Computational Analysis of Variation in Temperature T in a 1-D Viscous Fluid Flow

1 Problem Statement

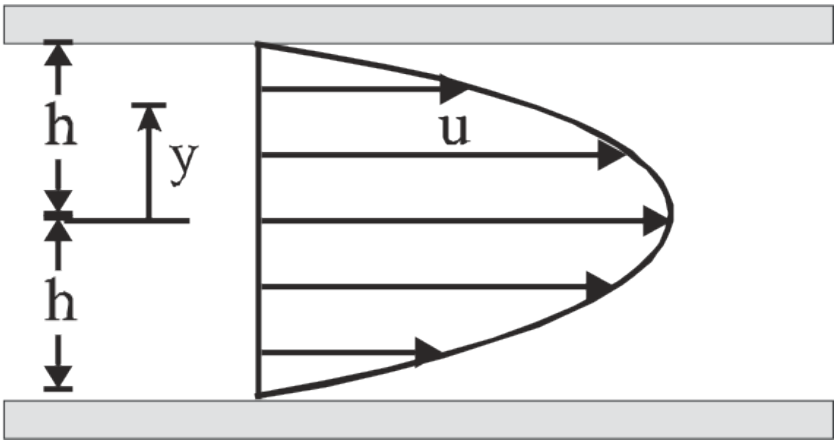

Figure 1: Viscous Fluid Flow between Parallel Plates

The equation governing the variation in temperature T in a viscous fluid flowing between two parallel-plates ($y = 0$ and $y = 2H$) is given to be,

$$\frac{d^2T}{dy^2} = -\frac{4U^2\mu}{H^4\kappa}(H - y)^2$$

where μ, κ and U are the viscosity, thermal conductivity and maximum velocity of the fluid respectively.

2 Boundary Value Problems

Some simple examples of *two-point* **linear** boundary value problems are:

$$y''(x) + f(x)y'(x) + g(x)y(x) = r(x)$$

with the boundary conditions

$$y(x_0) = a \quad \text{and} \quad y(x_n) = b$$
$$y^{\text{iv}}(x) = p(x)y(x) + q(x)$$

with

$$y(x_0) = y'(x_0) = A \quad \text{and} \quad y(x_n) = y'(x_n) = B$$

where $y'(x)$ is the first derivative of $y(x)$, $y''(x)$ is the second derivative of $y(x)$, $y'''(x)$ is the third derivative of $y(x)$ and $y^{\text{iv}}(x)$ is the fourth derivative of $y(x)$.

3 Finite Difference Method

The Finite-Difference Method consists in replacing the derivatives occurring in the differential equation by means of their Finite-Difference approximations and solving the large system of linear equations using Row-Echelon form of matrix.

To obtain approximations to the derivatives, we proceed as follows: Expanding $y(x+h)$ in Taylor's Series,

$$y(x+h) = y(x) + hy'(x) + \frac{h^2}{2}y''(x) + \frac{h^3}{6}y'''(x) + \cdots$$

form which we obtain

$$y'(x) = \frac{y(x+h) - y(x)}{h} - \frac{h}{2}y''(x) - \cdots$$

Thus we have,

$$y'(x) = \frac{y(x+h) - y(x)}{h} + \mathrm{O}(h)$$

which is the *forward-difference approximation* for $y'(x)$. Similarly, the expansion for $y(x-h)$ in Taylor's Series gives

$$y(x-h) = y(x) - hy'(x) + \frac{h^2}{2}y''(x) - \frac{h^3}{6}y'''(x) + \cdots$$

from which we obtain

$$y'(x) = \frac{y(x) - y(x-h)}{h} + \mathrm{O}(h)$$

which is the *backward-difference approximation* for $y'(x)$.

A *Central difference approximation* for $y'(x)$ can be obtained by subtracting Taylor Series equations for $y(x+h)$ and $y(x-h)$. We thus have,

$$y'(x) = \frac{y(x+h) - y(x-h)}{2h} + \mathrm{O}\left(h^2\right)$$

It is clear that the above approximation for first derivative is **better** than the forward-difference and backward-difference approximations. Adding Taylor Series equations for $y(x+h)$ and $y(x-h)$, we get an approximation for $y''(x)$:

$$y''(x) = \frac{y(x-h) - 2y(x) + y(x+h)}{h^2} + \mathrm{O}\left(h^2\right)$$

In a similar manner, it is possible to derive finite-difference approximations to higher derivatives. To solve the boundary value problems defined in the above section, we divide the range $[x_0, x_n]$ into n equal sub-intervals of width h so that,

$$x_i = x_0 + ih, \quad i = 1, 2, \ldots, n$$

The corresponding values of y at these points are denoted by,

$$y(x_i) = y_i = y(x_0 + ih), \quad i = 0, 1, 2, \ldots, n$$

From the central-difference approximations corresponding to $y'(x)$ and $y''(x)$, the values for $y'(x)$ and $y''(x)$ at the point $x = x_i$ can now be written as

$$y'_i = \frac{y_{i+1} - y_{i-1}}{2h} + \mathrm{O}\left(h^2\right)$$

and

$$y''_i = \frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} + \mathrm{O}\left(h^2\right)$$

Satisfying the differential equation at the point $x = x_i$, we get

$$y''_i + f_i y'_i + g_i y_i = r_i$$

Substituting the expressions for y'_i and y''_i, this gives

$$\frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} + f_i \frac{y_{i+1} - y_{i-1}}{2h} + g_i y_i = r_i, \quad i = 1, 2, \ldots, n-1$$

where $y_i = y(x_i), g_i = g(x_i)$, etc.
Multiplying through by h^2 and simplifying, we obtain

$$\left(1 - \frac{h}{2} f_i\right) y_{i-1} + \left(-2 + g_i h^2\right) y_i + \left(1 + \frac{h}{2} f_i\right) y_{i+1} = r_i h^2, \quad i = 1, 2, \ldots, n-1$$

with

$$y_0 = a \quad \text{and} \quad y_n = b$$

The above equation with above conditions comprise a *tri-diagonal system* which can be solved by the method outlined later in this paper.
The solution of this tri-diagonal system constitutes an approximate solution of the boundary value problem defined by the previous section.
To estimate the error in the numerical solution, we define the *local truncation error*, τ, by

$$\tau = \left(\frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} - y''_i\right) + f_i \left(\frac{y_{i+1} - y_{i-1}}{2h} - y'_i\right)$$

Expanding y_{i-1} and y_{i+1} by Taylor's Series and simplifying, the above gives

$$\tau = \frac{h^2}{12}\left(y_i^{\mathrm{iv}} + 2 f_i y'''_i\right) + \mathrm{O}\left(h^4\right)$$

Thus, the finite difference approximation defined above has *second-order accuracy* for functions with continuous fourth derivatives on $[x_0, x_n]$.
Further, it follows that $\tau \to 0$ as $h \to 0$, implying that greater accuracy in the result can be achieved by using a smaller value of h. In such a case, of course, more computational effort would be required since the number of equations become larger.

An easier way to improve accuracy is to employ *Richardson's deferred approach to the limit*, assuming that the $\mathrm{O}(h^2)$ error is propotional to h^2.
This means that the error has the form,

$$y(x_i) - y_i = h^2 e(x_i) + \mathrm{O}\left(h^4\right)$$

For Extrapolation to the limit, we can solve our finite-difference approximation differential equation twice, with the interval lengths h and $h/2$ respectively. Let the corresponding solutions be denoted by $y_i(h)$ and $y_i(h/2)$. For a point x_i common to both, we therefore have

$$y(x_i) - y_i(h) = h^2 e(x_i) + \mathrm{O}\left(h^4\right)$$

and

$$y(x_i) - y_i\left(\frac{h}{2}\right) = \frac{h^2}{4} e(x_i) + \mathrm{O}\left(h^4\right)$$

for which we obtain

$$y(x_i) = \frac{4y_i(h/2) - y_i(h)}{3}$$

The above method is explained with the simple boundary conditions where the function values on the boundary are prescribed. In many applied problems, however, *derivative boundary conditions* may be prescribed, and this requires a modification of the procedures described above.

4 Solution of Tri-Diagonal Systems

Consider the system of linear equations defined by:

$$\begin{aligned} b_1u_1 + c_1u_2 &= d_1 \\ a_2u_1 + b_2u_2 + c_2u_3 &= d_2 \\ a_3u_2 + b_3u_3 + c_3u_4 &= d_3 \\ &\vdots \\ a_nu_{n-1} + b_nu_n &= d_n \end{aligned}$$

The Matrix of coefficients is

$$A = \begin{bmatrix} b & c_1 & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ a_2 & b_2 & c_2 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & a_3 & b_3 & c_3 & \cdots & \cdots & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & 0 & a_n & b_n \end{bmatrix}$$

Matrices of the above type are called *tri-diagonal matrices*, occur frequently in the solutions of ordinary and partial differential equations by finite difference methods. The method of

factorization which is discussed below can be conveniently applied to solve the system can also be employed to solve this system. For example,

$$\begin{bmatrix} b_1 & c_1 & 0 \\ a_2 & b_2 & c_2 \\ 0 & a_3 & b_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ 0 & l_{32} & 1 \end{bmatrix} \begin{bmatrix} b_1 & c_1 & 0 \\ 0 & u_{22} & c_2 \\ 0 & 0 & u_{33} \end{bmatrix}$$

This Matrix Equation gives,

$$\begin{aligned} l_{21}b_1 &= a_2, \quad & l_{21}c_1 + u_{22} &= b_2 \\ l_{32}u_{22} &= a_3, \quad & l_{32}c_2 + u_{33} &= b_3 \end{aligned}$$

From these four equations we can compute, $l_{21}, u_{22}, l_{32}, u_{33}$ and these values are stored in the locations occupied by a_2, b_2, a_3 and b_3 respectively.
These computations can be achieved by the following statements:

Algorithm 1 *LU* Decomposition

```
for i = 2 to N do
    a(i) ← a(i)/b(i − 1)
    b(i) ← b(i) − a(i)c(i − 1)
end for
```

When the decomposition is complete, forward and back substitutions give the required solution. The algorithm is due to Thomas and possesses all the advantages of the *LU Decomposition*.

5 LU Decomposition from Gauss Elimination

We know that the Gaussian Elimination consists of reducing the coefficient matrix to an *upper-triangular* form. We show here how to arrive at *LU* decomposition using Gaussian Elimination. The upper-triangular form which the coefficient matrix is reduced is actually the upper triangular matrix U of the decomposition LU. For determining L, we consider,

$$Ax = B$$

where,

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}, \quad X = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

To eliminate x_1 from the equation, we multiply the first equation by a_{21}/a_{11} and subtract it from the second equation. We then obtain,

$$\left(a_{22} - a_{12}\frac{a_{21}}{a_{11}}\right)x_2 + \left(a_{23} - a_{13}\frac{a_{21}}{a_{11}}\right)X_3 = \left(b_2 - b_1\frac{a_{21}}{a_{11}}\right)$$

or

$$a'_{22}x_2 + a'_{23}x_3 = b'_2$$

The Factor $l_{21} = a_{21}/a_{11}$ is called the *multiplier* for eliminating x_1 from the second equation. Similarly, the multiplier for eliminating x_1 from third equation is given by $l_{31} = a_{31}/a_{11}$. After this elimination, the system is of the form,

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\ a'_{22}x_2 + a'_{23}x_3 &= b'_2 \\ a'_{32}x_2 + a'_{33}x_3 &= b'_3 \end{aligned}$$

In the final step, we have to eliminate x_2 from the third equation. For this we multiply second equation with $l_{32} = a'_{32}/a'_{11}$ and then subtract it from the third equation to obtain,

$$a''_{33}x_3 = b''_3$$

where the double primes indicate that the concerned has changed values twice. The final form of matrix of coefficients is the upper-triangular matrix given by

$$U = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a'_{22} & a'_{23} \\ 0 & 0 & a''_{33} \end{bmatrix}$$

The above matrix suggests that in any computer program, the places occupied by zero elements may be used to store the value of multipliers. Thus after elimination, matrix A can be written as

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ l_{21} & a'_{22} & a'_{23} \\ l_{31} & l_{32} & a''_{33} \end{bmatrix}$$

which represents the storage of LU decomposition of A with

$$L = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a'_{22} & a'_{23} \\ 0 & 0 & a''_{33} \end{bmatrix}$$

It is easily verified that $A = LU$.

6 Back Substitution in Gauss Elimination

For a system of n-linear equations in n-variables, the above discussed Gauss Elimination Method can be employed to get an equivalent upper-triangular system, which can be solved using *back substitution.*
Let the system of n linear equations in n unknowns be given as:

$$\begin{aligned}
&a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1\\
&a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = b_2\\
&\vdots\\
&a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_n = b_n
\end{aligned}$$

There are two steps in the solution of the above given system namely elimination of unknowns and back substitution.
Step 1: The unknowns are eliminated to get the upper-triangular system.
To eliminate x_1 from the second equation, we multiply the first equation by $(-a_{21}/a_{11})$ and obtain

$$-a_{21}x_1 - a_{12}\frac{a_{21}}{a_{11}}x_2 - a_{13}\frac{a_{21}}{a_{11}}x_3 - \cdots - a_{1n}\frac{a_{21}}{a_{11}}x_n = -b_1\frac{a_{21}}{a_{11}}$$

Adding the above equation to the second equation we get:

$$\left(a_{22} - a_{12}\frac{a_{21}}{a_{11}}\right)x_2 + \left(a_{23} - a_{13}\frac{a_{21}}{a_{11}}\right)x_3 + \cdots + \left(a_{2n} - a_{1n}\frac{a_{21}}{a_{11}}\right)x_n = b_2 - b_1\frac{a_{21}}{a_{11}}$$

which can be written as,

$$a'_{22}x_2 + a'_{23}x_3 + \cdots + a'_{2n}x_n = b'_2$$

In a similar fashion we can eliminate x_1 from the remaining equations and after eliminating x_1 from the last equation we obtain a system of equations represented as:

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1\\
a'_{22}x_2 + a'_{23}x_3 + \cdots + a'_{2n}x_n &= b'_2\\
a'_{32}x_2 + a'_{33}x_3 + \cdots + a'_{3n}x_n &= b'_3\\
&\vdots\\
a'_{n2}x_2 + a'_{n3}x_3 + \cdots + a'_{nn}x_n &= b'_n
\end{aligned}$$

We next eliminate x_2 from the last $(n-2)$ equations. Before this, it is important to notice that in the process of obtaining the above system, we have multiplied the first row by $(-a_{21}/a_{11})$ i.e we have divided it by a_{11} which is therefore assumed to be nonzero. For this reason, the first equation in the system is called the pivot equation and a_{11} is called the pivot element. This method obviously fails if $a_{11} = 0$. We shall discuss this important point after completing the description of the elimination method. Now to eliminate x_2 from the third equation, we multiply the second equation by $(-a'_{32}/a'_{22})$ and add it to the third equation.

Repeating this process with the remaining equations, we obtain the system

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\ a'_{22}x_2 + a'_{23}x_3 + \cdots + a'_{2n}x_n &= b'_2 \\ a''_{33}x_3 + \cdots + a''_{3n}x_n &= b''_3 \\ &\vdots \\ a''_{n3}x_3 + \cdots + a''_{nn}x_n &= b''_n \end{aligned}$$

The double primes indicate that the elements have changed twice. It is easily seen that this procedure can be continued to eliminate x_3 from the fourth equation onward, x_4 from the fifth equation onward, etc.,till we finally obtain upper triangular form:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\ a'_{22}x_2 + a'_{23}x_3 + \cdots + a'_{2n}X_n &= b'_2 \\ a''_{33}x_3 + \cdots + a''_{3n}x_n &= b''_3 \\ &\vdots \\ a^{(n-1)}_{nn}x_n &= b^{(n-1)}_n \end{aligned}$$

where $a^{(n-1)}_{nn}$ indicates that the element a_{nn} has changed $(n-1)$ times. We thus have completed the first step of elimination of unknowns and reduction to the upper triangular form.

$$x_n = \frac{b^{(n-1)}_n}{a^{(n-1)}_{nn}}$$

Step 2: We now have to obtain the required solution from the system. From the last equation of this system, we obtain This is then substituted in $(n-1)^{th}$ equation to obtain xn-1 and the process is repeated to compute the other unknowns. we have therefore first computed x_n, then $x_{n-1}, x_{n-2}, \cdots, x_2, x_1$ in that order. Due to this reason, the process is called *back substitution*.

7 Tools and Objectives

You are given the C++ code snippet that transforms any matrix M into *row-canonical* form. Source:https://rosettacode.org/wiki/Reduced_row_echelon_form

Algorithm 2 Gauss Elimination or Row - Echelon Form of Matrix

Require: Matrix M
  $lead \leftarrow 0$
  $rowCount \leftarrow$ the number of rows in M
  $columnCount \leftarrow$ the number of columns in M
  for $0 \leq r < rowCountdo$ **do**
    if $columnCount \leq lead$ **then**
      end
    end if
    $i \leftarrow r$
    while $M[i, lead] = 0$ **do**
      $i \leftarrow i + 1$
      if $rowCount = i$ **then**
        $i \leftarrow r$
        $lead \leftarrow lead + 1$
        if $columnCount = lead$ **then**
          end
        end if
      end if
    end while
    Swap rows i and r
    if $M[r, lead]$ is not 0 **then**
      divide row r by $M[r, lead]$
    end if
    for $0 \leq i < rowCount$ **do**
      if $i \neq r$ **then**
        Subtract $M[i, lead]$ multiplied by row r from row i
      end if
    end for
    $lead \leftarrow lead + 1$
  end for

Task Objectives:

i Translate the given differential equation to finite difference approximation as illustrated in Finite Difference Method section.

ii Use boundary condtions $\mu = 0.1$, $\kappa = 0.08$, $H = 3.0$, $U = 5.0$, $T_1 = 0$ and $T_2 = 5$.

iii Compute the tri-diagonal system out of the finite difference approximation for a step size $h = 0.001$ and $h = 0.0001$.

iv Use the C++ code snippet *row_echelon_form.cpp* to convert tri-diagonal system into a row-canonical form of Matrix.

v Do back substitution as illustrated above and find y_i's corresponding to x_i's.

vi Save the data in a **.csv* file and plot them using Excel or MATLAB.