Numerical Methods I

System of Linear Algebraic Equations: Naïve Gauss Elimination

```
program mainGaussEliminationNaive
    implicit none

    integer :: n
    real, dimension(:,:), allocatable :: a
    real, dimension(:), allocatable :: x

    integer :: rowCount

    n = 5
    allocate(a(n, n+1))
    allocate(x(n))

    a(1,1) = 1
    a(1,2) = -1
    a(1,3) = 2
    a(1,4) = -3
    a(1,5) = 4
    a(1,6) = -35.4

    a(2,1) = 2
    a(2,2) = 3
    a(2,3) = -1
    a(2,4) = 5
    a(2,5) = -2
    a(2,6) = 32.4

    a(3,1) = -1
    a(3,2) = 3
    a(3,3) = 2
    a(3,4) = -5
    a(3,5) = 1
    a(3,6) = -17.9

    a(4,1) = 1
    a(4,2) = 2
    a(4,3) = 1
    a(4,4) = 2
    a(4,5) = 3
    a(4,6) = -13.9

    a(5,1) = -4
    a(5,2) = -6
    a(5,3) = -2
    a(5,4) = 8
    a(5,5) = -3
    a(5,6) = 4.9

    call gaussEliminationNaive(a, n, x)
```

```
    write(*,*) "Solution:"
    do rowCount = 1, n
        write(*,10) x(rowCount)
    end do
    10 format(f5.2)
end program mainGaussEliminationNaive
```

```
subroutine gaussEliminationNaive(aIn, n, x)
    implicit none

    integer, intent(in) :: n
    real, dimension(n, n+1), intent(in) :: aIn
    real, dimension(n), intent(out) :: x

    real, dimension(n, n+1) :: a

    integer :: stepCount, rowCount, columnCount

    integer :: pivotRow, pivotColumn
    real :: pivot
    real :: factor

    write(*,*)
    write(*,*) "System of Linear Algebraic Equations"
    write(*,*) "Method: Naive Gauss Elimination"
    write(*,*)

    a = aIn
    write(*,*) "The Augmented Matrix"
    call printMatrix2D(a, n, (n + 1))

    write(*,*) "Part 1: Forward Elimination"
    write(*,*) "(Reduce the coefficient matrix to upper triangular form)"
    write(*,*)
    do stepCount = 1, (n - 1)
        write(*,20) "Step #", stepCount
        write(*,30) "Eliminate sub-diagonal elements of column #", stepCount

        pivotRow = stepCount
        pivotColumn = stepCount
        pivot = a(pivotRow, pivotColumn)

        do rowCount = (pivotRow + 1), n
            factor = a(rowCount, pivotColumn) / pivot
            do columnCount = pivotColumn, (n + 1)
                a(rowCount, columnCount) = a(rowCount, columnCount) - factor *
a(pivotRow, columnCount)
            end do
        end do
        call printMatrix2D(a, n, (n + 1))
    end do

    write(*,*) "Part 2: Back Substitution"
    write(*,50) "Evaluate x(", n, ")"
    x(n) = a(n, n+1) / a(n, n)
    write(*,40) "x(n) = ", x(n)
    write(*,*)

    do rowCount = (n - 1), 1, -1
        write(*,50) "Evaluate x(",rowCount, ")"
        factor = 0
```

```
        do stepCount = (rowCount + 1), n
            factor = factor + a(rowCount, stepCount) * x(stepCount)
        end do
        x(rowCount) = (1 / a(rowCount, rowCount)) * (a(rowCount, n+1) - factor)
        write(*,60) "x(",rowCount, ") = ", x(rowCount)
        write(*,*)
    end do
    20 format(a6, i1)
    30 format(a43, i1)
    40 format(a6, f5.2)
    50 format(a11, i1, a1)
    60 format(a3, i1, a4, f5.2)
end subroutine gaussEliminationNaive
```

```
subroutine printMatrix2D(matrix, rows, columns)
    implicit none

    integer, intent(in) :: rows, columns
    real, dimension(rows, columns), intent(in) :: matrix

    integer :: rowCount, columnCount

    do rowCount = 1, rows
        do columnCount = 1, columns
            write(*,10, advance='no') matrix(rowCount, columnCount)
        end do
        write(*,*)
    end do
    write(*,*)

    10 format(f7.2)
end subroutine printMatrix2D
```

Output

```
 System of Linear Algebraic Equations
 Method: Naive Gauss Elimination

 The Augmented Matrix
   1.00  -1.00   2.00  -3.00   4.00 -35.40
   2.00   3.00  -1.00   5.00  -2.00  32.40
  -1.00   3.00   2.00  -5.00   1.00 -17.90
   1.00   2.00   1.00   2.00   3.00 -13.90
  -4.00  -6.00  -2.00   8.00  -3.00   4.90

 Part 1: Forward Elimination
 (Reduce the coefficient matrix to upper triangular form)

Step #1
Eliminate sub-diagonal elements of column #1
   1.00  -1.00   2.00  -3.00   4.00 -35.40
   0.00   5.00  -5.00  11.00 -10.00 103.20
   0.00   2.00   4.00  -8.00   5.00 -53.30
   0.00   3.00  -1.00   5.00  -1.00  21.50
   0.00 -10.00   6.00  -4.00  13.00-136.70

Step #2
Eliminate sub-diagonal elements of column #2
   1.00  -1.00   2.00  -3.00   4.00 -35.40
   0.00   5.00  -5.00  11.00 -10.00 103.20
   0.00   0.00   6.00 -12.40   9.00 -94.58
   0.00   0.00   2.00  -1.60   5.00 -40.42
   0.00   0.00  -4.00  18.00  -7.00  69.70

Step #3
Eliminate sub-diagonal elements of column #3
   1.00  -1.00   2.00  -3.00   4.00 -35.40
   0.00   5.00  -5.00  11.00 -10.00 103.20
   0.00   0.00   6.00 -12.40   9.00 -94.58
   0.00   0.00   0.00   2.53   2.00  -8.89
   0.00   0.00   0.00   9.73  -1.00   6.65

Step #4
Eliminate sub-diagonal elements of column #4
   1.00  -1.00   2.00  -3.00   4.00 -35.40
   0.00   5.00  -5.00  11.00 -10.00 103.20
   0.00   0.00   6.00 -12.40   9.00 -94.58
   0.00   0.00   0.00   2.53   2.00  -8.89
   0.00   0.00   0.00   0.00  -8.68  40.82

 Part 2: Back Substitution
Evaluate x(5)
x(n) =-4.70

Evaluate x(4)
 x(4) =  0.20
```

```
Evaluate x(3)
 x(3) = -8.30

Evaluate x(2)
 x(2) =  2.50

Evaluate x(1)
 x(1) =  3.10

 Solution:
 3.10
 2.50
-8.30
 0.20
-4.70
```