{18} := -0.5000000000$$
$$X_{20} := -0.5000000000$$
$$X_{22} := -0.5000000000$$
$$X_{24} := -0.5000000000$$
$$X_{26} := -0.5000000000$$
$$X_{28} := -0.5000000000$$
$$X_{30} := -0.5000000000$$
$$X_{32} := -0.5000000000$$
$$X_{34} := -0.5000000000$$
$$X_{36} := -0.5000000000$$

$$X_{38} := -0.5000000000$$
$$X_{40} := -0.5000000000$$
$$X_{42} := -0.5000000000$$
$$X_{44} := -0.5000000000$$
$$X_{46} := -0.5000000000$$
$$X_{48} := -0.5000000000$$
$$X_{50} := -0.5000000000$$
$$X_{52} := -0.5000000000$$
$$X_{54} := -0.5000000000$$
$$X_{56} := -0.5000000000$$
$$X_{58} := -0.5000000000$$
$$X_{60} := -0.5000000000$$
$$X_{62} := -0.5000000000$$
$$X_{64} := -0.5000000000$$
$$X_{66} := -0.5000000000$$
$$X_{68} := -0.5000000000$$
$$X_{70} := -0.5000000000$$
$$X_{72} := -0.5000000000$$
$$X_{74} := -0.5000000000$$
$$X_{76} := -0.5000000000$$
$$X_{78} := -0.5000000000$$
$$X_{80} := -0.5000000000$$
$$X_{82} := -0.5000000000$$
$$X_{84} := -0.5000000000$$
$$X_{86} := -0.5000000000$$
$$X_{88} := -0.5000000000$$
$$X_{90} := -0.5000000000$$
$$X_{92} := -0.5000000000$$
$$X_{94} := -0.5000000000$$
$$X_{96} := -0.5000000000$$
$$X_{98} := -0.5000000000$$
$$X_{100} := -0.5000000000$$
$$X_{102} := -0.5000000000$$
$$X_{104} := -0.5000000000$$
$$X_{106} := -0.5000000000$$
$$X_{108} := -0.5000000000$$
$$X_{110} := -0.5000000000$$
$$X_{112} := -0.5000000000$$
$$X_{114} := -0.4999999988$$
$$X_{116} := -0.4999999818$$
$$X_{118} := -0.4999997445$$
$$X_{120} := -0.4999964402$$

$$X_{122} := -0.4999504165$$
$$X_{124} := -0.4993093910$$
$$X_{126} := -0.4903810568$$
$$X_{128} := -0.3660254038$$

```
> for i from 1 to m do
> x[i]:=X[i+1]:
> od:
> print(x);
```

$$[-0.3660254038, -0.4641016151, -0.4903810568, -0.4974226118, -0.4993093910,$$
$$-0.4998149520, -0.4999504165, -0.4999867143, -0.4999964402, -0.4999990463,$$
$$-0.4999997445, -0.4999999316, -0.4999999818, -0.4999999951, -0.4999999988,$$
$$-0.4999999996, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.4999999999, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.4999999999, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.4999999999, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.4999999999,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.4999999999, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.4999999996, -0.4999999988, -0.4999999951, -0.4999999818,$$
$$-0.4999999316, -0.4999997445, -0.4999990463, -0.4999964402, -0.4999867143,$$
$$-0.4999504165, -0.4998149520, -0.4993093910, -0.4974226118, -0.4903810568,$$
$$-0.4641016151, -0.3660254038]$$

Check

In this phase, we will determine the error Tx-d as a check on the routine.

```
> T:=matrix(m,m,0):
> for i from 1 to m do
> T[i,i]:=a[1]:
> od:
> for i from 2 to m do
> T[i,i-1]:=b[1];
> T[i-1,i]:=b[1];
> od:
```

```
> dcheck:=evalm(T&*x):
> derror:=norm(col(d,1)-dcheck);
```

$$derror := 0.10\,10^{-8}$$

Appendix D

Example 3.3.1

Example 3.3.1

This program is designed to solve systems of linear equations of the form Tx=d using the Bondeli-Gander algorithm. We are assuming that T is symmetric, tridiagonal, and diagonally dominant. All diagonal elements of T are equal to a, and all sub- and super-diagonal entries are equal to b. Also, m (the dimension of T) is assumed to be 2^k-1.

```
> restart: with(linalg):
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

The Set-Up Phase

In this phase of the routine, we are defining and determining the parameters of the system. Because we will be modifying the values of a, b, and d[i] at each step of the reduction, we will use vectors to store the values of a and b and a matrix to store the values of d[i].

```
> m:=127: k:=ilog2(m+1): Digits:=10: epsilon:=evalf(1*10^(-Digits));
```

$$\varepsilon := 0.1000000000\,10^{-9}$$

```
> a:=vector(k,0): b:=vector(k,0): d:=matrix(m,k,0):
> X:=vector(m+2,0): x:=vector(m,0):
> a[1]:=-4: b[1]:=1:
> for i from 1 to m do
> d[i,1]:=1:
> od:
```

The Reduction Phase

In this phase of the routine, we will be decoupling the variables from each other. This will allow us to solve a single equation for a single unknown for eventual back-substitution.

```
> printlevel:=2:
> z:=evalf(log10(log10(2/epsilon))-log10(log10(abs(a[1])+sqrt(a[1]^2-4*
> b[1]^2)-log10(2*abs(b[1])))));
```

$$z := 1.080864104$$

```
> ka:=min(k-1,ceil(z/log10(2)));
```

$$ka := 4$$

```
> s:=1:
> for r from 2 to ka+1 do
> b[r]:=evalf(-b[r-1]^2/a[r-1]);
> a[r]:=evalf(a[r-1]+2*b[r]);
> t:=s;
> s:=2*s;
> for j from s to m+1-s by s do
> d[j,r]:=evalf(d[j,r-1]-b[r-1]*(d[j-t,r-1]+d[j+t,r-1])/a[r-1]);
> od;
> od;
```

$$b_2 := 0.2500000000$$
$$a_2 := -3.500000000$$

$$t := 1$$
$$s := 2$$
$$d_{2,2} := 1.500000000$$
$$d_{4,2} := 1.500000000$$
$$d_{6,2} := 1.500000000$$
$$d_{8,2} := 1.500000000$$
$$d_{10,2} := 1.500000000$$
$$d_{12,2} := 1.500000000$$
$$d_{14,2} := 1.500000000$$
$$d_{16,2} := 1.500000000$$
$$d_{18,2} := 1.500000000$$
$$d_{20,2} := 1.500000000$$
$$d_{22,2} := 1.500000000$$
$$d_{24,2} := 1.500000000$$
$$d_{26,2} := 1.500000000$$
$$d_{28,2} := 1.500000000$$
$$d_{30,2} := 1.500000000$$
$$d_{32,2} := 1.500000000$$
$$d_{34,2} := 1.500000000$$
$$d_{36,2} := 1.500000000$$
$$d_{38,2} := 1.500000000$$
$$d_{40,2} := 1.500000000$$
$$d_{42,2} := 1.500000000$$
$$d_{44,2} := 1.500000000$$
$$d_{46,2} := 1.500000000$$
$$d_{48,2} := 1.500000000$$
$$d_{50,2} := 1.500000000$$
$$d_{52,2} := 1.500000000$$
$$d_{54,2} := 1.500000000$$
$$d_{56,2} := 1.500000000$$
$$d_{58,2} := 1.500000000$$
$$d_{60,2} := 1.500000000$$
$$d_{62,2} := 1.500000000$$
$$d_{64,2} := 1.500000000$$
$$d_{66,2} := 1.500000000$$
$$d_{68,2} := 1.500000000$$
$$d_{70,2} := 1.500000000$$
$$d_{72,2} := 1.500000000$$
$$d_{74,2} := 1.500000000$$

$$d_{76,2} := 1.500000000$$
$$d_{78,2} := 1.500000000$$
$$d_{80,2} := 1.500000000$$
$$d_{82,2} := 1.500000000$$
$$d_{84,2} := 1.500000000$$
$$d_{86,2} := 1.500000000$$
$$d_{88,2} := 1.500000000$$
$$d_{90,2} := 1.500000000$$
$$d_{92,2} := 1.500000000$$
$$d_{94,2} := 1.500000000$$
$$d_{96,2} := 1.500000000$$
$$d_{98,2} := 1.500000000$$
$$d_{100,2} := 1.500000000$$
$$d_{102,2} := 1.500000000$$
$$d_{104,2} := 1.500000000$$
$$d_{106,2} := 1.500000000$$
$$d_{108,2} := 1.500000000$$
$$d_{110,2} := 1.500000000$$
$$d_{112,2} := 1.500000000$$
$$d_{114,2} := 1.500000000$$
$$d_{116,2} := 1.500000000$$
$$d_{118,2} := 1.500000000$$
$$d_{120,2} := 1.500000000$$
$$d_{122,2} := 1.500000000$$
$$d_{124,2} := 1.500000000$$
$$d_{126,2} := 1.500000000$$
$$b_3 := 0.01785714286$$
$$a_3 := -3.464285714$$
$$t := 2$$
$$s := 4$$
$$d_{4,3} := 1.714285714$$
$$d_{8,3} := 1.714285714$$
$$d_{12,3} := 1.714285714$$
$$d_{16,3} := 1.714285714$$
$$d_{20,3} := 1.714285714$$
$$d_{24,3} := 1.714285714$$
$$d_{28,3} := 1.714285714$$
$$d_{32,3} := 1.714285714$$
$$d_{36,3} := 1.714285714$$

$$d_{40,3} := 1.714285714$$
$$d_{44,3} := 1.714285714$$
$$d_{48,3} := 1.714285714$$
$$d_{52,3} := 1.714285714$$
$$d_{56,3} := 1.714285714$$
$$d_{60,3} := 1.714285714$$
$$d_{64,3} := 1.714285714$$
$$d_{68,3} := 1.714285714$$
$$d_{72,3} := 1.714285714$$
$$d_{76,3} := 1.714285714$$
$$d_{80,3} := 1.714285714$$
$$d_{84,3} := 1.714285714$$
$$d_{88,3} := 1.714285714$$
$$d_{92,3} := 1.714285714$$
$$d_{96,3} := 1.714285714$$
$$d_{100,3} := 1.714285714$$
$$d_{104,3} := 1.714285714$$
$$d_{108,3} := 1.714285714$$
$$d_{112,3} := 1.714285714$$
$$d_{116,3} := 1.714285714$$
$$d_{120,3} := 1.714285714$$
$$d_{124,3} := 1.714285714$$
$$b_4 := 0.00009204712816$$
$$a_4 := -3.464101620$$
$$t := 4$$
$$s := 8$$
$$d_{8,4} := 1.731958763$$
$$d_{16,4} := 1.731958763$$
$$d_{24,4} := 1.731958763$$
$$d_{32,4} := 1.731958763$$
$$d_{40,4} := 1.731958763$$
$$d_{48,4} := 1.731958763$$
$$d_{56,4} := 1.731958763$$
$$d_{64,4} := 1.731958763$$
$$d_{72,4} := 1.731958763$$
$$d_{80,4} := 1.731958763$$
$$d_{88,4} := 1.731958763$$
$$d_{96,4} := 1.731958763$$
$$d_{104,4} := 1.731958763$$

$$d_{112,4} := 1.731958763$$
$$d_{120,4} := 1.731958763$$
$$b_5 := 0.2445850247\,10^{-8}$$
$$a_5 := -3.464101615$$
$$t := 8$$
$$s := 16$$
$$d_{16,5} := 1.732050805$$
$$d_{32,5} := 1.732050805$$
$$d_{48,5} := 1.732050805$$
$$d_{64,5} := 1.732050805$$
$$d_{80,5} := 1.732050805$$
$$d_{96,5} := 1.732050805$$
$$d_{112,5} := 1.732050805$$

The Back-Substitution Phase

In this phase of the routine, we will use the dummy vector X to determine the solution. This will allow us to define x[0]=x[n+1]=0 (X[1] and X[m+2]) for use in the j-loop of the back-substitution routine. We will then eliminate these artificial entries to obtain our true solution x.

```
> for k from s to m+1-s by s do
> X[k+1]:=d[s,ka+1]/a[ka+1];
> od:
> for r from ka to 1 by -1 do
> t:=s;
> s:=s/2;
> for j from s to m+2-s by t do
> X[j+1]:=(d[j,r]-b[r]*(X[j-s+1]+X[j+s+1]))/a[r];
> od:
> od:
> for i from 1 to m do
> x[i]:=X[i+1]:
> od:
> print(x);
```

$$[-0.3660254038, -0.4641016151, -0.4903810568, -0.4974226118, -0.4993093910,$$
$$-0.4998149520, -0.4999504165, -0.4999867143, -0.4999964402, -0.4999990463,$$
$$-0.4999997445, -0.4999999316, -0.4999999818, -0.4999999951, -0.4999999985,$$
$$-0.4999999993, -0.4999999998, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.4999999998, -0.4999999993, -0.4999999998, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.4999999998, -0.4999999993, -0.4999999998, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.4999999998, -0.4999999993, -0.4999999998,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.4999999998, -0.4999999993,$$
$$-0.4999999998, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.4999999998,$$
$$-0.4999999993, -0.4999999998, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000, -0.5000000000,$$
$$-0.4999999998, -0.4999999993, -0.4999999985, -0.4999999951, -0.4999999818,$$
$$-0.4999999316, -0.4999997445, -0.4999990463, -0.4999964402, -0.4999867143,$$
$$-0.4999504165, -0.4998149520, -0.4993093910, -0.4974226118, -0.4903810568,$$
$$-0.4641016151, -0.3660254038]$$

Check

In this phase of the routine, we find the norm of the error vector Tx-d to check the reliability of the algorithm.

```
> T:=matrix(m,m,0):
> for i from 1 to m do
> T[i,i]:=a[1]:
> od:
> for i from 2 to m do
> T[i,i-1]:=b[1];
> T[i-1,i]:=b[1];
> od:
> dcheck:=evalm(T&*x):  derror:=norm(col(d,1)-dcheck);
```

$$derror := 0.28\,10^{-8}$$

Appendix E

Example 4.1.1

Example 4.1.1

This example is used to demonstrate the individual steps of the reduction phase of block cyclic reduction.

```
> restart:  with(linalg):
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

```
> m:=7:  A:=matrix(3,3,[-4,1,0,1,-4,1,0,1,-4]); B:=matrix(3,3,0):
> d:=matrix(3,m,0):
```

$$A := \begin{bmatrix} -4 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & -4 \end{bmatrix}$$

```
> for i from 1 to 3 do
> B[i,i]:=1:
> od:
> print(B);
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

```
> for i from 1 to m do
> d[1,i]:=1;
> od:
> k:=ilog2(m+1);
```

$$k := 3$$

The First Reduction

```
> B1:=evalf(evalm(-1*B&*B&*inverse(A))); A1:=evalm(A+2*B1);
```

$$B1 := \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}$$

$$A1 := \begin{bmatrix} -3.464285714 & 1.142857143 & 0.03571428572 \\ 1.142857143 & -3.428571429 & 1.142857143 \\ 0.03571428572 & 1.142857143 & -3.464285714 \end{bmatrix}$$

```
> d1:=matrix(3,7,0):
> for j from 2 to 6 by 2 do
> col(d1,j):=evalf(evalm(col(d,j)-B&*inverse(A)&*(col(d,j-1)+col(d,j+1))
> ));
> od;
```

$$linalg : -col(d1, 2) := [1.535714286, 0.1428571429, 0.03571428571]$$

$$linalg : -col(d1, 4) := [1.535714286, 0.1428571429, 0.03571428571]$$

$$linalg : -col(d1, 6) := [1.535714286, 0.1428571429, 0.03571428571]$$

The Second Reduction

```
> B2:=evalf(evalm(-1*B1&*B1&*inverse(A1))); A2:=evalm(A1+2*B2);
```

$$B2 := \begin{bmatrix} 0.03125000002 & 0.02678571430 & 0.01339285715 \\ 0.02678571430 & 0.04464285716 & 0.02678571429 \\ 0.01339285715 & 0.02678571430 & 0.03125000002 \end{bmatrix}$$

$$A2 := \begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000002 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix}$$

```
> d2:=matrix(3,7,0):
> col(d2,4):=evalf(evalm(col(d1,4)-B1&*inverse(A1)&*(col(d1,2)+col(d1,6
> ))));
```

$$linalg : -col(d2, 4) := [1.857142858, 0.3750000001, 0.1428571429]$$

Appendix F

Example 4.2.1

Example 4.2.1

This example is used to demonstrate the individual steps of the back-substitution phase of block cyclic reduction.

```
> restart: with(linalg):
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

The Initial Solution

```
> A2:=matrix(3,3,[-3.401785714, 1.196428572, 0.06250000002,
> 1.196428572, -3.339285715, 1.196428572, 0.06250000002, 1.196428572,
> -3.401785714]); d2:=vector(3,[1.857142858,0.3750000001,
> 0.1428571429]);
```

$$A2 := \begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000002 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix}$$

$$d2 := [1.857142858, 0.3750000001, 0.1428571429]$$

```
> x4:=evalm(inverse(A2)&*d2);
```

$$x4 := [-0.7044655033, -0.4398060869, -0.2096201423]$$

The First Back-Substitution

```
> B1:=matrix(3,3,[0.2678571429, 0.07142857143, 0.01785714286,
> 0.07142857143, 0.2857142857, 0.07142857143, 0.01785714286,
> 0.07142857143, 0.2678571429]); A1:=matrix(3,3,[-3.464285714,
> 1.142857143, 0.03571428572, 1.142857143, -3.428571429,
> 1.142857143,0.03571428572, 1.142857143, -3.464285714]);
> d1:=vector(3,[1.535714286, 0.1428571429, 0.03571428571]);
```

$$B1 := \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}$$

$$A1 := \begin{bmatrix} -3.464285714 & 1.142857143 & 0.03571428572 \\ 1.142857143 & -3.428571429 & 1.142857143 \\ 0.03571428572 & 1.142857143 & -3.464285714 \end{bmatrix}$$

$$d1 := [1.535714286, 0.1428571429, 0.03571428571]$$

```
> x2:=evalm(inverse(A1)&*(d1-B1&*x4));
> x6:=evalm(inverse(A1)&*(d1-B1&*x4));
```

$$x2 := [-0.6290769962, -0.3621061140, -0.1651594705]$$

$$x6 := [-0.6290769962, -0.3621061140, -0.1651594705]$$

The Second Back-Substitution

```
> B:=matrix(3,3,[1,0,0,0,1,0,0,0,1]);
> A:=matrix(3,3,[-4,1,0,1,-4,1,0,1,-4]); d:=vector(3,[1,1,1]);
```

$$B := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$A := \begin{bmatrix} -4 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & -4 \end{bmatrix}$$

$$d := [1, 1, 1]$$

```
> x1:=evalm(inverse(A)&*(d-B&*x2));
> x3:=evalm(inverse(A)&*(d-B&*(x2+x4)));
> x5:=evalm(inverse(A)&*(d-B&*(x4+x6)));
> x7:=evalm(inverse(A)&*(d-B&*x6));
```

$$x1 := [-0.5544596226, -0.5887614944, -0.4384802411]$$

$$x3 := [-0.7783136772, -0.7797122083, -0.5386229554]$$

$$x5 := [-0.7783136772, -0.7797122083, -0.5386229554]$$

$$x7 := [-0.5544596226, -0.5887614944, -0.4384802411]$$

Appendix G

Example 4.3.1

Example 4.3.1

This example is designed to solve systems of linear equations of the form Tx=d. We are assuming that T is symmetric, block tridiagonal, diagonally dominant, and Toeplitz. The diagonal blocks are equal to A, and the off-diagonal blocks are equal to B. Also, we are assuming that m (the dimension of T) is 2^k-1.

```
> restart: with(linalg): printlevel:=1:
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

The Set-Up Phase

In this phase of the routine, we are defining and determining the parameters of the system. Because we will be modifying the values of A, B, and d[i] at each step of the reduction phase, we will use three-dimensional arrays to store the values.

```
> m:=1023: k:=ilog2(m+1); n:=3: s:=1:
```

$$k := 10$$

```
> A:=array(sparse,1..n,1..n,1..k): B:=array(sparse,1..n,1..n,1..k):
> d:=array(sparse,1..n,1..m+2,1..k):
> Atemp:=matrix(n,n,0): Btemp:=matrix(n,n,0): dtemp:=matrix(n,m,0):
> Anew:=matrix(n,n,0): Bnew:=matrix(n,n,0): dnew:=vector(n,0):
> dt:=vector(n,0): X:=matrix(n,m+2,0): x:=vector(n*m,0):
> xt1:=vector(n,0): xt2:=vector(n,0): errtemp:=vector(n,0):
> err:=vector(m,0): dcheck:=matrix(n,m+2,0):
> for i from 1 to n do
> A[i,i,1]:=-4;
> B[i,i,1]:=1;
> for j from 2 to m+1 do
> d[i,j,1]:=1;
> od;
> od:
> for i from 1 to n-1 do
> A[i,i+1,1]:=1;
> A[i+1,i,1]:=1;
> od:
```

The Reduction Phase

In this phase of the routine, we will be decoupling the variables from each other. This will allow us to solve a single matrix equation for a single vector variable for eventual use in the back-substitution.

```
> for r from 2 to k do
> for i from 1 to n do
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,r-1];
> Btemp[i,j]:=B[i,j,r-1];
> od;
> for j from 1 to m do
> dtemp[i,j]:=d[i,j+1,r-1];
```

```
> od;
> od;
> M:=evalf(evalm(-1*Btemp&*inverse(Atemp))):
> Bnew:=evalf(evalm(M&*Btemp)):
> Anew:=evalf(evalm(Atemp+2*Bnew));
> t:=s;
> s:=2*s;
> for j from s to m+1-s by s do
> dnew:=evalf(evalm(col(dtemp,j)+M&*(col(dtemp,j-t)+col(dtemp,j+t))));
> for i from 1 to n do
> d[i,j+1,r]:=dnew[i];
> od;
> od;
> print(dnew);
> for i from 1 to n do
> for j from 1 to n do
> A[i,j,r]:=Anew[i,j];
> B[i,j,r]:=Bnew[i,j];
> od;
> od;
> od;
```

$$M := \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.464285714 & 1.142857143 & 0.03571428572 \\ 1.142857143 & -3.428571429 & 1.142857143 \\ 0.03571428572 & 1.142857143 & -3.464285714 \end{bmatrix}$$

$$t := 1$$

$$s := 2$$

$$[1.714285714, 1.857142857, 1.714285714]$$

$$M := \begin{bmatrix} 0.09821428575 & 0.06250000004 & 0.02678571431 \\ 0.06250000003 & 0.1250000000 & 0.06250000003 \\ 0.02678571430 & 0.06250000004 & 0.09821428574 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.03125000002 & 0.02678571431 & 0.01339285716 \\ 0.02678571430 & 0.04464285715 & 0.02678571430 \\ 0.01339285715 & 0.02678571430 & 0.03125000001 \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000004 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix}$$

$$t := 2$$

$$s := 4$$

[2.375000000, 2.750000000, 2.375000000]

$$M := \begin{bmatrix} 0.01543754045 & 0.01723673580 & 0.01028290127 \\ 0.01723673580 & 0.02572044169 & 0.01723673580 \\ 0.01028290126 & 0.01723673580 & 0.01543754044 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.001081838848 & 0.001458437538 & 0.0009897917195 \\ 0.001458437537 & 0.002071630566 & 0.001458437537 \\ 0.0009897917191 & 0.001458437538 & 0.001081838847 \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399622036 & 1.199345447 & 0.06447958348 \\ 1.199345447 & -3.335142454 & 1.199345447 \\ 0.06447958346 & 1.199345447 & -3.399622036 \end{bmatrix}$$

$$t := 4$$

$$s := 8$$

[2.591974145, 3.055211419, 2.591974145]

$$M := \begin{bmatrix} 0.0006443266570 & 0.0008911491475 & 0.0006177549398 \\ 0.0008911491471 & 0.001262081595 & 0.0008911491471 \\ 0.0006177549397 & 0.0008911491472 & 0.0006443266566 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.2608191700\,10^{-5} & 0.3686798990\,10^{-5} & 0.2605745850\,10^{-5} \\ 0.3686798986\,10^{-5} & 0.5213937545\,10^{-5} & 0.3686798986\,10^{-5} \\ 0.2605745848\,10^{-5} & 0.3686798988\,10^{-5} & 0.2608191698\,10^{-5} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479497 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479495 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 8$$

$$s := 16$$

[2.603962009, 3.072162613, 2.603962009]

$$M := \begin{bmatrix} 0.1590945050\,10^{-5} & 0.2249434502\,10^{-5} & 0.1590238994\,10^{-5} \\ 0.2249434500\,10^{-5} & 0.3181184040\,10^{-5} & 0.2249434500\,10^{-5} \\ 0.1590238992\,10^{-5} & 0.2249434500\,10^{-5} & 0.1590945049\,10^{-5} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.1658646118\,10^{-10} & 0.2345679711\,10^{-10} & 0.1658645944\,10^{-10} \\ 0.2345679709\,10^{-10} & 0.3317292058\,10^{-10} & 0.2345679709\,10^{-10} \\ 0.1658645943\,10^{-10} & 0.2345679710\,10^{-10} & 0.1658646116\,10^{-10} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 16$$

$$s := 32$$

[2.603992398, 3.072205589, 2.603992398]

$$M := \begin{bmatrix} 0.1011992178\,10^{-10} & 0.1431173027\,10^{-10} & 0.1011992128\,10^{-10} \\ 0.1431173026\,10^{-10} & 0.2023984303\,10^{-10} & 0.1431173026\,10^{-10} \\ 0.1011992127\,10^{-10} & 0.1431173027\,10^{-10} & 0.1011992177\,10^{-10} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.6714147064\,10^{-21} & 0.9495237837\,10^{-21} & 0.6714147064\,10^{-21} \\ 0.9495237828\,10^{-21} & 0.1342829411\,10^{-20} & 0.9495237827\,10^{-21} \\ 0.6714147062\,10^{-21} & 0.9495237832\,10^{-21} & 0.6714147061\,10^{-21} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 32$$

$$s := 64$$

$$[2.603992398,\ 3.072205589,\ 2.603992398]$$

$$M := \begin{bmatrix} 0.4096512468\,10^{-21} & 0.5793343488\,10^{-21} & 0.4096512468\,10^{-21} \\ 0.5793343483\,10^{-21} & 0.8193024921\,10^{-21} & 0.5793343483\,10^{-21} \\ 0.4096512466\,10^{-21} & 0.5793343484\,10^{-21} & 0.4096512466\,10^{-21} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.1100183486\,10^{-41} & 0.1555894406\,10^{-41} & 0.1100183485\,10^{-41} \\ 0.1555894404\,10^{-41} & 0.2200366967\,10^{-41} & 0.1555894404\,10^{-41} \\ 0.1100183485\,10^{-41} & 0.1555894406\,10^{-41} & 0.1100183485\,10^{-41} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 64$$

$$s := 128$$

$$[2.603992398,\ 3.072205589,\ 2.603992398]$$

$$M := \begin{bmatrix} 0.6712565755\,10^{-42} & 0.9493001523\,10^{-42} & 0.6712565752\,10^{-42} \\ 0.9493001513\,10^{-42} & 0.1342513148\,10^{-41} & 0.9493001513\,10^{-42} \\ 0.6712565752\,10^{-42} & 0.9493001522\,10^{-42} & 0.6712565752\,10^{-42} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.2954021592\,10^{-83} & 0.4177617398\,10^{-83} & 0.2954021592\,10^{-83} \\ 0.4177617393\,10^{-83} & 0.5908043174\,10^{-83} & 0.4177617392\,10^{-83} \\ 0.2954021592\,10^{-83} & 0.4177617397\,10^{-83} & 0.2954021591\,10^{-83} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 128$$

$$s := 256$$

$$[2.603992398,\ 3.072205589,\ 2.603992398]$$

$$M := \begin{bmatrix} 0.1802341558\,10^{-83} & 0.2548895875\,10^{-83} & 0.1802341558\,10^{-83} \\ 0.2548895872\,10^{-83} & 0.3604683109\,10^{-83} & 0.2548895871\,10^{-83} \\ 0.1802341558\,10^{-83} & 0.2548895874\,10^{-83} & 0.1802341558\,10^{-83} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.2129662350\,10^{-166} & 0.3011797378\,10^{-166} & 0.2129662350\,10^{-166} \\ 0.3011797373\,10^{-166} & 0.4259324691\,10^{-166} & 0.3011797373\,10^{-166} \\ 0.2129662350\,10^{-166} & 0.3011797377\,10^{-166} & 0.2129662349\,10^{-166} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 256$$

$$s := 512$$

$$[2.603992398,\ 3.072205589,\ 2.603992398]$$

The Back-Substitution Phase

In this phase of the routine, we will use the dummy vector X to determine the solution. This will allow us to define x[0]=x[n+1]=0 (X[1] and X[m+2]) for use in the j-loop of the back-substitution. We will then eliminate these artificial entries to obtain our true solution x.

```
> for i from 1 to n do
> dt[i]:=d[i,s+1,k];
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,k];
> od;
> od:
> Xtemp:=evalf(evalm(inverse(Atemp)&*dt)):
> for i from 1 to n do
> X[i,s+1]:=Xtemp[i];
> od:
> for r from k-1 to 1 by -1 do
> for i from 1 to n do
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,r]:
> Btemp[i,j]:=B[i,j,r];
> od;
> for j from 1 to m do
> dtemp[i,j]:=d[i,j+1,r];
> od;
> od;
> t:=s;
> s:=s/2;
> for j from s to m+1-s by t do
> for i from 1 to n do
> xt1[i]:=X[i,j+1-s];
> xt2[i]:=X[i,j+1+s];
> od;
> Xtemp:=evalf(evalm(inverse(Atemp)&*(col(dtemp,j)-Btemp&*(xt1+xt2)))):
> for i from 1 to n do
> X[i,j+1]:=Xtemp[i]:
> od;
> od;
> od:
> for i from 1 to n do
> for j from 1 to m do
> x[n*j-i+1]:=X[i,j+1];
> od;
> od:
> print(x);
```

[-0.8019961987, -1.036102795, -0.8019961987, -1.171882000,
-1.540418783, -1.171882000, -1.345113019, -1.781808336, -1.345113019,
-1.426761742, -1.896588518, -1.426761742, -1.465345428, -1.951022256,
-1.465345428, -1.483597717, -1.976809647, -1.483597717, -1.492235793,
-1.989020896, -1.492235793, -1.496324557, -1.994802356, -1.496324557,
-1.498260075, -1.997539416, -1.498260076, -1.499176328, -1.998835161,
-1.499176329, -1.499610077, -1.999448566, -1.499610077, -1.499815412,
-1.999738952, -1.499815412, -1.499912617, -1.999876421, -1.499912617,
-1.499958633, -1.999941498, -1.499958634, -1.499980417, -1.999972306,
-1.499980417, -1.499990731, -1.999986890, -1.499990730, -1.499995612,
-1.999993794, -1.499995612, -1.499997923, -1.999997063, -1.499997923,
-1.499999017, -1.999998609, -1.499999017, -1.499999535, -1.999999341,
-1.499999535, -1.499999780, -1.999999688, -1.499999780, -1.499999896,
-1.999999853, -1.499999896, -1.499999950, -1.999999931, -1.499999950,
-1.499999977, -1.999999966, -1.499999977, -1.499999989, -1.999999985,
-1.499999989, -1.499999996, -1.999999993, -1.499999995, -1.499999998,
-1.999999996, -1.499999998, -1.499999999, -1.999999999, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,

```
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000,

 -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
```

-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,

-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,

-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,

```
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,


-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
```

-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,

-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,

-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,

```
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000,

-1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
```

-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.499999999, -1.999999999, -1.500000000, -1.499999998,
-1.999999996, -1.499999998, -1.499999996, -1.999999993, -1.499999995,
-1.499999989, -1.999999985, -1.499999989, -1.499999977, -1.999999966,
-1.499999977, -1.499999950, -1.999999931, -1.499999950, -1.499999896,
-1.999999853, -1.499999896, -1.499999780, -1.999999688, -1.499999780,
-1.499999535, -1.999999341, -1.499999535, -1.499999017, -1.999998609,
-1.499999017, -1.499997923, -1.999997063, -1.499997923, -1.499995612,
-1.999993794, -1.499995612, -1.499990731, -1.999986890, -1.499990730,
-1.499980417, -1.999972306, -1.499980417, -1.499958633, -1.999941498,
-1.499958634, -1.499912617, -1.999876421, -1.499912617, -1.499815412,
-1.999738952, -1.499815412, -1.499610077, -1.999448566, -1.499610077,
-1.499176328, -1.998835161, -1.499176329, -1.498260075, -1.997539416,
-1.498260076, -1.496324557, -1.994802356, -1.496324557, -1.492235793,
-1.989020896, -1.492235793, -1.483597717, -1.976809647, -1.483597717,
-1.465345428, -1.951022256, -1.465345428, -1.426761742, -1.896588518,
-1.426761742, -1.345113019, -1.781808336, -1.345113019, -1.171882000,
-1.540418783, -1.171882000, -0.8019961987, -1.036102795, -0.8019961987]

Check

In this phase of the routine, we test the reliability of the algorithm by determining the infinity-norm of Tx-d.

```
> for i from 1 to n do
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,1];
> Btemp[i,j]:=B[i,j,1];
> od;
> for j from 1 to m+2 do
> dcheck[i,j]:=d[i,j,1];
> od;
> od:
> for j from 2 to m+1 do
> errtemp:=evalf(evalm(col(dcheck,j) -
> (Btemp&*(col(X,j-1)+col(X,j+1)) + Atemp&*col(X,j)))):
> err[j-1]:=norm(errtemp):
> od:
> derror:=norm(err);
```

$$derror := 0.6\,10^{-8}$$

Appendix H

Example 5.3.1

Example 5.3.1

This program is designed to solve systems of linear equations of the form Tx=d using the truncated block cyclic reduction algorithm of Heller. We are assuming that T is symmetric, block tridiagonal, and diagonally dominant. The diagonal blocks of T are equal to A, and the off-diagonal blocks are equal to B. Also, we are assuming that the dimension of T (m) is 2^k-1.

```
> restart: with(linalg): printlevel:=1:
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

The Set-up Phase

In this phase of the routine, we are defining and determining the parameters of the system. Because we will be modifying A, B, and d[i] at each step of the reduction phase, we will use three-dimensional arrays to store their values.

```
> m:=1023: k:=ilog2(m+1); n:=3: s:=1: Digits:=10:
> epsilon:=evalf(1*10^(-Digits));
```

$$k := 10$$

$$\varepsilon := 0.1000000000\,10^{-9}$$

```
> A:=array(sparse,1..n,1..n,1..k): B:=array(sparse,1..n,1..n,1..k):
> d:=array(sparse,1..n,1..m+2,1..k):
> Atemp:=matrix(n,n,0): Btemp:=matrix(n,n,0): dtemp:=matrix(n,m,0):
> Anew:=matrix(n,n,0): Bnew:=matrix(n,n,0): dnew:=vector(n,0):
> dt:=vector(n,0): X:=matrix(n,m+2,0): x:=vector(n*m,0):
> xt1:=vector(n,0): xt2:=vector(n,0): errtemp:=vector(n,0):
> err:=vector(m,0): dcheck:=matrix(n,m+2,0):
> for i from 1 to n do
> A[i,i,1]:=-4;
> B[i,i,1]:=1;
> for j from 2 to m+1 do
> d[i,j,1]:=1;
> od;
> od:
> for i from 1 to n-1 do
> A[i,i+1,1]:=1;
> A[i+1,i,1]:=1;
> od:
> for i from 1 to n do
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,1];
> Btemp[i,j]:=B[i,j,1];
> od;
> od:
> G:=evalf(evalm(inverse(Atemp)&*Btemp)):
> gamma0:=2*norm(G,infinity);
```

$$\gamma 0 := 0.8571428570$$

```
> z:=evalf(log10(log10(epsilon)/log10(gamma0)));
```

$$z := 2.174270244$$

```
> ka:=min(k-1,ceil(z/log10(2)));
```

$$ka := 8$$

The Reduction Phase

In this phase of the routine, we are decoupling the vector variables from each other. This will allow us to solve a single matrix equation for a single vector variable, which we use in the eventual back-substitution.

```
> for r from 2 to ka+1 do
> for i from 1 to n do
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,r-1];
> Btemp[i,j]:=B[i,j,r-1];
> od;
> for j from 1 to m do
> dtemp[i,j]:=d[i,j+1,r-1];
> od;
> od;
> M:=evalf(evalm(-1*Btemp&*inverse(Atemp)));
> Bnew:=evalf(evalm(M&*Btemp));
> Anew:=evalf(evalm(Atemp+2*Bnew));
> t:=s;
> s:=2*s;
> for j from s to m+1-s by s do
> dnew:=evalf(evalm(col(dtemp,j)+M&*(col(dtemp,j-t)+col(dtemp,j+t))));
> for i from 1 to n do
> d[i,j+1,r]:=dnew[i];
> od;
> od;
> print(dnew);
> for i from 1 to n do
> for j from 1 to n do
> A[i,j,r]:=Anew[i,j];
> B[i,j,r]:=Bnew[i,j];
> od;
> od:
> od;
```

$$M := \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.2678571429 & 0.07142857143 & 0.01785714286 \\ 0.07142857143 & 0.2857142857 & 0.07142857143 \\ 0.01785714286 & 0.07142857143 & 0.2678571429 \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.464285714 & 1.142857143 & 0.03571428572 \\ 1.142857143 & -3.428571429 & 1.142857143 \\ 0.03571428572 & 1.142857143 & -3.464285714 \end{bmatrix}$$

$$t := 1$$

$$s := 2$$

$$[1.714285714, 1.857142857, 1.714285714]$$

$$M := \begin{bmatrix} 0.09821428575 & 0.06250000004 & 0.02678571431 \\ 0.06250000003 & 0.1250000000 & 0.06250000003 \\ 0.02678571430 & 0.06250000004 & 0.09821428574 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.03125000002 & 0.02678571431 & 0.01339285716 \\ 0.02678571430 & 0.04464285715 & 0.02678571430 \\ 0.01339285715 & 0.02678571430 & 0.03125000001 \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.401785714 & 1.196428572 & 0.06250000004 \\ 1.196428572 & -3.339285715 & 1.196428572 \\ 0.06250000002 & 1.196428572 & -3.401785714 \end{bmatrix}$$

$$t := 2$$

$$s := 4$$

$$[2.375000000, 2.750000000, 2.375000000]$$

$$M := \begin{bmatrix} 0.01543754045 & 0.01723673580 & 0.01028290127 \\ 0.01723673580 & 0.02572044169 & 0.01723673580 \\ 0.01028290126 & 0.01723673580 & 0.01543754044 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.001081838848 & 0.001458437538 & 0.0009897917195 \\ 0.001458437537 & 0.002071630566 & 0.001458437537 \\ 0.0009897917191 & 0.001458437538 & 0.001081838847 \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399622036 & 1.199345447 & 0.06447958348 \\ 1.199345447 & -3.335142454 & 1.199345447 \\ 0.06447958346 & 1.199345447 & -3.399622036 \end{bmatrix}$$

$$t := 4$$

$$s := 8$$

$$[2.591974145, 3.055211419, 2.591974145]$$

$$M := \begin{bmatrix} 0.0006443266570 & 0.0008911491475 & 0.0006177549398 \\ 0.0008911491471 & 0.001262081595 & 0.0008911491471 \\ 0.0006177549397 & 0.0008911491472 & 0.0006443266566 \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.2608191700\,10^{-5} & 0.3686798990\,10^{-5} & 0.2605745850\,10^{-5} \\ 0.3686798986\,10^{-5} & 0.5213937545\,10^{-5} & 0.3686798986\,10^{-5} \\ 0.2605745848\,10^{-5} & 0.3686798988\,10^{-5} & 0.2608191698\,10^{-5} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479497 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479495 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 8$$

$$s := 16$$

$$[2.603962009, 3.072162613, 2.603962009]$$

$$M := \begin{bmatrix} 0.1590945050\,10^{-5} & 0.2249434502\,10^{-5} & 0.1590238994\,10^{-5} \\ 0.2249434500\,10^{-5} & 0.3181184040\,10^{-5} & 0.2249434500\,10^{-5} \\ 0.1590238992\,10^{-5} & 0.2249434500\,10^{-5} & 0.1590945049\,10^{-5} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.1658646118\,10^{-10} & 0.2345679711\,10^{-10} & 0.1658645944\,10^{-10} \\ 0.2345679709\,10^{-10} & 0.3317292058\,10^{-10} & 0.2345679709\,10^{-10} \\ 0.1658645943\,10^{-10} & 0.2345679710\,10^{-10} & 0.1658646116\,10^{-10} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 16$$

$$s := 32$$

$$[2.603992398, 3.072205589, 2.603992398]$$

$$M := \begin{bmatrix} 0.1011992178\,10^{-10} & 0.1431173027\,10^{-10} & 0.1011992128\,10^{-10} \\ 0.1431173026\,10^{-10} & 0.2023984303\,10^{-10} & 0.1431173026\,10^{-10} \\ 0.1011992127\,10^{-10} & 0.1431173027\,10^{-10} & 0.1011992177\,10^{-10} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.6714147064\,10^{-21} & 0.9495237837\,10^{-21} & 0.6714147064\,10^{-21} \\ 0.9495237828\,10^{-21} & 0.1342829411\,10^{-20} & 0.9495237827\,10^{-21} \\ 0.6714147062\,10^{-21} & 0.9495237832\,10^{-21} & 0.6714147061\,10^{-21} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 32$$

$$s := 64$$

$$[2.603992398, 3.072205589, 2.603992398]$$

$$M := \begin{bmatrix} 0.4096512468\,10^{-21} & 0.5793343488\,10^{-21} & 0.4096512468\,10^{-21} \\ 0.5793343483\,10^{-21} & 0.8193024921\,10^{-21} & 0.5793343483\,10^{-21} \\ 0.4096512466\,10^{-21} & 0.5793343484\,10^{-21} & 0.4096512466\,10^{-21} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.1100183486\,10^{-41} & 0.1555894406\,10^{-41} & 0.1100183485\,10^{-41} \\ 0.1555894404\,10^{-41} & 0.2200366967\,10^{-41} & 0.1555894404\,10^{-41} \\ 0.1100183485\,10^{-41} & 0.1555894406\,10^{-41} & 0.1100183485\,10^{-41} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 64$$

$$s := 128$$

$$[2.603992398, 3.072205589, 2.603992398]$$

$$M := \begin{bmatrix} 0.6712565755\,10^{-42} & 0.9493001523\,10^{-42} & 0.6712565752\,10^{-42} \\ 0.9493001513\,10^{-42} & 0.1342513148\,10^{-41} & 0.9493001513\,10^{-42} \\ 0.6712565752\,10^{-42} & 0.9493001522\,10^{-42} & 0.6712565752\,10^{-42} \end{bmatrix}$$

$$Bnew := \begin{bmatrix} 0.2954021592\,10^{-83} & 0.4177617398\,10^{-83} & 0.2954021592\,10^{-83} \\ 0.4177617393\,10^{-83} & 0.5908043174\,10^{-83} & 0.4177617392\,10^{-83} \\ 0.2954021592\,10^{-83} & 0.4177617397\,10^{-83} & 0.2954021591\,10^{-83} \end{bmatrix}$$

$$Anew := \begin{bmatrix} -3.399616820 & 1.199352821 & 0.06448479500 \\ 1.199352821 & -3.335132026 & 1.199352821 \\ 0.06448479498 & 1.199352821 & -3.399616820 \end{bmatrix}$$

$$t := 128$$

$$s := 256$$

$$[2.603992398, 3.072205589, 2.603992398]$$

The Back-Substitution Phase

In this phase of the routine, we use the dummy vector X to determine our solution. This allows us to define x[0]=x[n+1]=0 (X[1] and X[m+2] in the routine) for the j-loop of the back-substitution. We will then eliminate these artificial entries to obtain our true solution x.

```
> for i from 1 to n do
> dt[i]:=d[i,s+1,ka];
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,ka];
> od;
> od:
> Xtemp:=evalf(evalm(inverse(Atemp)&*dt)):
> for i from 1 to n do
> for j from s to m+1-s do
> X[i,j+1]:=Xtemp[i];
> od;
> od:
> for r from ka to 1 by -1 do
> for i from 1 to n do
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,r];
> Btemp[i,j]:=B[i,j,r];
> od;
> for j from 1 to m do
> dtemp[i,j]:=d[i,j+1,r];
> od;
> od;
> t:=s:
> s:=s/2:
```

```
> for j from s to m+1-s by t do
> for i from 1 to n do
> xt1[i]:=X[i,j+1-s];
> xt2[i]:=X[i,j+1+s];
> od;
> Xtemp:=evalf(evalm(inverse(Atemp)&*(col(dtemp,j)-Btemp&*(xt1+xt2))));
> for i from 1 to n do
> X[i,j+1]:=Xtemp[i];
> od;
> od;
> od:
> for i from 1 to n do
> for j from 1 to m do
> x[n*j-i+1]:=X[i,j+1]:
> od:
> od:
> print(x);
```

```
[-0.8019961987, -1.036102795, -0.8019961987, -1.171882000,
-1.540418783, -1.171882000, -1.345113019, -1.781808336, -1.345113019,
-1.426761742, -1.896588518, -1.426761742, -1.465345428, -1.951022256,
-1.465345428, -1.483597717, -1.976809647, -1.483597717, -1.492235793,
-1.989020896, -1.492235793, -1.496324557, -1.994802356, -1.496324557,
-1.498260075, -1.997539416, -1.498260076, -1.499176328, -1.998835161,
-1.499176329, -1.499610077, -1.999448566, -1.499610077, -1.499815412,
-1.999738952, -1.499815412, -1.499912617, -1.999876421, -1.499912617,
-1.499958633, -1.999941498, -1.499958634, -1.499980417, -1.999972306,
-1.499980417, -1.499990731, -1.999986890, -1.499990730, -1.499995612,
-1.999993794, -1.499995612, -1.499997923, -1.999997063, -1.499997923,
-1.499999017, -1.999998609, -1.499999017, -1.499999535, -1.999999341,
-1.499999535, -1.499999780, -1.999999688, -1.499999780, -1.499999896,
-1.999999853, -1.499999896, -1.499999950, -1.999999931, -1.499999950,
-1.499999977, -1.999999966, -1.499999977, -1.499999989, -1.999999985,
-1.499999989, -1.499999996, -1.999999993, -1.499999995, -1.499999998,
-1.999999996, -1.499999998, -1.499999999, -1.999999999, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
```

-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,

-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,

-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,

-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,

-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,

-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,

-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,

-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000001,
-1.500000001, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,

-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000000,
-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000000, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000001, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000001, -1.500000001, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,

-1.500000000, -1.500000000, -2.000000001, -1.499999999, -1.500000000,
-2.000000000, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000000, -1.500000001, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000001, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000001, -1.500000001, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000000, -1.500000001, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.499999999, -1.500000000, -2.000000000, -1.500000000,
-1.500000000, -2.000000001, -1.499999999, -1.500000000, -2.000000001,
-1.500000000, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000000, -1.500000001, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000001, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000001, -1.500000001,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000000,
-1.500000001, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000001, -1.499999999,
-1.500000000, -2.000000000, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.500000000, -2.000000001, -1.500000000, -1.500000000,
-2.000000001, -1.500000000, -1.500000000, -2.000000000, -1.500000001,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.500000000, -1.500000001, -2.000000001, -1.500000000, -1.500000001,
-2.000000001, -1.500000001, -1.500000001, -2.000000001, -1.500000000,
-1.500000000, -2.000000001, -1.500000000, -1.500000000, -2.000000001,
-1.499999999, -1.499999999, -1.999999999, -1.500000000, -1.499999998,
-1.999999996, -1.499999998, -1.499999996, -1.999999993, -1.499999995,
-1.499999989, -1.999999985, -1.499999989, -1.499999977, -1.999999966,
-1.499999977, -1.499999950, -1.999999931, -1.499999950, -1.499999896,
-1.999999853, -1.499999896, -1.499999780, -1.999999688, -1.499999780,
-1.499999535, -1.999999341, -1.499999535, -1.499999017, -1.999998609,
-1.499999017, -1.499997923, -1.999997063, -1.499997923, -1.499995612,
-1.999993794, -1.499995612, -1.499990731, -1.999986890, -1.499990730,
-1.499980417, -1.999972306, -1.499980417, -1.499958633, -1.999941498,
-1.499958634, -1.499912617, -1.999876421, -1.499912617, -1.499815412,
-1.999738952, -1.499815412, -1.499610077, -1.999448566, -1.499610077,
-1.499176328, -1.998835161, -1.499176329, -1.498260075, -1.997539416,
-1.498260076, -1.496324557, -1.994802356, -1.496324557, -1.492235793,
-1.989020896, -1.492235793, -1.483597717, -1.976809647, -1.483597717,
-1.465345428, -1.951022256, -1.465345428, -1.426761742, -1.896588518,
-1.426761742, -1.345113019, -1.781808336, -1.345113019, -1.171882000,
-1.540418783, -1.171882000, -0.8019961987, -1.036102795, -0.8019961987]

Check

In this phase of the routine, we will test the reliability of the algorithm by measuring the infinity-norm of Tx-d.

```
> for i from 1 to n do
> for j from 1 to n do
> Atemp[i,j]:=A[i,j,1];
> Btemp[i,j]:=B[i,j,1];
> od;
> for j from 1 to m+2 do
> dcheck[i,j]:=d[i,j,1];
> od;
> od:
> for j from 2 to m+1 do
> errtemp:=evalf(evalm(col(dcheck,j) -
> (Btemp&*(col(X,j-1)+col(X,j+1)) + Atemp&*col(X,j)))):
> err[j-1]:=norm(errtemp):
> od:
> derror:=norm(err);
```

$$derror := 0.6\,10^{-8}$$

Bibliography

[1] S. Bondeli, W. Gander, *Cyclic Reduction for Special Tridiagonal Systems*, SIAM J. Matrix Anal. Appl., **15** (1994), 321-330.

[2] R. L. Burden, J. D. Faires, *Numerical Analysis*, seventh ed., Brooks/Cole, Pacific Grove, CA, 2001.

[3] B. L. Buzbee, F. W. Dorr, *The Direct Solution of the Biharmonic Equation on Rectangular Regions and the Poisson Equation on Irregular Regions*, SIAM J. Numer. Anal., **11** (1974), 753-763.

[4] B. L. Buzbee, F. W. Dorr, J. A. George, G. H. Golub, *The Direct Solution of the Discrete Poisson Equation on Irregular Regions*, SIAM J. Numer. Anal., **8** (1971), 722-736.

[5] B. L. Buzbee, G. H. Golub, C. W. Nielson, *On Direct Methods for Solving Poisson's Equation*, SIAM J. Numer. Anal., **7** (1970), 627-656.

[6] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[7] F. W. Dorr, *The Direct Solution of the Discrete Poisson Equation on a Rectangle*, SIAM Review, **12** (1970), 248-263.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 1983.

[9] D. Heller, *Some Aspects of the Cyclic Reduction Algorithm for Block Tridiagonal Linear Systems*, SIAM J. Numer. Anal., **13** (1976), 484-496.

[10] R. W. Hockney, *A Fast Direct Solution of Poisson's Equation Using Fourier Analysis*, J. Assoc. Comp. Mach, **12** (1965), 95-113.

[11] P. N. Swarztrauber, *A Direct Method for the Discrete Solution of Separable Elliptic Equations*, SIAM J. Numer. Anal., **11** (1974), 1136-1150.

[12] P. N. Swarztrauber, *The Methods of Cyclic Reduction, Fourier Analysis and the FACR Algorithm for the Discrete Solution of Poisson's Equation on a Rectangle*, SIAM Review, **19** (1977), 490-501.

[13] P. N. Swarztrauber, R. A. Sweet, *The Direct Solution of the Discrete Poisson Equation on a Disk*, SIAM J. Numer. Anal., **10** (1973), 900-907.

[14] R. A. Sweet, *A Generalized Cyclic Reduction Algorithm*, SIAM J. Numer. Anal., **11** (1974), 506-520.

[15] R. A. Sweet, *A Cyclic Reduction Algorithm for Solving Block Tridiagonal Systems of Arbitrary Dimensions*, SIAM J. Numer. Anal., **14** (1977), 706-720.