Problem Set 2

DATA 605, Spring 2021: Assignment 2

Philip Tanofsky

07 February 2021

Prompt

Write an R function to factorize a square matrix A into LU or LDU, whichever you prefer. Please submit your response in an R Markdown document.

You don't have to worry about permuting rows of A and you can assume that A is less than 5×5, if you need to hard-code any variables in your code.

Solution

Based on approach from "LU Decomposition Shortcut Method" video.

```
# Using the algorithm from the LU Decomposition Shortcut Method video
# http://www.youtube.com/watch?v=UlWcofkUDDU

lu_decomposition <- function(matrix_in) {

  # Get row count of matrix
  n <- dim(matrix_in)[1]

  # Get column count of matrix
  m <- dim(matrix_in)[2]

  # Return error message if matrix not a square
  if (n != m) {
    return("Error: Matrix not a square")
  }

  # Return error message if matrix too big
  if (n > 4) {
    return("Matrix greater than 4x4")
  }

  # Set upper matrix to input matrix
  upper <- matrix_in
  # Define lower matrix with diagonal (identity matrix)
  lower <- diag(n)
```

```
  # Outer loop to traverse the matrices top to bottom by row
  for (i in 2:n) {

    # Inner loop to traverse the matrices left to right by column
    for (j in 1:(i-1)) {

      # Calculate multiplier
      # Really, just divide the values and set lower matrix value per i,j
      lower[i,j] <- upper[i,j] / upper[j,j]

      # Calculate the row values for the upper matrix
      # Opposite of multiplier (lower value just set) mutiplied by above row in upper matrix
      # then added to row under consideration in upper matrix
      upper[i, ] <- -lower[i,j] * upper[j, ] + upper[i, ]
      }
  }

  # Outputs
  print("Input Matrix")
  print(matrix_in)
  print("Upper Matrix")
  print(upper)
  print("Lower Matrix")
  print(lower)

  # Return resulting upper and lower matrices
  return(list(upper,lower))
}
```

Validations

```
# Validations
#Not Square
#mat <- matrix(c(1, 2, 1, 2, 4, 3), nrow=2, ncol=3)
#Too big
#mat <- matrix(c(rep(2,36)), nrow=6, ncol=6)
#Invalid
#mat <- matrix(c(1, 2, 1, 2, 4, 3, 3, 5, 4), nrow=3, ncol=3)
#Valid
#mat <- matrix(c(2, -4, -4, -1, 6, -2, -2, 3, 8), nrow=3, ncol=3)
#Valid
mat <- matrix(c(-3, -12, 6, 0, -3, -15, 12, -15, 2, 6, 4, 6, 2, 12, -10, 29), nrow=4, ncol=4)

result <- lu_decomposition(mat)
```

```
## [1] "Input Matrix"
##      [,1] [,2] [,3] [,4]
## [1,]   -3   -3    2    2
## [2,]  -12  -15    6   12
## [3,]    6   12    4  -10
```

```
## [4,]    0  -15    6   29
## [1] "Upper Matrix"
##      [,1] [,2] [,3] [,4]
## [1,]   -3   -3    2    2
## [2,]    0   -3   -2    4
## [3,]    0    0    4    2
## [4,]    0    0    0    1
## [1] "Lower Matrix"
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    4    1    0    0
## [3,]   -2   -2    1    0
## [4,]    0    5    4    1
```

```
# Output result
result
```

```
## [[1]]
##      [,1] [,2] [,3] [,4]
## [1,]   -3   -3    2    2
## [2,]    0   -3   -2    4
## [3,]    0    0    4    2
## [4,]    0    0    0    1
##
## [[2]]
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    4    1    0    0
## [3,]   -2   -2    1    0
## [4,]    0    5    4    1
```

```
# If result was not an error, then compare results to original matrix
if (is.list(result)) {
  l_times_u <- result[[2]] %*% result[[1]]
  (mat == l_times_u)
}
```

```
##      [,1] [,2] [,3] [,4]
## [1,] TRUE TRUE TRUE TRUE
## [2,] TRUE TRUE TRUE TRUE
## [3,] TRUE TRUE TRUE TRUE
## [4,] TRUE TRUE TRUE TRUE
```