Tutorial Letter 202/1/2015

NUMERICAL METHODS 1

COS2633

Semester 1

Department of Mathematical Sciences

This tutorial letter contains solutions
for assignment 02

BAR CODE

Introduction

By this time you should have received the following tutorial matter. If you have not received all these tutorial matter, please contact the Department of Dispatch at the telephone number given in the inventory you received upon registration.
Tutorial letters:

- COS2633/101/3/2015 General information about the module and the assignments
- COS2633/102/3/2015 Background material
- COS2633/201/1/2015 Solutions to assignment 1
- COS2633/202/1/2015 This letter: Solutions to assignment 2

1 Discussion

1.1 The marking

The marking of this assignment was automated and performed at the assignment division of UNISA. The result published to students is therefore not, in any case, under lecturers control. All the questions were marked.

Next we present a model solution on which the marking was based.

Question 1

Given

$$\begin{array}{rcrcrcrcr} 2.141x_1 & - & 2.718x_2 & + & 1.414x_3 & - & 1.732x_4 & = & 3.316 \\ 9.869x_1 & + & 2.718x_2 & - & 7.389x_3 & + & 0.428x_4 & = & 0 \\ 2.236x_1 & - & 2.449x_2 & + & x_3 & - & 1.414x_4 & = & 3.141 \\ 31.006x_1 & + & 7.389x_2 & - & 2.645x_3 & + & 0.111x_4 & = & 1.414 \end{array}$$

(1.1) Matrix Notation.

$$\begin{bmatrix} 2.141 & -2.718 & 1.414 & -1.732 \\ 9.869 & 2.718 & -7.389 & 0.428 \\ 2.236 & -2.449 & 1 & -1.414 \\ 31.006 & 7.389 & -2.645 & 0.111 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3.316 \\ 0 \\ 3.141 \\ 1.414 \end{bmatrix}$$

(1.2) Solving the system

(a) Gaussian elimination without pivoting

When the number of linear equations is not very large, the elimination methods are the most important methods of solving the set, either by hand or computer. The most basic technique is usually attributed to Gauss, for example the Gaussian elimination method with or without pivoting, and the Gauss-Jordan method. The Gaussian elimination method is described in [1, pp.358-378]. In this section, we solve the given system using Gaussian elimination without pivoting. Thus we obtain the following:

$$A^{(1)}|b^{(1)} = \left[\begin{array}{cccc|c} 2.141 & -2.718 & 1.414 & -1.732 & 3.316 \\ 9.869 & 2.718 & -7.389 & 0.428 & 0 \\ 2.236 & -2.449 & 1 & -1.414 & 3.141 \\ 31.006 & 7.389 & -2.645 & 0.111 & 1.414 \end{array}\right] \begin{array}{l} \\ Row_2 \leftarrow Row_2 - \frac{9.869}{2.141} Row_1 \\ Row_3 \leftarrow Row_3 - \frac{2.236}{2.141} Row_1 \\ Row_4 \leftarrow Row_4 - \frac{31.006}{2.141} Row_1 \end{array}$$

$$A^{(2)}|b^{(2)} = \left[\begin{array}{cccc|c} 2.1410 & -2.7180 & 1.4140 & -1.7320 & 3.3160 \\ 0 & 15.2467 & -13.9069 & 8.4117 & -15.2852 \\ 0 & 0.3896 & -0.4767 & 0.3949 & -0.3221 \\ 0 & 46.7511 & -23.1226 & 25.1939 & -46.6084 \end{array}\right] \begin{array}{l} \\ \\ Row_3 \leftarrow Row_3 - \frac{0.3896}{15.2467} Row_2 \\ Row_4 \leftarrow Row_4 - \frac{46.7511}{15.2467} Row_2 \end{array}$$

$$A^{(3)}|b^{(3)} = \left[\begin{array}{cccc|c} 2.1410 & -2.7180 & 1.4140 & -1.7320 & 3.3160 \\ 0 & 15.2467 & -13.9069 & 8.4117 & -15.2852 \\ 0 & 0 & -0.1214 & 0.1799 & 0.0684 \\ 0 & 0 & 19.5202 & -0.5990 & 0.2608 \end{array}\right] \begin{array}{l} \\ \\ \\ Row_4 \leftarrow Row_4 - \frac{19.5202}{-0.1214} Row_3 \end{array}$$

$$A^{(4)}|b^{(4)} = \left[\begin{array}{cccc|c} 2.1410 & -2.7180 & 1.4140 & -1.7320 & 3.3160 \\ 0 & 15.2467 & -13.9069 & 8.4117 & -15.2852 \\ 0 & 0 & -0.1214 & 0.1799 & 0.0684 \\ 0 & 0 & 0 & 28.3342 & 11.2691 \end{array}\right]$$

Thus, from back-substitution, we get

$$\begin{aligned} x_4 &= \frac{11.2691}{28.3342} && = 0.3977 \\ x_3 &= \frac{0.0684 - 0.1799 \times 0.3977}{-0.1214} && = 0.0256 \\ x_2 &= \frac{-15.2852 - 8.4117 \times 0.3977 - (-13.9069) \times 0.0256}{15.2467} && = -1.1986 \\ x_1 &= \frac{3.316 - (-1.732) \times 0.3977 - 1.414 \times 0.0256 - (-2.718) \times (-1.1986)}{2.1410} && = 0.3320 \end{aligned}$$

Any error obtained from the computation of x_1, x_2, x_3 and x_4 using Gaussian elimination would just be round-off errors.

A variant of the Gaussian elimination method is to make the diagonal elements ones at the same time that the reduction is performed. Using this variant, we obtain:

$$A^{(2)}|b^{(2)} = \left[\begin{array}{cccc|c} 1.0000 & -1.2695 & 0.6604 & -0.8090 & 1.5488 \\ 0 & 15.2467 & -13.9069 & 8.4117 & -15.2852 \\ 0 & 0.3896 & -0.4767 & 0.3949 & -0.3221 \\ 0 & 46.7511 & -23.1226 & 25.1939 & -46.6084 \end{array}\right] \begin{array}{l} Row_2 \leftarrow \frac{1}{15.2467} Row_2 \\ Row_3 \leftarrow Row_3 - 0.3896 Row_2 \\ Row_4 \leftarrow Row_4 - 46.7511 Row_2 \end{array}$$

$$A^{(3)}|b^{(3)} = \left[\begin{array}{cccc|c} 1.0000 & -1.2695 & 0.6604 & -0.8090 & 1.5488 \\ 0 & 1.0000 & -0.9121 & 0.5517 & -1.0025 \\ 0 & 0 & -0.1214 & 0.1799 & 0.0684 \\ 0 & 0 & 19.5202 & -0.5990 & 0.2608 \end{array}\right] \quad \begin{array}{c} Row_3 \leftarrow \frac{1}{-0.1214} Row_3 \\ Row_4 \leftarrow Row_4 - 19.5202 Row_3 \end{array}$$

$$A^{(4)}|b^{(4)} = \left[\begin{array}{cccc|c} 1.0000 & -1.2695 & 0.6604 & -0.8090 & 1.5488 \\ 0 & 1.0000 & -0.9121 & 0.5517 & -1.0025 \\ 0 & 0 & 1.0000 & -1.4822 & -0.5639 \\ 0 & 0 & 0 & 28.3342 & 11.2691 \end{array}\right]$$

And then, from back-substitution,

$$\begin{aligned} x_4 &= \frac{11.2691}{28.3342} &&= 0.3977 \\ x_3 &= -0.5639 - (-1.4822) \times 0.3977 &&= 0.0256 \\ x_2 &= -1.0025 - 0.5517 \times 0.3977 - (-0.9121) \times 0.0256 &&= -1.1986 \\ x_1 &= 1.5488 - (-0.809) \times 0.3977 - 0.6604 \times 0.0256 - 1.2695 \times 1.1986 &&= 0.3320 \end{aligned}$$

(b) Gaussian elimination with scaled partial pivoting

This is not to be confused with Gaussian elimination with partial pivoting. The simple Gaussian elimination used above uses the rows 1, 2, ..., n *as the pivot equations (hence without pivoting). A better approach called Gaussian elimination with pivoting, is to pivot on row* l_1*, then row* l_2*, and so on, until finally pivoting on row* l_{n-1} *for some permutation* $\{l_i\}_{i=1}^n$ *of the integers* 1, 2, ..., n*. The methods (with partial pivoting and with scaled partial pivoting) are described in [1, pp.373-379]. The object of scaled partial pivoting is to adjust the coefficients to make the largest entry in each row to be of the same relative magnitude as* 1 *before the comparison for row interchange is performed. The principle is as follows:*

- *We move from the first column to the last. Let's call the matrix* $A = (a_{ij})_{1 \le i,j \le n}$.
- *On each column (say column* i*) of the matrix* A*, we construct a vector* SF *(my notation) of scaling factors as follows: For each* j $(i \le j \le n)$*, we divide the absolute value at row* j *(column* i*) by the maximum absolute value at row* j *for all the columns, that is,* $SF(j) = \dfrac{|a_{ji}|}{\max\limits_{1 \le k \le n} |a_{jk}|}$*. We can choose* $SF(j) = 0$ *for* $j < i$.
- *If* p *is the value (index) for which* $SF(p)$ *has the highest value, we swap row* i *and row* p *if* $p \neq i$*, assuming that* we are choosing among the indices greater or equal to i.
- *We make the elements below the diagonal zero in column* i *by row operations.*
- *We move to the next column and repeat the process above on the new matrix.*

In the following we do not actually show the interchanging of rows, but we indicate which rows are interchanged (if applicable). Again, our system in augmented form

is given by:

$$A^{(1)}|b^{(1)} = \left[\begin{array}{cccc|c} 2.141 & -2.718 & 1.414 & -1.732 & 3.316 \\ 9.869 & 2.718 & -7.389 & 0.428 & 0 \\ 2.236 & -2.449 & 1 & -1.414 & 3.141 \\ 31.006 & 7.389 & -2.645 & 0.111 & 1.414 \end{array}\right]$$

Then the scale factor s_i for each row is given as

(1) $$s_i = \max_{1\leq j\leq n} |a_{ij}|$$

From (1), we obtain

$$s = [2.718, 9.869, 2.449, 31.006]$$

The appropriate row interchange to place zeros in the first column is determined by choosing the least p such that

$$\frac{|a_{p1}|}{s_p} = \max_{1\leq k\leq n} \frac{|a_{k1}|}{s_k} \qquad \left[\textit{that is, } SF(p) = \max_{1\leq k\leq n} SF(k) \textit{ in column 1}\right]$$

and interchanging row 1 and row p (if $p \neq 1$). Then we have

$$\frac{|a_{11}|}{s_1} = 0.7877, \quad \frac{|a_{21}|}{s_2} = 1.0000, \quad \frac{|a_{31}|}{s_3} = 0.9130, \quad \frac{|a_{41}|}{s_4} = 1.0000$$

Since 2 is the least integer p such that

$$\frac{|a_{p1}|}{s_p} = \max_{1\leq k\leq 4} \frac{|a_{k1}|}{s_k}$$

a row interchange (swap rows 1 and 2) is required to place zeros in the first column. Performing the first three eliminations, we obtain

$$A^{(2)}|b^{(2)} = \left[\begin{array}{cccc|c} 9.869 & 2.718 & -7.389 & 0.428 & 0 \\ 0 & -3.3076 & 3.0170 & -1.8249 & 3.3160 \\ 0 & -3.0648 & 2.6741 & -1.5110 & 3.1410 \\ 0 & -1.1503 & 20.5694 & -1.2337 & 1.4140 \end{array}\right]$$

Similarly, before making zero the entries below the diagonal in the second column using the Gaussian elimination procedure, we select the smallest integer $p \geq 2$ such that

$$\frac{|a_{p2}|}{s_p} = \max_{2\leq k\leq 4} \frac{|a_{k2}|}{s_k}$$

and interchange row 2 and row p (if $p \neq 2$). Now for the second column we have

$$\frac{|a_{22}|}{s_2} = 0.3352, \quad \frac{|a_{32}|}{s_3} = 1.2515, \quad \frac{|a_{42}|}{s_4} = 0.0371$$

Therefore we need to interchange rows 2 and 3 in order to place zeros below the diagonal in the second column. So we have

$$A^{(3)}|b^{(3)} = \left[\begin{array}{cccc|c} 9.8690 & 2.7180 & -7.3890 & 0.4280 & 0 \\ 0 & -3.0648 & 2.6741 & -1.5110 & 3.1410 \\ 0 & 0 & 0.1310 & -0.1942 & -0.0739 \\ 0 & 0 & 19.5658 & -0.6666 & 0.2351 \end{array}\right]$$

Finally for the third column we have

$$\frac{|a_{33}|}{s_3} = 0.0535, \quad \frac{|a_{43}|}{s_4} = 0.6310$$

Therefore we need to interchange rows 3 and 4 in order to place zeros below the diagonal in the third column. So we have

$$A^{(4)}|b^{(4)} = \left[\begin{array}{cccc|c} 9.8690 & 2.7180 & -7.3890 & 0.4280 & 0 \\ 0 & -3.0648 & 2.6741 & -1.5110 & 3.1410 \\ 0 & 0 & 19.5658 & -0.6666 & 0.2351 \\ 0 & 0 & 0 & -0.1897 & -0.0754 \end{array}\right]$$

And then, from back-substitution,

$$\begin{aligned} x_4 &= \frac{-0.0754}{-0.1897} &&= 0.3977 \\ x_3 &= \frac{0.2351 - (-0.6666) \times 0.3977}{19.5658} &&= 0.0256 \\ x_2 &= \frac{3.1410 - (-1.5110 \times 0.3977 - 2.6741 \times 0.0256}{-3.0648} &&= -1.1986 \\ x_1 &= \frac{0 - 0.428 \times 0.3977 - (-7.389) \times 0.0256 - 2.718 \times (-1.1986)}{9.869} &&= 0.3320 \end{aligned}$$

(c) LU decomposition

The idea behind the LU decomposition is to use Gaussian elimination to express the given square matrix (not the augmented matrix) as the product of a lower triangular matrix L and an upper triangular matrix U ($A = LU$). The procedure is described in [1, pp.400-406], especially in Theorem 6.19 (see [1, p.403]). This gives

$$\begin{aligned} A &= \begin{bmatrix} 2.141 & -2.718 & 1.414 & -1.732 \\ 9.869 & 2.718 & -7.389 & 0.428 \\ 2.236 & -2.449 & 1 & -1.414 \\ 31.006 & 7.389 & -2.645 & 0.111 \end{bmatrix} \\ &= \begin{bmatrix} 1.0000 & & & \\ 4.6095 & 1.0000 & & \\ 1.0444 & 0.0256 & 1.0000 & \\ 14.4820 & 3.0663 & -160.8244 & 1.0000 \end{bmatrix} \begin{bmatrix} 2.1410 & -2.7180 & 1.4140 & -1.7320 \\ & 15.2467 & -13.9069 & 8.4117 \\ & & -0.1214 & 0.1799 \\ & & & 28.3342 \end{bmatrix} \end{aligned}$$

Notice that the entries of the upper triangular matrix U are the same as the entries obtained by applying the Gaussian elimination without pivoting as done above. On the other hand, the entries in L are obtained by replacing the zeros below the main

diagonal of the identity matrix (from the Gaussian elimination process) with the ratios of coefficients (i.e., the multipliers) at each step.
Since $A = LU$, we have $Ax = b \iff LUx = b$. Therefore if we let $y = Ux$, then $LUx = b \iff Ly = b$. Hence the system can be solved in two steps:

- *Solve $Ly = b$ to find y*
- *Solve $Ux = y$ to find x*

Following this procedure, the first step gives

$$\begin{bmatrix} 1.0000 & & & \\ 4.6095 & 1.0000 & & \\ 1.0444 & 0.0256 & 1.0000 & \\ 14.4820 & 3.0663 & -160.8244 & 1.0000 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 3.316 \\ 0 \\ 3.141 \\ 1.414 \end{bmatrix}$$

Using forward substitution, we obtain

$$\begin{aligned} y_1 &= 3.3160 \\ y_2 &= -15.2952 \\ y_3 &= 0.0684 \\ y_4 &= 11.2691 \end{aligned}$$

The second step gives

$$\begin{bmatrix} 2.1410 & -2.7180 & 1.4140 & -1.7320 \\ & 15.2467 & -13.9069 & 8.4117 \\ & & -0.1214 & 0.1799 \\ & & & 28.3342 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3.3160 \\ -15.2952 \\ 0.0684 \\ 11.2691 \end{bmatrix}$$

Using back-substitution, we obtain

$$\begin{aligned} x_4 &= 0.3977 \\ x_3 &= 0.0256 \\ x_2 &= -1.1986 \\ x_1 &= 0.3320 \end{aligned}$$

(1.3) Number of Arithmetic Operations

The efficiency of a numerical method is usually measured by counting the number of arithmetic operations required. The augmented matrix is $n \times n+1$ in size, where $n = 4$. The number of arithmetic operations required for the Gaussian elimination method is calculated as follows:

- ***Triangularization part***
  - *Divisions:* $\sum_{i=1}^{n-1}(n-i) = \sum_{i=1}^{n-1} i = (n^2 - n)/2$;
  - *Multiplications:* $\sum_{i=1}^{n-1}(n-i+1)(n-i) = \sum_{i=1}^{n-1} i(i+1) = (n^3 - n)/3$;
  - *Subtractions: Same as multiplications;*

This makes a total of $2n^3/3 + n^2/2 - 7n/6$ *operations;*

- ***Back-substitution part***
  - *Multiplications:* $\sum_{i=1}^{n-1}(n-i) = \sum_{i=1}^{n-1} i = (n^2-n)/2$;
  - *Subtractions: Same as multiplications;*
  - *Divisions:* n;

  This makes a total of n^2 *operations;*

So the combined number of arithmetic operations is $2n^3/3 + 3n^2/2 - 7n/6 = 62$.

(1.4) Solving $Ax = b$.

(a) Let A be the matrix from (1.1) and $\omega = 1$.

(i) *From (1.1), we have*

$$A = \begin{bmatrix} 2.141 & -2.718 & 1.414 & -1.732 \\ 9.869 & 2.718 & -7.389 & 0.428 \\ 2.236 & -2.449 & 1 & -1.414 \\ 31.006 & 7.389 & -2.645 & 0.111 \end{bmatrix}$$

and therefore

$$M = \begin{bmatrix} 2.141 & 0 & 0 & 0 \\ 0 & 2.718 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.111 \end{bmatrix}$$

Now, using algorithm,

$$Mx^{(k)} = \omega b + (M - \omega A)\, x^{(k-1)}$$

with $\omega = 1$ *and starting at* $x^{(0)} = (3,0,3,1)^T$, *we obtain*

$$\begin{bmatrix} 2.141 & 0 & 0 & 0 \\ 0 & 2.718 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.111 \end{bmatrix} \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ x_3^{(1)} \\ x_4^{(1)} \end{bmatrix} = \begin{bmatrix} 3.316 \\ 0 \\ 3.141 \\ 1.414 \end{bmatrix} - \begin{bmatrix} 0 & -2.718 & 1.414 & -1.732 \\ 9.869 & 0 & -7.389 & 0.428 \\ 2.236 & -2.449 & 0 & -1.414 \\ 31.006 & 7.389 & -2.645 & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 0 \\ 3 \\ 1 \end{bmatrix}$$

and after calculation we get

$$x^{(1)} = (0.3765, -2.8948, -2.1530, -753.7748)^T$$

Applying similarly iterations, we have

$$x^{(2)} = (-610.4837, 111.4760, -1070.6276, 48.9757)^T$$

and

$$x^{(3)} = (889.7717, -701.6061, 1710.4390, 137608.6923)^T$$

(ii) *The Jacobi method is explained in [1, pp.450-454] and the algorithm is given in [1, pp.453-454]. The explicit formula is*

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{\substack{j=1\\ j\neq i}}^{n} a_{ij}x_j^{(k)}\right] = \frac{b_i}{a_{ii}} - \sum_{\substack{j=1\\ j\neq i}}^{n} \frac{a_{ij}}{a_{ii}}x_j^{(k)} \qquad \text{for } i = 1, 2, \ldots, n$$

and the matrix notation actually coincides with the given algorithm, namely

$$Mx^{(k)} = \omega b + (M - \omega A)\, x^{(k-1)}$$

Moreover, a sufficient (but not necessary) condition for this algorithm to converge is for the matrix A to be diagonally dominant, that is,

$$|a_{ii}| > \sum_{\substack{j=1\\ j\neq i}}^{n} |a_{ij}| \qquad \text{for } i = 1, 2, \ldots, n$$

By checking this condition on the matrix A given in this question, we see for example that $|a_{11}| < |a_{12}| + |a_{13}| + |a_{14}|$, which means our matrix A is not diagonally dominant. Therefore we don't know if the Jacobi method will converge or not.

Using the explicit formula for the Jacobi method, our equation can be written as

$$\begin{aligned}
x_1^{(k+1)} &= \frac{1}{2.141}\left[3.316 + 2.718x_2^{(k)} - 1.414x_3^{(k)} + 1.732x_4^{(k)}\right] \\
x_2^{(k+1)} &= \frac{1}{2.718}\left[0 - 9.869x_1^{(k)} + 7.389x_3^{(k)} - 0.428x_4^{(k)}\right] \\
x_3^{(k+1)} &= \frac{1}{1.000}\left[3.141 - 2.236x_1^{(k)} + 2.449x_2^{(k)} + 1.414x_4^{(k)}\right] \\
x_4^{(k+1)} &= \frac{1}{0.111}\left[1.414 - 31.006x_1^{(k)} - 7.389x_2^{(k)} + 2.645x_3^{(k)}\right]
\end{aligned}$$

We have $x^{(0)} = (3, 0, 3, 1)^T$ hence

Iteration 1

$$\begin{aligned}
x_1^{(1)} &= \frac{1}{2.141}\left[3.316 + 2.718 \times 0 - 1.414 \times 3 + 1.732 \times 1\right] \\
&= 0.3765 \\
x_2^{(1)} &= \frac{1}{2.718}\left[0 - 9.689 \times 3 + 7.389 \times 3 - 0.428 \times 1\right] \\
&= -2.8948 \\
x_3^{(1)} &= \frac{1}{1.000}\left[3.141 - 2.236 \times 3 + 2.449 \times 0 + 1.414 \times 1\right] \\
&= -2.1530 \\
x_4^{(1)} &= \frac{1}{0.111}\left[1.414 - 31.006 \times 3 - 7.389 \times 0 + 2.645 \times 3\right] \\
&= -753.7748
\end{aligned}$$

Iteration 2

$$
\begin{aligned}
x_1^{(2)} &= \frac{1}{2.141}\left[3.316 + 2.718 \times (-2.8948) - 1.414 \times (-2.1530) + 1.732 \times (-753.7748)\right] \\
&= -610.4837 \\
x_2^{(2)} &= \frac{1}{2.718}\left[0 - 9.689 \times 0.3765 + 7.389 \times (-2.1530) - 0.428 \times (-753.7748)\right] \\
&= 111.4760 \\
x_3^{(2)} &= \frac{1}{1.000}\left[3.141 - 2.236 \times 0.3765 + 2.449 \times (-2.8948) + 1.414 \times (-753.7748)\right] \\
&= -1070.6276 \\
x_4^{(2)} &= \frac{1}{0.111}\left[1.414 - 31.006 \times 0.3765 - 7.389 \times (-2.8948) + 2.645 \times (-2.1530)\right] \\
&= 48.9757
\end{aligned}
$$

Iteration 3

$$
\begin{aligned}
x_1^{(3)} &= \frac{1}{2.141}\left[3.316 + 2.718 \times (111.4760) - 1.414 \times (-1070.6276) + 1.732 \times 48.9757\right] \\
&= 889.7717 \\
x_2^{(3)} &= \frac{1}{2.718}\left[0 - 9.689 \times (-610.4837) + 7.389 \times (-1070.6276) - 0.428 \times 48.9757\right] \\
&= -701.6061 \\
x_3^{(3)} &= \frac{1}{1.000}\left[3.141 - 2.236 \times (-610.4837) + 2.449 \times (111.4760) + 1.414 \times 48.9757\right] \\
&= 1710.4390 \\
x_4^{(3)} &= \frac{1}{0.111}\left[1.414 - 31.006 \times (-610.4837) - 7.389 \times (111.4760) + 2.645 \times (-1070.6276)\right] \\
&= 137608.6923
\end{aligned}
$$

In brief, after three iterations of the Jacobi algorithm, starting at $x^{(0)} = (3, 0, 3, 1)^T$*, we obtain the following results*

$$
\begin{aligned}
x^{(1)} &= (0.3765, -2.8948, -2.1530, -753.7748)^T \\
x^{(2)} &= (-610.4837, 111.4760, -1070.6276, 48.9757)^T \\
x^{(3)} &= (889.7717, -701.6061, 1710.4390, 137608.6923)^T
\end{aligned}
$$

It is clear that in this case the Jacobi method does not converge.

(iii) *The algorithm suggested here, with the given matrix* M *being the diagonal part of the matrix* A*, coincides with the matrix formulation of the Jacobi method, therefore the results are the same, as to be expected.*

(b) Let A be the matrix from (1.1) and $\omega = 1$.

(i) *Here we choose* M *to be the lower triangular part of* A *including the diagonal entries, that is*

$$
M = \begin{bmatrix} 2.141 & 0 & 0 & 0 \\ 9.869 & 2.718 & 0 & 0 \\ 2.236 & -2.449 & 1 & 0 \\ 31.006 & 7.389 & -2.645 & 0.111 \end{bmatrix}
$$

Again, using the given algorithm, and starting at $x^{(0)} = (3, 0, 3, 1)^T$, *we obtain*

$$\begin{bmatrix} 2.141 & 0 & 0 & 0 \\ 9.869 & 2.718 & 0 & 0 \\ 2.236 & -2.449 & 1 & 0 \\ 31.006 & 7.389 & -2.645 & 0.111 \end{bmatrix} \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ x_3^{(1)} \\ x_4^{(1)} \end{bmatrix} = \begin{bmatrix} 3.316 \\ 0 \\ 3.141 \\ 1.414 \end{bmatrix} - \begin{bmatrix} 0 & -2.718 & 1.414 & -1.732 \\ 0 & 0 & -7.389 & 0.428 \\ 0 & 0 & 0 & -1.414 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 0 \\ 3 \\ 1 \end{bmatrix}$$

and after calculation we get

$$x^{(1)} = (0.3765, 6.6312, 19.9532, -58.3845)^T$$

Applying similarly iterations, we have

$$x^{(2)} = (-50.4418, 246.5905, 637.2733, 12873.3750)^T$$

and

$$x^{(3)} = (10307.8608, -37722.3215, -97224.2488, -2684973.1714)^T$$

(ii) *The Gauss-Seidel method is explained in [1, pp.454-457] and the algorithm is given in [1, p.456]. The explicit formula is*

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right] = \frac{b_i}{a_{ii}} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(k+1)} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(k)}$$

for $i = 1, 2, \ldots, n$, *and the matrix notation, again, coincides with the given algorithm.*

Moreover, as in the Jacobi case, a sufficient (but not necessary) condition for this algorithm to converge is for the matrix A *to be diagonally dominant, that is,*

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \qquad \text{for } i = 1, 2, \ldots, n$$

By checking this condition on the matrix A *given in this question, again, we see that our matrix* A *is not diagonally dominant. Therefore we don't know if the Gauss-Seidel method will converge or not. However, it does happen that with some systems, the Gauss-Seidel method converges after many iterations when the Jacobi method does not.*

The difference between the Jacobi and the Gauss-Seidel methods is that in the Gauss-Seidel method, the newly computed values are used in the next equation. For example, if x_1 *is computed, this value is used to compute* x_2, *and so on.*

Using the explicit formula for the Gauss-Seidel method, our equation can be written as

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{2.141} \left[3.316 + 2.718 x_2^{(k)} - 1.414 x_3^{(k)} + 1.732 x_4^{(k)} \right] \\ x_2^{(k+1)} &= \frac{1}{2.718} \left[0 - 9.869 x_1^{(k+1)} + 7.389 x_3^{(k)} - 0.428 x_4^{(k)} \right] \\ x_3^{(k+1)} &= \frac{1}{1.000} \left[3.141 - 2.236 x_1^{(k+1)} + 2.449 x_2^{(k+1)} + 1.414 x_4^{(k)} \right] \\ x_4^{(k+1)} &= \frac{1}{0.111} \left[1.414 - 31.006 x_1^{(k+1)} - 7.389 x_2^{(k+1)} + 2.645 x_3^{(k+1)} \right] \end{aligned}$$

We have $x^{(0)} = (3, 0, 3, 1)^T$ *hence*

Iteration 1

$$
\begin{aligned}
x_1^{(1)} &= \frac{1}{2.141}\left[3.316 + 2.718 \times 0 - 1.414 \times 3 + 1.732 \times 1\right] \\
&= 0.3765 \\
x_2^{(1)} &= \frac{1}{2.718}\left[0 - 9.689 \times 0.3765 + 7.389 \times 3 - 0.428 \times 1\right] \\
&= 6.6312 \\
x_3^{(1)} &= \frac{1}{1.000}\left[3.141 - 2.236 \times 0.3765 + 2.449 \times 6.6312 + 1.414 \times 1\right] \\
&= 19.9532 \\
x_4^{(1)} &= \frac{1}{0.111}\left[1.414 - 31.006 \times 0.3765 - 7.389 \times 6.6312 + 2.645 \times 19.9532\right] \\
&= -58.3845
\end{aligned}
$$

Iteration 2

$$
\begin{aligned}
x_1^{(2)} &= \frac{1}{2.141}\left[3.316 + 2.718 \times 6.6312 - 1.414 \times 19.9532 + 1.732 \times (-58.3845)\right] \\
&= -50.4418 \\
x_2^{(2)} &= \frac{1}{2.718}\left[0 - 9.689 \times (-50.4418) + 7.389 \times (-2.1530) - 0.428 \times (-753.7748)\right] \\
&= 246.5905 \\
x_3^{(2)} &= \frac{1}{1.000}\left[3.141 - 2.236 \times (-50.4418) + 2.449 \times 246.5905 + 1.414 \times (-753.7748)\right] \\
&= 637.2733 \\
x_4^{(2)} &= \frac{1}{0.111}\left[1.414 - 31.006 \times (-50.4418) - 7.389 \times 246.5905 + 2.645 \times 637.2733\right] \\
&= 12873.3750
\end{aligned}
$$

Iteration 3

$$
\begin{aligned}
x_1^{(3)} &= \frac{1}{2.141}\left[3.316 + 2.718 \times 246.5905 - 1.414 \times 637.2733 + 1.732 \times 12873.3750\right] \\
&= 10307.8608 \\
x_2^{(3)} &= \frac{1}{2.718}\left[0 - 9.689 \times 10307.8608 + 7.389 \times (-1070.6276) - 0.428 \times 48.9757\right] \\
&= -37722.3215 \\
x_3^{(3)} &= \frac{1}{1.000}\left[3.141 - 2.236 \times 10307.8608 + 2.449 \times (-37722.3215) + 1.414 \times 48.9757\right] \\
&= -97224.2488 \\
x_4^{(3)} &= \frac{1}{0.111}\left[1.414 - 31.006 \times 10307.8608 - 7.389 \times (-37722.3215) + 2.645 \times (-97224.2488)\right] \\
&= -2684973.1714
\end{aligned}
$$

In brief, after three iterations of the Gauss-Seidel algorithm, starting at $x^{(0)} = (3, 0, 3, 1)^T$, *we obtain the following results*

$$x^{(1)} = (0.3765, 6.6312, 19.9532, -58.3845)^T$$
$$x^{(2)} = (-50.4418, 246.5905, 637.2733, 12873.3750)^T$$
$$x^{(3)} = (10307.8608, -37722.3215, -97224.2488, -2684973.1714)^T$$

This method also does not converge.

(iii) *The algorithm suggested here, with the given matrix* M *being the lower triangular part of the matrix* A, *coincides with the matrix formulation of the Gauss-Seidel method, therefore the results are the same, as to be expected.*

(c) Solving the system using SOR with $\omega = 0.5$ and $x_0 = (3, 0, 3, 1)^T$.

By doing what is called over-relaxation, *the Gauss-Seidel method can be speeded up. The successive over-relaxation method (SOR) is described in [1, pp.462-467], the explicit formula is in [1, p.464] and the algorithm in [1, p.467]. The explicit formula is*

$$\begin{aligned} x_i^{(k+1)} &= (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right] \\ &= x_i^{(k)} + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i}^{n} a_{ij}x_j^{(k)}\right] \end{aligned}$$

Applying the SOR algorithm to our question with $\omega = 0.5$ *we get*

$$x_1^{(k+1)} = (1-0.5)x_1^{(k)} + \frac{0.5}{2.141}\left[3.316 + 2.718x_2^{(k)} - 1.414x_3^{(k)} + 1.732x_4^{(k)}\right]$$
$$x_2^{(k+1)} = (1-0.5)x_2^{(k)} + \frac{0.5}{2.718}\left[0 - 9.869x_1^{(k+1)} + 7.389x_3^{(k)} - 0.428x_4^{(k)}\right]$$
$$x_3^{(k+1)} = (1-0.5)x_3^{(k)} + \frac{0.5}{1.000}\left[3.141 - 2.236x_1^{(k+1)} + 2.449x_2^{(k+1)} + 1.414x_4^{(k)}\right]$$
$$x_4^{(k+1)} = (1-0.5)x_4^{(k)} + \frac{0.5}{0.111}\left[1.414 - 31.006x_1^{(k+1)} - 7.389x_2^{(k+1)} + 2.645x_3^{(k+1)}\right]$$

We have $x^{(0)} = (3, 0, 3, 1)^T$, *hence*

Iteration 1

$$x_1^{(1)} = (1 - 0.5) \times 3 + \frac{0.5}{2.141} [3.316 + 2.718 \times 0 - 1.414 \times 3 + 1.732 \times 1]$$
$$= 2.4167$$
$$x_2^{(1)} = (1 - 0.5) \times 0 + \frac{0.5}{2.718} [0 - 9.869 \times 2.4167 + 7.389 \times 3 - 0.428 \times 1]$$
$$= 0.6167$$
$$x_3^{(1)} = (1 - 0.5) \times 3 + \frac{0.5}{1.000} [3.141 - 2.236 \times 2.4167 + 2.449 \times 0.6167 + 1.414 \times 1]$$
$$= 0.5883$$
$$x_4^{(1)} = (1 - 0.5) \times 1 + \frac{0.5}{0.111} [1.414 - 31.006 \times 2.4167 - 7.389 \times 0.6167 + 2.645 \times 0.5833]$$
$$= 0.4671$$

Iteration 2

$$x_1^{(2)} = (1 - 0.5) \times 2.4167 + \frac{0.5}{2.141} [3.316 + 2.718 \times 0.6167 - 1.414 \times 0.5833 + 1.732 \times 0.4671]$$
$$= 2.5593$$
$$x_2^{(2)} = (1 - 0.5) \times 0.6167 + \frac{0.5}{2.718} [0 - 9.869 \times 2.5593 + 7.389 \times 0.5883 - 0.428 \times 0.4671]$$
$$= 0.6824$$
$$x_3^{(2)} = (1 - 0.5) \times 0.5833 + \frac{0.5}{1.000} [3.141 - 2.236 \times 2.5593 + 2.449 \times 0.6824 + 1.414 \times 0.4671]$$
$$= 0.7260$$
$$x_4^{(2)} = (1 - 0.5) \times 0.4671 + \frac{0.5}{0.111} [1.414 - 31.006 \times 2.5593 - 7.389 \times 0.6824 + 2.645 \times 0.7260]$$
$$= 0.6317$$

Iteration 3

$$x_1^{(3)} = (1 - 0.5) \times 2.5593 + \frac{0.5}{2.141} [3.316 + 2.718 \times 0.6824 - 1.414 \times 0.7260 + 1.732 \times 0.6317]$$
$$= 2.3968$$
$$x_2^{(3)} = (1 - 0.5) \times 0.6824 + \frac{0.5}{2.718} [0 - 9.869 \times 2.3968 + 7.389 \times 0.7260 - 0.428 \times 0.6317]$$
$$= 0.6879$$
$$x_3^{(3)} = (1 - 0.5) \times 0.7260 + \frac{0.5}{1.000} [3.141 - 2.236 \times 2.3968 + 2.449 \times 0.6879 + 1.414 \times 0.6317]$$
$$= 0.7808$$
$$x_4^{(3)} = (1 - 0.5) \times 0.6317 + \frac{0.5}{0.111} [1.414 - 31.006 \times 2.3968 - 7.389 \times 0.6879 + 2.645 \times 0.7808]$$
$$= 0.7235$$

In brief, after three iterations of the SOR algorithm with $\omega = 0.5$*, starting at* $x^{(0)} = (3, 0, 3, 1)^T$*, we obtain the following results*

$$x^{(1)} = (1.6882, 0.9341, 3.0339, -223.8640)^T$$
$$x^{(2)} = (-89.3398, 184.4122, 170.5102, 8265.8226)^T$$
$$x^{(3)} = (3360.2464, -6427.3201, -5696.2468, -319117.0484)^T$$

The SOR method also does not converge.
Note that if $\omega = 1$ *then the SOR explicit formula becomes*

$$x_i^{(k+1)} = \frac{b_i}{a_{ii}} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(k+1)} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(k)}$$

which is exactly the explicit formula for the Gauss-Seidel algorithm.

(d) (Comparison of techniques

There are three basic methods for solving a set of linear equations, namely:

- *Direct methods*
- *Iterative methods*
- *Error function minimization methods*

Direct methods *A direct method is a non-iterative method for solving a set of linear equations. The most common direct methods are:*

- *The Gaussian elimination methods*
- *Matrix decomposition methods, for instance the* LU *and* $L^T L$ *method*

For small matrices as in our cases, a direct method like that of Gaussian elimination can be used with or without pivoting. Pivot selection is necessary if the diagonal elements of the coefficient matrix A *are not higher in magnitude than the other elements in the corresponding rows.*
A decomposition method is a method where the matrix A *is decomposed in two triangular matrices* L *and* U*, so that* $A = LU$*. An advantage of using this method in computer programs is that it is economical in terms of storage space. If the matrix* A *is symmetric, which is not always the case, and which is not the case here, it can be decomposed in two symmetric triangular matrices so that* $A = L^T L$*. This is the so-called* Cholesky decomposition*.*

Iterative methods *In these methods, the solution of a set of linear equations is obtained by using an iterative scheme. These methods are very useful for large, sparse systems. The most common iterative methods are:*

- *The Jacobi method*
- *The Gauss-Seidel method*

- *The Gauss-Seidel method with over-relaxation (GSOR or simply SOR)*

The Gauss-Seidel method will, in general converge more rapidly than the Jacobi. Moreover, the GSOR method with $\omega = 1$ is the same as the Gauss-Seidel. The GSOR method is very popular. For an iterative method, it is important (preferable) that the matrix be diagonally dominant and that a reasonable initial starting vector (approximation to the solution, or initial guess) is known. As we could see from our results, the iterative methods are not appropriate for solving our system of equations, this is because the coefficient matrix, A is not diagonally dominant, not positive definite, not symmetric and not tridiagonal.

Error function minimization methods *An error function minimization method is one in which the error function $r = a\bar{x} - b$, with $\bar{x}$ the trial vector, is to be minimized. These methods are discussed in [1, pp.462-463].*

SUMMARY *In choosing a method for solving a set of linear equations, the following points have to be taken into consideration:*

- *The number of equations to be solved*
- *The characteristics of the coefficient matrix A (sparse, diagonally dominant, ill-conditioned, and so on . . .)*

The set of equations given in Problem 1 is a small set of order 4, and the coefficient matrix is not diagonally dominant, that is,

$$|a_{ii}| < \sum_{j \neq i} |a_{ij}| \quad \text{for some values of } i$$

The best method for our problem is therefore the Gaussian elimination method or the LU ($L^T L$) decomposition.

Question 2

Given

$$A = \begin{bmatrix} 3 & 3/2 & 1 \\ 3/2 & 1 & 3/4 \\ 1 & 3/4 & 3/5 \end{bmatrix}$$

We want to find the inverse of A using Gaussian elimination.
The method is described in [1, pp.386-389].
Gaussian elimination produces the LU decomposition of the given matrix A, where L is a lower triangular and U an upper triangular matrix. We augment the given matrix with the identity matrix of the same order. We now transform the original matrix to an upper triangular matrix by using elementary row transformations. The elements below *the diagonal are made zero via these elementary row transformations. A is replaced by:*

$$(A|I)^{(1)} = \left[\begin{array}{ccc|ccc} 3 & 3/2 & 1 & 1 & 0 & 0 \\ 3/2 & 1 & 3/4 & 0 & 1 & 0 \\ 1 & 3/4 & 3/5 & 0 & 0 & 1 \end{array}\right] \quad \begin{array}{c} Row_2 \leftarrow Row_2 - \frac{3/2}{3} Row_1 \\ Row_3 \leftarrow Row_3 - \frac{1}{3} Row_1 \end{array}$$

$$(A|I)^{(2)} = \left[\begin{array}{ccc|ccc} 3 & 3/2 & 1 & 1 & 0 & 0 \\ 0 & 1/4 & 1/4 & -1/2 & 1 & 0 \\ 0 & 1/4 & 4/15 & -1/3 & 0 & 1 \end{array}\right] \quad Row_3 \leftarrow Row_3 - \tfrac{1/4}{1/4} Row_2$$

$$(A|I)^{(3)} = \left[\begin{array}{ccc|ccc} 3 & 3/2 & 1 & 1 & 0 & 0 \\ 0 & 1/4 & 1/4 & -1/2 & 1 & 0 \\ 0 & 0 & 1/60 & 1/6 & -1 & 1 \end{array}\right]$$

The first three columns of the augmented matrix above constitute, as required by Gaussian elimination, an upper triangular matrix U (and the remaining columns constitute a lower triangular matrix L with 1 on the diagonal, and the two matrices L and U constitute the LU decomposition of A). And the back-substitution process can be applied after augmenting *the matrix U with one column of the matrix L at a time, to find the entries of the corresponding column in the matrix $B = A^{-1}$. Thus*

$$\left[\begin{array}{ccc|c} 3 & 3/2 & 1 & 1 \\ 0 & 1/4 & 1/4 & -1/2 \\ 0 & 0 & 1/60 & 1/6 \end{array}\right] \implies \begin{array}{rcr} b_{31} & = & 10 \\ b_{21} & = & -12 \\ b_{11} & = & 3 \end{array}$$

$$\left[\begin{array}{ccc|c} 3 & 3/2 & 1 & 0 \\ 0 & 1/4 & 1/4 & 1 \\ 0 & 0 & 1/60 & -1 \end{array}\right] \implies \begin{array}{rcr} b_{32} & = & -60 \\ b_{22} & = & 64 \\ b_{12} & = & -12 \end{array}$$

$$\left[\begin{array}{ccc|c} 3 & 3/2 & 1 & 0 \\ 0 & 1/4 & 1/4 & 0 \\ 0 & 0 & 1/60 & 1 \end{array}\right] \implies \begin{array}{rcr} b_{33} & = & 60 \\ b_{23} & = & -60 \\ b_{13} & = & 10 \end{array}$$

An alternative would be to proceed with further row operations on $(A|I)^{(3)} = (U|L)$, with the purpose of transforming the first three columns of the augmented matrix to the identity matrix. This gives:

$$(A|I)^{(3)} = \left[\begin{array}{ccc|ccc} 3 & 3/2 & 1 & 1 & 0 & 0 \\ 0 & 1/4 & 1/4 & -1/2 & 1 & 0 \\ 0 & 0 & 1/60 & 1/6 & -1 & 1 \end{array}\right] \quad Row_3 \leftarrow \tfrac{1}{1/60} Row_3$$

$$\left[\begin{array}{ccc|ccc} 3 & 3/2 & 1 & 1 & 0 & 0 \\ 0 & 1/4 & 1/4 & -1/2 & 1 & 0 \\ 0 & 0 & 1 & 10 & -60 & 60 \end{array}\right] \quad Row_2 \leftarrow \tfrac{1}{1/4}(Row_2 - \tfrac{1}{4} Row_3)$$

$$\left[\begin{array}{ccc|ccc} 3 & 3/2 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -12 & 64 & -60 \\ 0 & 0 & 1 & 10 & -60 & 60 \end{array}\right] \quad Row_1 \leftarrow \tfrac{1}{3}(Row_1 - \tfrac{3}{2} Row_2 - 1 Row_3)$$

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 3 & -12 & 10 \\ 0 & 1 & 0 & -12 & 64 & -60 \\ 0 & 0 & 1 & 10 & -60 & 60 \end{array}\right]$$

Hence the inverse of A is

$$B = A^{-1} = \begin{bmatrix} 3 & -12 & 10 \\ -12 & 64 & -60 \\ 10 & -60 & 60 \end{bmatrix}$$

We work with fractions to obtain exact values. Check this result by convincing yourself that indeed $AB = I$.

Question 3

In order to use Newton's method to approximate the solution of the given non-linear system, we need initial values for x and y. From the question, we use the initial values $x = 1$ and $y = -1$.
Read [1, pp.638-642] carefully and convince yourself that the $i+1$-st iteration of Newton's method involves two steps:

(i) Solve the system of linear equations

$$\begin{bmatrix} 2x^{(i)} & 2y^{(i)} \\ 3\left(x^{(i)}\right)^2 & 3\left(y^{(i)}\right)^2 \end{bmatrix} \begin{bmatrix} \Delta x^{(i)} \\ \Delta y^{(i)} \end{bmatrix} = - \begin{bmatrix} \left(x^{(i)}\right)^2 + \left(y^{(i)}\right)^2 - 5 \\ \left(x^{(i)}\right)^3 + \left(y^{(i)}\right)^3 - 2 \end{bmatrix}$$

for $\Delta x^{(i)}$ and $\Delta y^{(i)}$.

(ii) Perform

$$\begin{bmatrix} x^{(i+1)} \\ y^{(i+1)} \end{bmatrix} = \begin{bmatrix} x^{(i)} \\ y^{(i)} \end{bmatrix} + \begin{bmatrix} \Delta x^{(i)} \\ \Delta y^{(i)} \end{bmatrix}$$

to produce the $(i+1)$-st approximation $\left[x^{(i+1)}, y^{(i+1)}\right]^T$.

Note that the superscript (i) denotes the approximation obtained in the i-th iteration. Here we use the starting values $\left(x^{(0)}, y^{(0)}\right) = (1, -1)$.

The first 5 iterations are shown in the table below.

i	x	y
0	1.0000	−1.0000
1	2.0833	−1.4167
2	1.7605	−1.4159
3	1.7105	−1.4413
4	1.7094	−1.4415
5	1.7094	−1.4415

This is obtained from the following computer code (written in MATLAB), which suggests that Newton's method converges, since the process stops after 5 iterations.

```
x(1)=1; y(1)=-1; dxy=[1;1]; tol=10^(-5);
i=1;
while (sqrt(dxy(1)^2+dxy(2)^2)>tol)
    jac=[2*x(i),2*y(i);3*x(i)^2,3*y(i)^2];
    b=[x(i)^2+y(i)^2-5;x(i)^3+y(i)^3-2];
    dxy=-inv(jac)*b;
    x(i+1)=x(i)+dxy(1);
    y(i+1)=y(i)+dxy(2);
    i=i+1;
end
```

To perform exactly three iterations of the Newton's algorithm, the code is changed as follows:

```
x(1)=1; y(1)=-1; dxy=[1;1];
for i=1:3
    jac=[2*x(i),2*y(i);3*x(i)^2,3*y(i)^2];
    b=[x(i)^2+y(i)^2-5;x(i)^3+y(i)^3-2];
    dxy=-inv(jac)*b;
    x(i+1)=x(i)+dxy(1);
    y(i+1)=y(i)+dxy(2);
end
```

Question 4

(4.1)

The required difference table (note that these are differences, not divided differences, as can be seen in [1, p.129]) is as follows:

x	f	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$
-0.5	5				
		10			
0	15		-16		
		-6		16	
0.5	9		0		12
		-6		4	
1	3		4		
		-2			
1.5	1				

On the other hand, considering the relationship

$$f_i^{[k]} = f\left[x_i, x_{i+1}, \ldots, x_{i+k}\right] = \frac{1}{k!h^K}\Delta^k f\left(x_i\right)$$

where $h = x_{i+1} - x_i$ *and* $s = \frac{x-x_0}{h}$*, the divided-difference table is as follows:*

x_i	f_i	$f_i^{[1]}$	$f_i^{[2]}$	$f_i^{[3]}$	$f_i^{[4]}$
-0.5	5				
		20			
0	15		-32		
		-12		$64/3$	
0.5	9		0		-8
		-12		$16/3$	
1	3		8		
		-4			
1.5	1				

(4.2)

From the theory, we know that the $n-$th order difference of any $n-$th degree polynomial is constant. Therefore, we require a fourth *degree polynomial to fit the five data points in the given table exactly.*

(4.3)

The (forward) Newton-Gregory interpolating polynomials of degree n (for equispaced points) are as follows (see [1, pp. 129-131]):

$$P_n(x) = f(x_0) + \sum_{k=1}^{n} \binom{s}{k} \Delta^k f(x_0) \tag{2}$$

Since we can always (uniquely) fit a second *degree polynomial through any* three *data points, we consider*

$$P_2(x) = f(x_0) + \binom{s}{1} \Delta f(x_0) + \binom{s}{2} \Delta^2 f(x_0) \tag{3}$$

and to fit $P_2(x)$ to the points with $x-$values 0.5, 1 *and* 1.5, *we let $x_0 = 0.5$ so that $s = 2x - 1$. (Remember that $h = \frac{1}{2}$). Simplifying and substituting s by $2x - 1$ in (3) yields*

$$P_2(x) = f(x_0) + (2x-1)\Delta f(x_0) + (2x-1)(x-1)\Delta^2 f(x_0).$$

From our difference table we know that $f(x_0) = 9$, $\Delta f(x_0) = -6$ and $\Delta^2 f(x_0) = 4$. So

$$P_2(x) = 9 - 6(2x-1) + 4(2x-1)(x-1) = 8x^2 - 24x + 19. \tag{4}$$

We may now approximate $f(1.25)$ by means of our interpolating polynomial (4):

$$f(1.25) \approx 8(1.25)^2 - 24(1.25) + 19 = 1.5.$$

(4.4)

The error of $P_n(x)$, an $n-$th degree interpolating polynomial, is given by

$$E(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_n) f^{(n+1)}(\xi)}{(n+1)!}$$

with ξ on the smallest interval that contains $\{x_0, x_1, \ldots, x_n, x\}$ [1, p. 135]. In the case of equispaced points, we may express $E(x)$ in terms of $s = (x - x_0)/h$:

$$E(x) = \binom{s}{n+1} h^{n+1} f^{(n+1)}(\xi),$$

with ξ as before. Since $f(x)$ and therefore also $f^{(n+1)}(x)$ are not known, we use

$$f^{(n+1)} \approx \frac{\Delta^{n+1} f(x)}{h^{n+1}},$$

yielding

(5) $$E(x) \approx \binom{s}{n+1} \Delta^{n+1} f(x).$$

As expected, this is just the first neglected term in (2).
Using (5), we obtain the following approximation to the error in (4.3):

$$E(x) \approx \frac{2(2(1.25)-1)(1.25-1)(2(1.25)-3)\Delta^3 f(x_0)}{6}$$

(4.5)

The $n-$th degree Lagrange interpolating polynomial through the points $(x_0, f(x_0))$, $(x_1, f(x_1)), \ldots, (x_n, f(x_n))$ is

$$P_n(x) = \sum_{j=0}^{n} L_{n,j}(x) f(x_j) \quad \text{where} \quad L_{n,j}(x) = \prod_{\substack{i=0 \\ i \neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)}.$$

Given the data points $(0.5, 9)$, $(1, 3)$ and $(1.5, 1)$, we fit the second degree Lagrange interpolating polynomial:

$$\begin{aligned} P_2(x) &= \left[\frac{(x-1)(x-1.5)}{(0.5-1)(0.5-1.5)}\right] 9 + \left[\frac{(x-0.5)(x-1.5)}{(1-0.5)(1-1.5)}\right] 3 + \left[\frac{(x-0.5)(x-1)}{(1.5-0.5)(1.5-1)}\right] 1 \\ &= 18(x-1)(x-3/2) - 12(x-1/2)(x-3/2) + 2(x-1/2)(x-1) \end{aligned}$$

So,

$$P_2(1.25) = 1.5.$$

As expected we obtain the same result as in (4.3), since the $n-$th degree polynomial that passes through $n+1$ data points, is unique.

(4.6)

Splines form an important and very useful class of approximating functions. In this module we are only concerned with cubic *splines. One possible reason for this is that they are in some sense* both *complex enough to be useful* and *simple enough to use with ease. We give a formal definition of cubic spline interpolation to help you understand the formulas used in solving the problem of the question. This definition does not appear* explicitly *in [1], so please take note!*

Construction of a Cubic spline interpolant

Given a function f on $[a, b]$ and a set of so-called nodes, $a = x_0 < x_1 < \cdots < x_n = b$, a cubic spline interpolant, Q, for f is a function that satisfies the following conditions:

(a) Q is a cubic polynomial, denoted Q_i, on the subinterval $[x_i, x_{i+1}]$ for each $i = 0, 1, \ldots, n-1$.

(b) $Q(x_i) = f(x_i)$ for each $i = 0, 1, \ldots, n$.

(c) $Q_{i+1}(x_{i+1}) = Q_i(x_{i+1})$ for each $i = 0, 1, \ldots, n-2$.

(d) $Q'_{i+1}(x_{i+1}) = Q'_i(x_{i+1})$ *for each* $i = 0, 1, \ldots, n-2$.

(e) $Q''_{i+1}(x_{i+1}) = Q''_i(x_{i+1})$ *for each* $i = 0, 1, \ldots, n-2$.

(f) The following boundary conditions (specific to a natural *cubic spline) are satisfied:*

$$Q''(x_0) = Q''(x_n) = 0.$$

Here, the conditions (a), (b), (c), (d) and (e) define a cubic spline. If on top of that, condition (f) is satisfied, then we have a natural cubic spline. However, condition (f) can be replaced by something else (a different set of boundary conditions), and this leads to a different type of cubic spline. Note that [1] uses $S(x)$ *for* $Q(x)$*. Also notice how we use* $Q(x)$ *when referring to the spline interpolant in general, but* $Q_i(x)$ *if we want to refer to the cubic polynomial on subinterval* $[x_i, x_{i+1}]$*. In the rest of our discussion we use whichever notation is clearer or more convenient.*

In principle, *to construct a cubic spline interpolant as defined above, we must determine* $4n$ *coefficients,* four *for each* Q_i*,* $i = 0, 1, \ldots, n-1$*. To be able to do this in a unique way, we need* $4n$ *(independent) equations. Where do they come from? Well, requirements (b), (c), (d) and (e) of the definition provides us with* $n+1$*,* $n-1$*,* $n-1$ *and* $n-1$ *respectively, a total of* $4n-2$*. The remaining two are obtained from the set of boundary conditions we use.*

Following the approach in [1], we let y_i *denote* $f(x_i)$*,* $i = 0, 1, \ldots, n$*,* $h_i = x_{i+1} - x_i$*, and write the cubic polynomial* Q_i *for the* $i-th$ *subinterval as*

(6) $$Q_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3.$$

Then

(7) $$Q'_i(x) = b_i + 2c_i(x - x_i) + 3d_i(x - x_i)^2$$

and

(8) $$Q''_i(x) = 2c_i + 6d_i(x - x_i).$$

So by condition (b) of the definition and (6), we have

(9) $$y_i = a_i$$

and

(10) $$y_{i+1} = y_i + b_i h_i + c_i h_i^2 + d_i h_i^3.$$

By condition (e) of the definition and (8), we have

(11) $$S_i = Q''(x_i) = Q''_i(x_i) = 2c_i$$

and

(12) $$S_{i+1} = Q''(x_{i+1}) = Q''_i(x_{i+1}) = 2c_i + 6d_i h_i.$$

By (11) and (12), we get

(13) $$c_i = \frac{S_i}{2}$$

and

(14) $$d_i = \frac{S_{i+1} - S_i}{6h_i}.$$

By (14), (13), (9) and (10), we get

$$y_{i+1} = \frac{S_{i+1} - S_i}{6h_i}h_i^3 + \frac{S_i}{2}h_i^2 + b_i h_i + y_i.$$

Solving for b_i yields

(15) $$b_i = \frac{y_{i+1} - y_i}{h_i} - \frac{2h_i S_i + h_i S_{i+1}}{6}.$$

By (b) and (d) of the definition (7), we get

$$y_i' = b_i$$

and

$$y_i' = b_{i-1} + 2c_{i-1}h_{i-1} + 3d_{i-1}h_{i-1}^2.$$

So

(16) $$b_{i-1} + 2c_{i-1}h_{i-1} + 3d_{i-1}h_{i-1}^2.$$

By (16), (15), (14), (13) and (9), and simplifying we obtain

(17) $$h_{i-1}S_{i-1} + (2h_{i-1} + 2h_i)S_i + h_i S_{i+1} = 6(f\,[x_i, x_{i+1}] - f\,[x_{i-1}, x_i])$$

for $i = 1, 2, \ldots, n-1$, i.e. $n-1$ equations in the $n+1$ unknowns S_i, $i = 0, 1, \ldots, n$. Notice that these equations hold irrespective *of the set of boundary conditions. Choosing the set of boundary conditions supplies us with* two *additional equations by which we can eliminate S_0 and S_n as unknowns, leaving us with an $(n-1)-$dimensional linear system with $S_1, \ldots, S_{n-1}$ as unknowns. Once $S_1, \ldots, S_{n-1}$ are known, a_i, b_i, c_i and d_i, $i = 1, 2, \ldots, n$ are obtained by means of (14), (13), (15) and (9).*
Up to now we have not taken in account the choice of boundary conditions. The following question arises: Having chosen a set of boundary conditions, how does this choice impact on (17)? We answer this question with respect to the boundary conditions given in (f) of the definition above.
$S_0 = S_n = 0$, so for $i = 1$ and $i = n-1$, (17) becomes

(18) $$(2h_0 + 2h_1)S_1 + h_1 S_2 = 6(f[x_1, x_2] - f[x_0, x_1])$$

and

(19) $$h_{n-2}S_{n-2} + (2h_{n-2} + 2h_{n-1})S_{n-1} = 6(f[x_{n-1}, x_n] - f[x_{n-2}, x_{n-1}])$$

respectively.

Application to the points in the question

Four evenly spaced points are given, as well as their images via f, where the cubic spline will match $f(x)$. Thus, if n is the number of subintervals determined by the given data points, then $n = 3$. The required cubic spline will be in the form

$$S(x) = \begin{cases} S_0(x) &= a_0 + b_0(x - x_0) + c_0(x - x_0)^2 + d_0(x - x_0)^3 & x \in [x_0, x_1] \\ S_1(x) &= a_1 + b_1(x - x_1) + c_1(x - x_1)^2 + d_1(x - x_1)^3 & x \in [x_1, x_2] \\ S_2(x) &= a_2 + b_2(x - x_2) + c_2(x - x_2)^2 + d_2(x - x_2)^3 & x \in [x_2, x_3] \end{cases}$$

Let $h_i = x_{i+1} - x_i$ for each i such that $0 \leq i \leq n - 1 = 2$ (see [1, pp. 144-160]) where the x_i are given data and $n = 3$. According to the theorem in [1, p. 149], the coefficient matrix is a square matrix, of size 4×4, and tridiagonal. The expressions for calculating each coefficient of the matrix is also known.

In this case, $h_0 = x_1 - x_0 = 0.5$; similarly, $h_1 = h_2 = 0.5$; also, $a_0 = 15$, $a_1 = 9$, $a_2 = 3$, $a_3 = 1$. The problem reduces to solving the system $Ax = b$ where

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ h_0 & 2(h_0 + h_1) & h_1 & 0 \\ 0 & h_1 & 2(h_1 + h_2) & h_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \frac{3}{h_2}(a_3 - a_2) - \frac{3}{h_1}(a_2 - a_1) \\ 0 \end{bmatrix} \quad x = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

and this simplifies to

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 2 & 0.5 & 0 \\ 0 & 0.5 & 2 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} 0 \\ 0 \\ 24 \\ 0 \end{bmatrix} \qquad x = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

which yields

$$c_0 = 0 \qquad c_1 = -\frac{16}{5} \qquad c_2 = \frac{64}{5} \qquad c_3 = 0$$

Solving for b_0, b_1, b_2, then d_0, d_1, d_2 yields

$$b_0 = -172/15; \qquad b_1 = -196/15; \qquad b_2 = -124/15;$$

and

$$d_0 = -32/15; \qquad d_1 = 32/3; \qquad d_2 = -128/15.$$

Hence the required cubic spline is defined as follows:

$$S(x) = \begin{cases} S_0(x) &= 15 - \frac{172}{15}(x - 0) + 0(x - 0)^2 - \frac{32}{15}(x - 0)^3 & x \in [0, 0.5] \\ S_1(x) &= 9 - \frac{196}{15}(x - 0.5) - \frac{16}{5}(x - 0.5)^2 + \frac{32}{3}(x - 0.5)^3 & x \in [0.5, 1] \\ S_2(x) &= 3 - \frac{124}{15}(x - 1) + \frac{64}{5}(x - 1)^2 - \frac{128}{15}(x - 1)^3 & x \in [1, 1.5] \end{cases}$$

Question 5

In order to construct and graph the required cubic Bezier polynomials, we follow the procedure of [1, pp. 165-169].

(5.1)

The given points are $\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ *with guide-point* $p_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$;
$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$ *with guide-point* $p_1 = \begin{bmatrix} 6 \\ 1 \end{bmatrix}$.
In parametric form the set of points on this cubic Bezier curve is

$$P(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}, \quad 0 \leq t \leq 1.$$

where $x(0) = x_0$, $y(0) = y_0$, $x(1) = x_1$, *and* $y(1) = y_1$.
The cubic Bezier polynomial is given by

$$\begin{aligned} x(t) &= [2(x_0 - x_1) + 3(\alpha_0 + \alpha_1)]t^3 + [3(x_1 - x_0) - 3(\alpha_1 + 2\alpha_0)]t^2 + 3\alpha_0 t + x_0 \\ y(t) &= [2(y_0 - y_1) + 3(\beta_0 + \beta_1)]t^3 + [3(y_1 - y_0) - 3(\beta_1 + 2\beta_0)]t^2 + 3\beta_0 t + y_0 \end{aligned}$$

Using the guide-points, the values of the coefficients are $\alpha_0 = 1$, $\beta_0 = 1$, $\alpha_1 = -1$, *and* $\beta_1 = 1$.
So, the constructed Bezier polynomial, as required, is

$$\begin{aligned} x(t) &= -10t^3 + 12t^2 + 3t; \\ y(t) &= 2t^3 - 3t^2 + 3t. \end{aligned}$$

(5.2)

The endpoints remain unchanged, but with the guide-points

$$p_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \quad \textit{and} \quad p_1 = \begin{bmatrix} 5.5 \\ 1.5 \end{bmatrix}.$$

In the same way as in (5.1) above, we obtain the values of the coefficients as; $\alpha_0 = 1/2$, $\beta_0 = 1/2$, $\alpha_1 = -1/2$, *and* $\beta_1 = 1/2$ *and the constructed Bezier polynomial, as required, is*

$$\begin{aligned} x(t) &= -10t^3 + \tfrac{27}{2}t^2 + \tfrac{3}{2}t; \\ y(t) &= -t^3 + \tfrac{3}{2}t^2 + \tfrac{3}{2}t. \end{aligned}$$

(5.3)

Again, the endpoints remain unchanged, but with the guide-points

$$p_0 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \quad \textit{and} \quad p_1 = \begin{bmatrix} 6 \\ 3 \end{bmatrix}.$$

In the same way as in (5.1) above, we obtain the values of the coefficients as; $\alpha_0 = 2$, $\beta_0 = 2$, $\alpha_1 = -1$, *and* $\beta_1 = -1$ *and the constructed Bezier polynomial, as required, is*

$$\begin{aligned} x(t) &= -7t^3 + 6t^2 + 6t; \\ y(t) &= -t^3 - 3t^2 + 6t. \end{aligned}$$

The graphs of the cubic Bezier curves for all three cases are given in Figure 1, where the output from question (5.1) is in red (bottom curve from the right), that from question (5.2) in green (middle curve from the right), and that from question (5.3) in blue (top curve from the right).

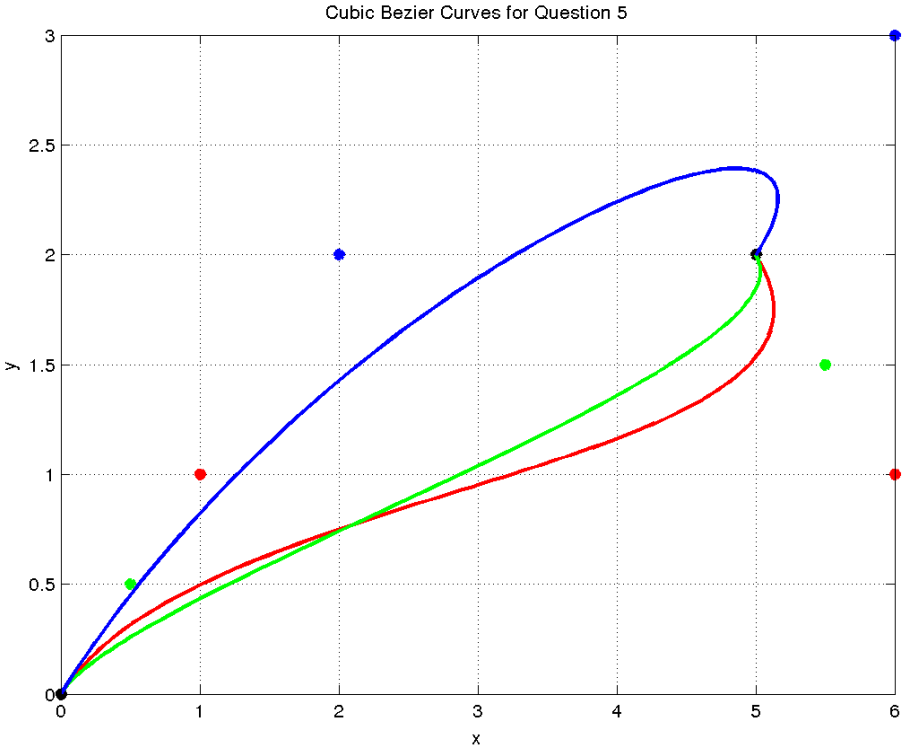

Figure 1: Cubic Bezier Curves for Question 5

Question 6

(6.1)

It is clear from the data that $m = 8$. *Note that* $\sum$ *denotes* $\sum_{i=1}^{m}$ *and in particular,* $\sum_{i=1}^{8}$. *We assume the functional relationship*

$$y = a_0 + a_1 x + a_2 x^2 + a_3 x^3,$$

and proceed as in [1, pp. 498-506]. This leads to the four *normal equations to be solved for* a_0, a_1, a_2, *and* a_3.

(20)
$$\begin{aligned} 8a_0 + a_1 \sum x_i + a_2 \sum x_i^2 + a_3 \sum x_i^3 &= \sum Y_i \\ a_0 \sum x_i + a_1 \sum x_i^2 + a_2 \sum x_i^3 + a_3 \sum x_i^3 &= \sum x_i Y_i \\ a_0 \sum x_i^2 + a_1 \sum x_i^3 + a_2 \sum x_i^4 + a_3 \sum x_i^5 &= \sum x_i^2 Y_i \\ a_0 \sum x_i^3 + a_1 \sum x_i^4 + a_2 \sum x_i^5 + a_3 \sum x_i^6 &= \sum x_i^3 Y_i \end{aligned}$$

From our given data we obtain

(21)
$$\begin{array}{lll} \sum x_i = 7.400 & \sum x_i^2 = 8.7200 & \sum x_i^3 = 11.3480 \\ \sum x_i^4 = 15.5108 & \sum x_i^5 = 21.8584 & \sum x_i^6 = 31.4840 \\ \sum Y_i = 8.2253 & \sum x_i Y_i = 10.7313 & \sum x_i^2 Y_i = 14.7269 \\ & & \sum x_i^3 Y_i = 20.8326 \end{array}$$

Solving the linear system, we obtain

$$\begin{aligned} a_0 &= -0.0181 \\ a_1 &= 0.2469 \\ a_2 &= 0.4047 \\ a_3 &= 0.2656 \end{aligned}$$

Our approximating function is therefore

$$y = -0.0181 + 0.2469x + 0.4047x^2 + 0.2656x^3.$$

We summarize the results as follows:

i	x_i	Y_i	y_i	$Y_i - y_i$
1	0.2	0.0504	0.0496	0.0009
2	0.3	0.0984	0.0996	−0.0011
3	0.6	0.3328	0.3331	−0.0003
4	0.9	0.7266	0.7255	0.0011
5	1.1	1.0972	1.0967	0.0005
6	1.3	1.5697	1.5703	−0.0006
7	1.4	1.8487	1.8496	−0.0009
8	1.6	2.5015	2.5009	−0.0006

The sum of square error, S, is 5.1×10^{-6}.

(6.2)

We use the same data as in question 6.1.
We assume the functional relationship

$$y = be^{ax}.$$

Linearizing the exponential form, we obtain

$$\ln y = \ln b + ax,$$

and fit the new variable $z = \ln y$ as a linear function of x, as before.
We now assume the functional relationship

$$z = a_0 + a_1 x,$$

and proceed as before. This leads to the two *normal equations to be solved for a_0 and a_1.*

(22)
$$\begin{aligned} m a_0 + a_1 \sum x_i &= \sum \ln Y_i \\ a_0 \sum x_i + a_1 \sum x_i^2 &= \sum x_i \ln Y_i. \end{aligned}$$

From our given data we obtain

(23)
$$\begin{array}{ll} \sum x_i = 7.4000 & \sum x_i^2 = 8.7200 \\ \sum \ln Y_i = -4.6500 & \sum x_i \ln Y_i = 0.7750 \end{array}$$

So, by (23), (22) becomes

(24)
$$\begin{aligned} 8a_0 + 7.4a_1 &= -4.65 \\ 7.4a_0 + 8.72a_1 &= 0.7750. \end{aligned}$$

Solving (24) as before, we obtain

$$\begin{aligned} a_0 &= -3.0855 \\ a_1 &= 2.7073 \end{aligned}$$

so that

$$b = e^{a_0} = 0.0457.$$

Our approximating function is therefore

$$y = 0.0457e^{2.7073x}.$$

We summarize the results as follows:

i	x_i	Y_i	y_i	$Y_i - y_i$
1	0.2	0.0504	0.0526	−0.0022
2	0.3	0.0984	0.0565	0.0419
3	0.6	0.3328	0.0699	0.2629
4	0.9	0.7266	0.0864	0.6402
5	1.1	1.0972	0.0995	0.9977
6	1.3	1.5697	0.1146	1.4551
7	1.4	1.8487	0.123	1.7257
8	1.6	2.5015	0.1417	2.3598

The sum of square errors, S, is 12.1402

(6.3)

We use the same data as in question 6.1.
We assume the functional relationship

$$y = bx^a.$$

Linearizing the exponential form, we obtain

$$\ln y = \ln b + a \ln x,$$

and fit the new variable $z = \ln y$ as a linear function of x, as before.
We now assume the functional relationship

$$z = a_0 + a_1 \ln x,$$

and proceed as before. This leads to the two *normal equations to be solved for* a_0 *and* a_1.

(25)
$$\begin{aligned} ma_0 + a_1 \sum \ln x_i &= \sum \ln Y_i \\ a_0 \sum \ln x_i + a_1 \sum (\ln x_i)^2 &= \sum (\ln x_i) \ln Y_i. \end{aligned}$$

From our given data we obtain

(26)
$$\begin{array}{ll} \sum \ln x_i = -2.26544654 & \sum (\ln x_i)^2 = 4.7239205788 \\ \sum \ln Y_i = -4.64996574 & \sum (\ln x_i) \ln Y_i = 8.959052285 \end{array}$$

So, by (26), (25) becomes

(27)
$$\begin{aligned} 8a_0 - 2.26544654a_1 &= -4.64996574 \\ -2.26544654a_0 + 4.7239205788a_1 &= 8.959052285. \end{aligned}$$

Solving (27) as before, we obtain

$$\begin{aligned} a_0 &= -0.0511285981 \\ a_1 &= 1.872009283, \end{aligned}$$

so that

$$b = e^{a_0} = 0.9501564743.$$

Our approximating function is therefore

$$y = 0.9501564743x^{1.872009283}.$$

We summarize the results as follows:

i	x_i	Y_i	y_i	$Y_i - y_i$
1	0.2	0.0504	0.0466999	0.003746009
2	0.3	0.0984	0.0997611	−0.001335108
3	0.6	0.3328	0.3651675	−0.032397579
4	0.9	0.7266	0.7800755	−0.053475583
5	1.1	1.0972	1.1357496	−0.038549679
6	1.3	1.5697	1.5527378	0.016962129
7	1.4	1.8487	1.7838082	0.064891736
8	1.6	2.5015	2.2903910	0.211108968

The sum of square errors, S, *is* 0.0545

(6.4)

The graph of the approximating function in (6.1), (6.2) and (6.3), as well as the given data points, are shown in Figure 2, where the output from question (6.1) is in red (middle curve from the right), that from question (6.2) in green (top curve from the right), and that from question (6.3) in blue (bottom curve from the right). Note that the approximation from question (6.1) is the closest to the given data.

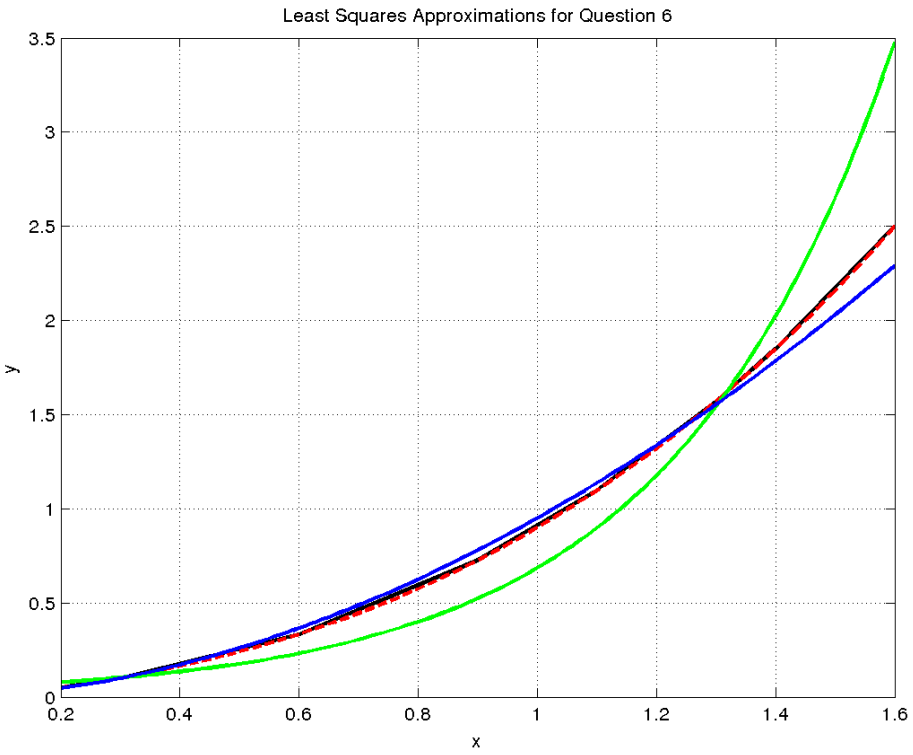

Figure 2: Least Squares Approximations for Question 6

Question 7

(7.1) We need to use different numerical techniques to approximate the integral

$$\int_{0.75}^{1.3} \left(\sin^2 x - 2x \sin x + 1\right)\ dx$$

(a) Composite trapezoidal rule with $n = 8$.

Here, we have $h = \frac{1.3-0.75}{8} = 0.06875$. *Therefore the interval* $[0.75, 3]$ *is divided into* 8 *subintervals with nodes* $x_0, x_1, \ldots, x_8$ *where* $x_0 = 0.75,\ x_8 = 1.3,\ x_i = x_0 + i \cdot h$. *Also,* $f_i = f(x_i)$ *for all* i, *where* $f(x) = \sin^2 x - 2x \sin x + 1$. *Hence the approximation of the integral is*

$$I \approx \frac{h}{2}\left(f_0 + 2f_1 + 2f_2 + 2f_3 + 2f_4 + 2f_5 + 2f_6 + 2f_7 + f_8\right)$$

Note that the values of x *are in* ***radians****. If you wish to convert them to degrees, you may do so by using the formula*

$$x_{degrees} = \frac{180}{\pi} x_{radians}$$

The values of $f(x_i)$ *at* $x_i,\ i = 0, 1, \ldots, 8$ *are given in the table below.*

i	x_i *(in radians)*	x_i *(in degrees)*	$f(x_i)$
0	0.75	42.95454545	$f_0 = 0.442173259$
1	0.81875	46.89204545	$f_1 = 0.33747317$
2	0.8875	50.82954545	$f_2 = 0.224888693$
3	0.95625	54.76704545	$f_3 = 0.104966722$
4	1.025	58.70454545	$f_4 = -0.021627743$
5	1.09375	62.64204545	$f_5 = -0.154101955$
6	1.1625	66.57954545	$f_6 = -0.291527471$
7	1.23125	70.51704545	$f_7 = -0.432834221$
8	1.3	74.45454545	$f_8 = -0.576806905$

And finally the approximation of the integral is $I \approx -0.020630474$

(b) Gaussian quadrature

The Gaussian quadrature process in general is described in [1, pp.228-234]. The three-term Gaussian quadrature method is implemented in two steps: Transform our integral from 0.75 *to* 1.3 *into an integral from* -1 *to* 1 *by a linear change of variable; then use the roots and corresponding coefficients of the* third Legendre polynomial *(see Table 4.12 [1, p.232]) to determine the required approximation. Note that the three-term Gaussian quadrature gives exact results for polynomials of degree* $2 \times 3 - 1 = 5$ *or less.*

- *To make the transformation*

$$\int_a^b f(x)dx \quad \longrightarrow \quad \int_{-1}^1 g(t)dt$$

we let $t = mx + p$ *and we find* m *and* p *such that when* $x = a$*, we have* $t = -1$ *and when* $x = b$*, we have* $t = 1$*. Thus we obtain the following system to be solved for* m *and* p

$$\begin{cases} ma + p &= -1 \\ mb + p &= 1 \end{cases} \quad \Longrightarrow \quad \begin{cases} m &= \dfrac{2}{b-a} \\ p &= \dfrac{a+b}{a-b} \end{cases}$$

Therefore

$$t = \frac{2}{b-a}x + \frac{a+b}{a-b} \quad \Longleftrightarrow \quad x = \frac{1}{2}\left[(b-a)t + a + b\right]$$

and hence (since this means $dx = \dfrac{b-a}{2}dt$*),*

$$\int_a^b f(x)dx = \int_{-1}^1 f\left(\frac{(b-a)t + (b+a)}{2}\right)\left(\frac{b-a}{2}\right)dt$$

as in equation (4.41) in [1, p.233]. Hence

$$g(t) = \frac{b-a}{2} f\left(\frac{(b-a)t + (b+a)}{2}\right)$$

- *Since we are implementing a three-term Gaussian quadrature, our nodes (x_i) are the roots of the third Legendre polynomial which is*

$$P_3(t) = t^3 - \frac{3}{5}t = t\left(t^2 - \frac{3}{5}\right)$$

The roots of this polynomial are

$$t_0 = -\sqrt{\frac{3}{5}} \approx -0.774597 \qquad t_1 = 0 \qquad t_2 = \sqrt{\frac{3}{5}} \approx 0.774597$$

and the corresponding coefficients (from Table 4.12 in [1, p.232]) are

$$c_0 = \frac{5}{9} \quad c_1 = \frac{8}{9} \quad c_2 = \frac{5}{9}$$

Since $a = 0.75$ and $b = 1.3$, we obtain

$$g(t) = \frac{0.55}{2} f\left(\frac{0.55t + 2.05}{2}\right) \quad \text{where} \quad f(x) = \sin^2 x - 2x \sin x + 1$$

And the required approximation is

$$\begin{aligned}\int_0^1 \frac{\sin x}{x} dx &\approx c_0 g(t_0) + c_1 g(t_1) + c_2 g(t_2) \\ &= \frac{5}{9} g\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9} g(0) + \frac{5}{9} g\left(\sqrt{\frac{3}{5}}\right) \\ &\approx -0.0203766519\end{aligned}$$

(c) Simpson's $\frac{3}{8}$ rule

The formula for the Simpson's $\frac{3}{8}$ rule is given in Equation (4.27) in [1, p.199]. To approximate $\int_a^b f(x)dx$, we divide $[a, b]$ into three intervals of equal length. Hence we get $x_0 = a$, $x_1 = a + h$, $x_2 = a + 2h$ and $x_3 = b$, where $h = \frac{b-a}{3}$. Note that we can also use the composite Simpson's rule. This would imply dividing the interval $[a, b]$ into a number of subintervals that is a multiple of three, and applying Simpson's rule on every three consecutive subintervals, starting from a.
In our case, we are applying the normal Simpson's rule. Here, $a = 0.75 = x_0$, $b = 1.3 = x_3$, $x_1 = 0.75 + \frac{0.55}{3} = \frac{2.8}{3}$, $x_2 = 0.75 + \frac{1.1}{3} = \frac{3.35}{3}$, where $h = \frac{b-a}{3} = \frac{0.55}{3}$, and the formula for the Simpson's rule is

$$\begin{aligned}\int_{x_0}^{x_3} f(x)dx &\approx \frac{3h}{8}\left[f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)\right] \\ &= \frac{0.55}{8}\left[f(0.75) + 3f(2.8/3) + 3f(3.35/3) + f(1.3)\right] \\ &\approx -0.0203301860\end{aligned}$$

(7.2) Estimating truncation errors

- *In (7.1(a)): For the composite trapezoidal rule, from Theorem 4.5 in [1, p.206], the truncation error is given by*

$$\frac{b-a}{12}h^2|f''(\xi)| \quad \textit{for} \quad 0.75 < \xi < 1.3$$

and with our values of a, b and h, the truncation error is (for $0.75 < \xi < 1.3$)

$$\frac{1.3-0.75}{12} \times \left(\frac{0.55}{8}\right)^2 |f''(\xi)| = \frac{13}{60009}|f''(\xi)|$$

We can go further and find a maximum value for $|f''(\xi)|$ to have a bound on the error. Indeed, from $f(x) = \sin^2 x - 2x \sin x + 1$, it follows that

$$f'(x) = \sin 2x - 2\sin x - 2x\cos x \quad \textit{and} \quad f''(x) = 2\cos 2x - 4\cos x + 2x \sin x$$

A rough sketch of the graph of f'' indicates that f'' is increasing on $[0.75, 1.3]$; note that $f''(a) \approx -1.7628$ and $f''(b) \approx -0.2785$; therefore the maximum value of the error is

$$\frac{13}{60009}|f''(a)| \approx 3.8189 \times 10^{-4}$$

- *In (7.1(c)): For Simpson's rule, from Equation (4.27) in [1, p.199], the truncation error is given by*

$$\frac{3h^5}{80}|f^{(4)}(\xi)| \quad \textit{for} \quad 0.75 < \xi < 1.3$$

and with our values of a, b and h, the truncation error is (for $0.75 < \xi < 1.3$)

$$\frac{3(0.55/8)^5}{80}|f^{(4)}(\xi)| = \frac{1}{17362219}|f^{(4)}(\xi)|$$

Again, we can go further and find a maximum value for $|f^{(4)}(\xi)|$ to have a bound on the error.

(7.3) We need to give the analytical solution (exact value) of the integral

$$\int_{0.75}^{1.3} \left(\sin^2 x - 2x \sin x + 1\right)\ dx$$

Note that Taylor expansion of $\sin x$ will just lead to an approximation of the integral while a direct approach with trigonometric transformations and integration by parts leads to the exact value of the integral.

First note that $\sin^2 x = \frac{1}{2}(1 - \cos 2x)$. *Also note that the values of* x *are in* ***radians****. Then our calculation follow:*

$$\begin{aligned}
&\int_{0.75}^{1.3} \left(\sin^2 x - 2x \sin x + 1\right)\ dx \\
&= \int_{0.75}^{1.3} \left(\frac{1}{2} - \frac{1}{2}\cos(2x) - 2x \sin x + 1\right)\ dx \\
&= \left[\frac{3}{2}x - \frac{1}{4}\sin(2x)\right]_{0.75}^{1.3} - 2\int_{0.75}^{1.3} x \sin x\ dx \qquad (\textit{then we use integration by parts}) \\
&= \left[\frac{3}{2}x - \frac{1}{4}\sin(2x)\right]_{0.75}^{1.3} - 2\left([-x\cos x]_{0.75}^{1.3} + \int_{0.75}^{1.3} \cos x\ dx\right) \\
&= \left[\frac{3}{2}x - \frac{1}{4}\sin(2x)\right]_{0.75}^{1.3} - 2\left([-x\cos x + \sin x]_{0.75}^{1.3}\right) \\
&= \left[\frac{3}{2}x - \frac{1}{4}\sin(2x) + 2x\cos x - 2\sin x\right]_{0.75}^{1.3} \\
&\approx -0.0203767960
\end{aligned}$$

The actual errors are as follows:

- *In (7.1(a)):* $E = |-0.0203767960 - (-0.020630474)| \approx 2.53678 \times 10^{-4}$
- *In (7.1(b)):* $E = |-0.0203767960 - (-0.0203766519)| \approx 1.4410 \times 10^{-7}$
- *In (7.1(c)):* $E = |-0.0203767960 - (-0.0203301860)| \approx 4.6610 \times 10^{-5}$

Question 8

Consider the integral

$$\int\int_{\mathcal{R}} \sqrt{1 + [f_x(x,y)]^2 + [f_y(x,y)]^2}\ dA$$

where, $z = f(x,y)$, $x^2 + y^2 + z^2 = 9$; $z \geq 0$ and $\mathcal{R} = \{(x,y) | 0 \leq x \leq 1,\ 0 \leq y \leq 1\}$. We have

$$x^2 + y^2 + z^2 = 9 \iff z^2 = 9 - x^2 - y^2 \iff z = \pm\sqrt{9 - x^2 - y^2}$$

But $z = f(x,y)$ and $z \geq 0$, hence $f(x,y) = \sqrt{9 - x^2 - y^2}$ therefore f_x and f_y (note that f_x means $\frac{\partial f}{\partial x}$ and f_y means $\frac{\partial f}{\partial y}$) are defined as follows:

$$f_x(x,y) = \frac{-x}{\sqrt{9 - x^2 - y^2}} \quad \text{and} \quad f_y(x,y) = \frac{-y}{\sqrt{9 - x^2 - y^2}}$$

Thus

$$1 + [f_x(x,y)]^2 + [f_y(x,y)]^2 = \frac{9}{9 - x^2 - y^2}$$

Our integral then becomes

$$\int_0^1 \int_0^1 \sqrt{1 + [f_x(x,y)]^2 + [f_y(x,y)]^2}\, dA = 3 \int_0^1 \int_0^1 \frac{1}{\sqrt{9 - x^2 - y^2}}\, dx\, dy$$

(8.1) Trapezoidal rule in both directions

The trapezoidal rule is explained in [1, p.194, 195, 206, 207, 235, 236] for standard and composite options, for single and double integrals.
Consider x as a constant and let $y_j = c + jk,\ j = 0, 1, 2, \ldots, m$, and $k = (d-c)/m$. By the Trapezoidal rule, we have

$$\int_c^d f(x,y)dy \approx \frac{k}{2}\left[f(x,y_0) + 2\sum_{j=1}^{m-1} f(x,y_i) + f(x,y_m)\right].$$

Therefore,

$$\int_a^b \int_c^d f(x,y)dydx \approx \frac{k}{2}\int_a^b f(x,y_0)\,dx + k\sum_{j=1}^{m-1}\int_a^b f(x,y_i)\,dx + \frac{k}{2}\int_a^b f(x,y_m)\,dx. \quad (28)$$

Now let $x_i = a + ih,\ i = 0, 1, 2, \ldots, n$ and $h = (b-a)/n$. Then, by approximating the single integrals in the right hand side of equation (28) with the Trapezoidal rule, the double integral may be approximated by

$$\begin{aligned}
&\frac{kh}{4}\left[f(x_0,y_0) + 2\sum_{i=1}^{n-1} f(x_i,y_0) + f(x_n,y_0)\right] \\
&+\frac{kh}{2}\sum_{j=1}^{m-1}\left[f(x_0,y_j) + 2\sum_{i=1}^{n-1} f(x_i,y_j) + f(x_n,y_j)\right] \\
&+\frac{kh}{4}\left[f(x_0,y_m) + 2\sum_{i=1}^{n-1} f(x_i,y_m) + f(x_n,y_m)\right].
\end{aligned}$$

Implementing the latter approximation with $n = 2$ and $m = 2$ (i.e. $h = 0.5$ and $k = 0.5$) yields

$$3\int_0^1 \int_0^1 \frac{1}{\sqrt{9 - x^2 - y^2}}\, dx\, dy \approx 1.046152967.$$

***Note**: As an exercise, find out the approximation to the given double integral when $n = m = 1$.*

(8.2) Simpson's $\frac{1}{3}$ rule in both directions

The Simpson $\frac{1}{3}$ rule is explained in [1, p.195, 196, 205, 206, 237] for standard and composite options, for single and double integrals.
Consider x as a constant and let $y_j = c + jk,\ j = 0, 1, 2, \ldots, m$, and $k = (d-c)/m$, where m is an even integer. By Simpson's $\frac{1}{3}$ rule, we have:

$$\int_c^d f(x,y)dy \approx \frac{k}{3}\left[f\left(x,y_0\right) + 2\sum_{j=1}^{m/2-1} f\left(x,y_{2j}\right) + 4\sum_{j=1}^{m/2} f\left(x,y_{2j-1}\right) + f\left(x,y_m\right)\right].$$

Therefore,

$$\begin{aligned}\int_a^b\int_c^d f(x,y)dydx \approx \frac{k}{3}\int_a^b f\left(x,y_0\right)dx + \frac{2k}{3}\sum_{j=1}^{m/2-1}\int_a^b f\left(x,y_{2j}\right)dx \\ + \frac{4k}{3}\sum_{j=1}^{m/2}\int_a^b f\left(x,y_{2j-1}\right)dx + \frac{k}{3}\int_a^b f\left(x,y_m\right)dx.\end{aligned} \tag{29}$$

Now let $x_i = a + ih,\ i = 0, 1, 2, \ldots, n$ and $h = (b-a)/n$, where n is also an even integer. Then, by approximating the single integrals in the right hand side of equation (29) with Simpson's $\frac{1}{3}$ rule, the double integral may be approximated by

$$\begin{aligned}&\frac{kh}{9}\left[f\left(x_0,y_0\right) + 2\sum_{i=1}^{n/2-1} f\left(x_{2i},y_0\right) + 4\sum_{i=1}^{n/2} f\left(x_{2i-1},y_0\right) + f\left(x_n,y_0\right)\right] \\ &+\frac{2kh}{9}\sum_{j=1}^{m/2-1}\left[f\left(x_0,y_{2j}\right) + 2\sum_{i=1}^{n/2-1} f\left(x_{2i},y_{2j}\right) + 4\sum_{i=1}^{n/2} f\left(x_{2i-1},y_{2j}\right) + f\left(x_n,y_{2j}\right)\right] \\ &+\frac{4kh}{9}\sum_{j=1}^{m/2}\left[f\left(x_0,y_{2j-1}\right) + 2\sum_{i=1}^{n/2-1} f\left(x_{2i},y_{2j-1}\right) + 4\sum_{i=1}^{n/2} f\left(x_{2i-1},y_{2j-1}\right) + f\left(x_n,y_{2j-1}\right)\right] \\ &+\frac{kh}{9}\left[f\left(x_0,y_m\right) + 2\sum_{i=1}^{n/2-1} f\left(x_{2i},y_m\right) + 4\sum_{i=1}^{n/2} f\left(x_{2i-1},y_m\right) + f\left(x_n,y_m\right)\right].\end{aligned}$$

Implementing the latter approximation with $n = 2$ and $m = 2$ (i.e. $h = 0.5$ and $k = 0.5$) yields

$$3\int_0^1\int_0^1 \frac{1}{\sqrt{9-x^2-y^2}}\,dx\,dy \approx 1.0403781466.$$

(8.3) Three-term Gaussian quadrature in both directions

The Gaussian Quadrature rule is explained in [1, p.228-234, 240] for single and double integrals.
In order to apply Gaussian quadrature we need to transform the intervals of integration (a,b) and (c,d) both to $(-1,1)$. We know from our textbook that

$$\int_{-1}^1 f(x)dx \approx \sum_{i=1}^n w_i f\left(t_i\right)$$

where w_i and t_i are the Gaussian weights and points respectively. Let us now consider the integral

$$\int_c^d f(x,y)dy.$$

By letting

$$y = \frac{(d-c)t + (d+c)}{2},$$

so that $dy = \left(\frac{d-c}{2}\right) dt$, *it follows that*

$$\int_c^d f(x,y)dy = \left(\frac{d-c}{2}\right) \int_{-1}^1 f\left(x, \frac{(d-c)t + (d+c)}{2}\right) dt.$$

Similarly, by letting

$$x = \frac{(b-a)s + (b+a)}{2},$$

so that $dx = \left(\frac{b-a}{2}\right) ds$, *it follows that*

$$\int_a^b \int_c^d f(x,y)dydx = \left(\frac{b-a}{2}\right)\left(\frac{d-c}{2}\right) \int_{-1}^1 \int_{-1}^1 f\left(\frac{(b-a)s + (b+a)}{2}, \frac{(d-c)t + (d+c)}{2}\right) dtds.$$

The double integral

$$\int_{-1}^1 \int_{-1}^1 f\left(\frac{(b-a)s + (b+a)}{2}, \frac{(d-c)t + (d+c)}{2}\right) dtds$$

may now be approximated by an $n-$term Gaussian quadrature formula

$$G_n = \sum_{i=1}^n w_i \sum_{j=1}^n w_j f\left(\frac{(b-a)p_j + (b+a)}{2}, \frac{(d-c)p_i + (d+c)}{2}\right).$$

So,

$$\int_a^b \int_c^d f(x,y)dydx = \left(\frac{b-a}{2}\right)\left(\frac{d-c}{2}\right) G_n.$$

By implementing this approximation with $n = 3$, since we have $a = c = 0$ and $b = d = 1$, it follows that

$$G_3 = \sum_{i=1}^3 w_i \sum_{j=1}^3 w_j f\left(\frac{p_j + 1}{2}, \frac{p_i + 1}{2}\right)$$

where

$$p_1 = -\sqrt{\frac{3}{5}}, \quad p_2 = 0, \quad p_3 = \sqrt{\frac{3}{5}} \qquad \text{and} \qquad w_1 = \frac{5}{9}, \quad w_2 = \frac{8}{9}, \quad w_3 = \frac{5}{9}$$

Then we get

$$3\int_0^1 \int_0^1 \frac{1}{\sqrt{9 - x^2 - y^2}}\, dx\, dy \approx \frac{1}{2} \times \frac{1}{2} \times 4.161006249325179 = 1.040251562331295.$$

REFERENCES

[1] Richard L. Burden and Douglas J. Faires. *Numerical Analysis*. ninth edition. BROOKS/COLE, CENGAGE learning 2011.