数値計算法　演習課題 3　提出日：2024 年 6 月 27 日

202310330　長田悠生

課題 1

(1.1)

M を求める関数

以下の関数は、M を求めるプログラムである。あたえられている関数である Pij を利用して作成した。

(1.1)

```
function Pij(i, j, alpha, n)
  In = Matrix{Float64}(I, n, n)  # n次元の単位行列の作成
  P = In + alpha * In[:, i] * In[:, j]'
  return P
end

function make_m(a::Matrix{Float64})::Vector{Matrix{Float64}}
  P_vec::Vector{Matrix{Float64}} = []
  for j = 1:(size(a)[2]-1)      #列
    m = Pij((j + 1), j, -(a[(j+1), j] / a[j, j]), (size(a)[1]))
    for i = (j+2):(size(a)[1]) #行
      m *= Pij(i, j, -(a[i, j] / a[j, j]), (size(a)[1]))

    end
    push!(P_vec, m)
  end
  return P_vec
end
```

U を求める関数

以下が U を求める関数である。上記の M を求める関数を U を求める関数の内部で利用している。

(1.1)

```
function make_u(a::Matrix{Float64})::Matrix{Float64}
  m::Vector{Matrix{Float64}} = make_m(a)
  for i = eachindex(m)
    a = m[i] * a
```

```
  end
  return a
end
```

L を求める関数

以下が L を求める関数である。上記の M を求める関数を L を求める関数の内部で利用している。

(1.1)

```
function make_l(a::Matrix{Float64})::Matrix{Float64}
  m::Vector{Matrix{Float64}} = make_m(a)
  l::Matrix{Float64} = inv(m[length(m)])
  for i = 1:(length(m)-1)
    l = inv(m[length(m)-i]) * l
  end
  return l
end
```

課題 1 の全体のプログラム

実行結果に $M_{(1)}$ から $M_{(3)}$ と U と L の値などを表示するようにしている。U と L の値はそれぞれ以下の値となった。

$$U = \begin{pmatrix} 4.0 & 3.0 & 2.0 & 1.0 \\ 0.0 & 1.75 & 1.5 & 1.25 \\ 0.0 & 0.1875 & 1.875 & 1.5625 \\ 0.0 & 0.234375 & 0.34375 & 1.953125 \end{pmatrix}$$

$$L = \begin{pmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.75 & 1.0 & 0.0 & 0.0 \\ 0.5 & 0.75 & 1.0 & 0.0 \\ 0.25 & 0.5 & 0.75 & 1.0 \end{pmatrix}$$

課題 1 の全体のプログラム

```
module Task1LU
  using LinearAlgebra

  function Pij(i, j, alpha, n)
    In = Matrix{Float64}(I, n, n)  # n次元の単位行列の作成
    P = In + alpha * In[:, i] * In[:, j]'
```

```
    return P
  end

  function make_m(a::Matrix{Float64})::Vector{Matrix{Float64}}
    P_vec::Vector{Matrix{Float64}} = []
    for j = 1:(size(a)[2]-1)      #列
      m = Pij((j + 1), j, -(a[(j+1), j] / a[j, j]), (size(a
          )[1]))
      for i = (j+2):(size(a)[1]) #行
        m *= Pij(i, j, -(a[i, j] / a[j, j]), (size(a)[1]))
      end
      push!(P_vec, m)
    end
    return P_vec
  end

  function make_u(a::Matrix{Float64})::Matrix{Float64}
    m::Vector{Matrix{Float64}} = make_m(a)
    for i = eachindex(m)
      a = m[i] * a
    end
    return a
  end

  function make_l(a::Matrix{Float64})::Matrix{Float64}
    m::Vector{Matrix{Float64}} = make_m(a)
    l::Matrix{Float64} = inv(m[length(m)])
    for i = 1:(length(m)-1)
      l = inv(m[length(m)-i]) * l
    end
    return l
  end

end

using .Task1LU
```

数値計算法　演習課題 3　提出日：2024 年 6 月 27 日

202310330　長田悠生

```
a = [
  4.0 3.0 2.0 1.0
  3.0 4.0 3.0 2.0
  2.0 3.0 4.0 3.0
  1.0 2.0 3.0 4.0
]

#(1.1)
m = Task1LU.make_m(a)
for i = eachindex(m)
  println("M($i) = $(m[i])")
end

println("====================")

u = Task1LU.make_u(a)
println("U = $u")

l = Task1LU.make_l(a)
println("L = $l")

print("L * U = $(l * u)")
```

課題 1 のプログラムの実行結果

課題 1 のプログラムの実行結果

```
$ julia --project ./src/1.jl
M(1) = [1.0 0.0 0.0 0.0; -0.75 1.0 0.0 0.0; -0.5 0.0 1.0 0.0;
    -0.25 0.0 0.0 1.0]
M(2) = [1.0 0.0 0.0 0.0; 0.0 1.0 0.0 0.0; 0.0 -0.75 1.0 0.0; 0.0
     -0.5 0.0 1.0]
M(3) = [1.0 0.0 0.0 0.0; 0.0 1.0 0.0 0.0; 0.0 0.0 1.0 0.0; 0.0
    0.0 -0.75 1.0]
====================
U = [4.0 3.0 2.0 1.0; 0.0 1.75 1.5 1.25; 0.0 0.1875 1.875
    1.5625; 0.0 0.234375 0.34375 1.953125]
```

数値計算法　演習課題 3　提出日：2024 年 6 月 27 日

202310330　長田悠生

```
L = [1.0 0.0 0.0 0.0; 0.75 1.0 0.0 0.0; 0.5 0.75 1.0 0.0; 0.25
    0.5 0.75 1.0]
L * U = [4.0 3.0 2.0 1.0; 3.0 4.0 3.0 2.0; 2.0 3.0 4.0 3.0; 1.0
    2.0 3.0 4.0]%
```

課題 2

(2.1)

以下が、A の第 1 行を $-\frac{3}{4}$ 倍したものを第 2 行に足すことで、A の 2 行 1 列の要素を 0 にする操作をするプログラムである。

(2.1)

```
a = [
  4.0 3.0 2.0 1.0
  3.0 4.0 3.0 2.0
  2.0 3.0 4.0 3.0
  1.0 2.0 3.0 4.0
]

#(2.1)
a[2, :] = (-3 / 4 * a[1, :]) + a[2, :]
```

(2.2)

以下が U を求める関数である。

(2.2)

```
function make_u(a::Matrix{Float64})::Matrix{Float64}
  for i = 1:(size(a)[1]-1)
    for j = (i+1):(size(a)[2])
      a[j, :] = -1 * (a[i, j] / a[i, i]) * a[i, :] + a[j, :]
    end
  end
  return a
end
```

数値計算法　演習課題 3　提出日：2024 年 6 月 27 日
202310330　長田悠生

(2.3)

以下が L を求める関数である。あたえられている関数である Pij を利用して作成した。

(2.3)

```
function Pij(i, j, alpha, n)
  In = Matrix{Float64}(I, n, n)  # n次元の単位行列の作成
  P = In + alpha * In[:, i] * In[:, j]'
  return P
end

function make_l(a::Matrix{Float64})::Matrix{Float64}
  l = Matrix{Float64}(I, size(a)[1], size(a)[2])
  for i = 1:(size(a)[1]-1)
    for j = (i+1):(size(a)[2])
      l *= Pij(j, i, (a[i, j] / a[i, i]), size(a)[1])
    end
  end
  return l
end
```

課題 2 の全体のプログラム

課題 2 の全体のプログラム

```
module Task2LU
  using LinearAlgebra

  function Pij(i, j, alpha, n)
    In = Matrix{Float64}(I, n, n)  # n次元の単位行列の作成
    P = In + alpha * In[:, i] * In[:, j]'
    return P
  end

  function make_u(a::Matrix{Float64})::Matrix{Float64}
    for i = 1:(size(a)[1]-1)
      for j = (i+1):(size(a)[2])
        a[j, :] = -1 * (a[i, j] / a[i, i]) * a[i, :] + a[j, :]
```

```
      end
    end
    return a
  end

  function make_l(a::Matrix{Float64})::Matrix{Float64}
    l = Matrix{Float64}(I, size(a)[1], size(a)[2])
    for i = 1:(size(a)[1]-1)
      for j = (i+1):(size(a)[2])
        l *= Pij(j, i, (a[i, j] / a[i, i]), size(a)[1])
      end
    end
    return l
  end
end

using .Task2LU

a = [
  4.0 3.0 2.0 1.0
  3.0 4.0 3.0 2.0
  2.0 3.0 4.0 3.0
  1.0 2.0 3.0 4.0
]

#(2.1)
a[2, :] = (-3 / 4 * a[1, :]) + a[2, :]
println("(2.1): $a")

a = [
  4.0 3.0 2.0 1.0
  3.0 4.0 3.0 2.0
  2.0 3.0 4.0 3.0
  1.0 2.0 3.0 4.0
]

#(2.2)
```

```
#Uの生成
u = Task2LU.make_u(a)
println("U = $u")

#(2.3)
#Lの生成
l = Task2LU.make_l(a)
println("L = $l")

# A = LU
  println("L * U = $(l * u)")
```

課題 2 のプログラムの実行結果

課題 2 のプログラムの実行結果

```
$ julia --project ./src/2.jl
(2.1): [4.0 3.0 2.0 1.0; 0.0 1.75 1.5 1.25; 2.0 3.0 4.0 3.0; 1.0
    2.0 3.0 4.0]
U = [4.0 3.0 2.0 1.0; 0.0 1.75 1.5 1.25; 0.0 0.0
   1.7142857142857144 1.4285714285714286; 0.0 0.0 0.0
   1.6666666666666667]
L = [1.0 0.0 0.0 0.0; 0.75 1.0 0.0 0.0; 0.5 0.8571428571428571
   1.0 0.0; 0.25 0.7142857142857143 0.8333333333333333 1.0]
L * U = [4.0 3.0 2.0 1.0; 3.0 4.0 3.0 2.0; 2.0 3.0 4.0 3.0; 1.0
   2.0 3.0 4.0]
```

数値計算法　演習課題 3　提出日：2024 年 6 月 27 日
202310330　長田悠生

(2.4)

n 次正方行列を A, n 行ある各要素が実数のベクトルを $\boldsymbol{x}, \boldsymbol{b}$ とおく。
そのとき、以下の式が与えられたとする。
$A\boldsymbol{x} = \boldsymbol{b}$
$\hat{A}$について、行列 A の i 行のα倍を j 行に足す操作に当たる計算を $P_{ij}(\alpha)A$,
行列 A の i 列について、$i+1$ 行から n までの要素が 0 になる操作に当たる計算を $M^{(i)}A$ とする。
すると、以下の式が得られる。
$\hat{A} = (A|\boldsymbol{b}) = [M^{(n-1)} \times M^{(n-2)} \times \cdots \times M^{(1)} \times A|\boldsymbol{b'}]$
$\left(M^{(i)} = P_{in}\left(-\dfrac{a_{in}}{a_{ii}}\right) \times P_{in-1}\left(-\dfrac{a_{n-1i}}{a_{ii}}\right) \times \cdots \times P_{i2}\left(-\dfrac{a_{2i}}{a_{ii}}\right)\right)$
したがって、以下の式が得られる。
$M^{(n-1)} \times M^{(n-2)} \times \cdots \times M^{(1)} = M$ とおくと、
$MA\boldsymbol{x} = M\boldsymbol{b}$
$MA = U$ とおく。
$U\boldsymbol{x} = \boldsymbol{b'}$
つまり、LU 分解で登場する U は、$U = MA$ である。
さらに、$U = MA$ について式変形を行う。
$MA = U$
$A = M^{-1}U$
$M^{-1} = L$ とおく。
$A = LU$
つまり、LU 分解で登場する L は、$L = M^{-1}$である。
このことから、U を導くために必要な M の逆行列が L となっていることがわかる。
また、L について、
$L = P_{12}(\alpha_1)^{-1} \times P_{13}(\alpha_2)^{-1} \times \cdots \times P_{n-1n}(\alpha_{\frac{n(n-1)}{2}})^{-1}$
$= E_n \times P_{12}(-\alpha_1) \times P_{13}(-\alpha_2) \times \cdots \times P_{n-1n}(-\alpha_{\frac{n(n-1)}{2}})$
となる。
U を導くためにαを求めるため、U を導く途中で計算した値を L を導くためにも用いることができる。

(2.5)

課題 1 で U や L を導くたときに、行列の積の計算をしていた。そのため、1 回行の基本行列を行う旅に $O(n^3)$ の計算量が必要になる。行うたびに $O(n^3)$ の計算量が必要になる。課題 2 で U や L を導くときに、対象の行列に対して、直接行の基本変形の計算をしていた。そのため、1 回行の

数値計算法　演習課題 3　提出日：2024 年 6 月 27 日

202310330　長田悠生

基本行列を行うたびに $O(n)$ の計算量が必要になる。そのため、課題 2 の方が計算量が少なく、良い実装だと考えられる。

数値計算法　演習課題 3　提出日：2024 年 6 月 27 日

202310330　長田悠生

課題 3

(3.1), (3.2)

以下のプログラムが、解 $\boldsymbol{x}$ と $\boldsymbol{b} - A\boldsymbol{x}$ 求めるためのプログラムである。

課題 3 の全体のプログラム

課題 3 の全体のプログラム

```
module BackwardSubstitution
  using LinearAlgebra

  module Task1LU
  using LinearAlgebra

  function Pij(i, j, alpha, n)
      In = Matrix{Float64}(I, n, n)  # n次元の単位行列の作成
      P = In + alpha * In[:, i] * In[:, j]'
      return P
  end

  function make_m(a::Matrix{Float64})::Vector{Matrix{Float64}}
      P_vec::Vector{Matrix{Float64}} = []
      for j = 1:(size(a)[2]-1)      #列
          for i = (j+1):(size(a)[1]) #行
              push!(P_vec, Pij(i, j, -(a[i, j] / a[j, j]), (size
                  (a)[1])))
          end
      end
      return P_vec
  end

  function make_u(a::Matrix{Float64})::Matrix{Float64}
      m::Vector{Matrix{Float64}} = make_m(a)
      for i = eachindex(m)
          a = m[i] * a
      end
```

```
    return a
end

function make_l(a::Matrix{Float64})::Matrix{Float64}
    m::Vector{Matrix{Float64}} = make_m(a)
    l::Matrix{Float64} = inv(m[length(m)])
    for i = 1:(length(m)-1)
        l = inv(m[length(m)-i]) * l
    end
    return l
end
end

module Task2LU
using LinearAlgebra

function Pij(i, j, alpha, n)
  In = Matrix{Float64}(I, n, n)  # n次元の単位行列の作成
  P = In + alpha * In[:, i] * In[:, j]'
  return P
end

function make_u(a::Matrix{Float64})::Matrix{Float64}
  for i = 1:(size(a)[1]-1)
    for j = (i+1):(size(a)[2])
      a[j, :] = -1 * (a[i, j] / a[i, i]) * a[i, :] + a[j, :]
    end
  end
  return a
end

function make_l(a::Matrix{Float64})::Matrix{Float64}
  l = Matrix{Float64}(I, size(a)[1], size(a)[2])
  for i = 1:(size(a)[1]-1)
    for j = (i+1):(size(a)[2])
      l *= Pij(j, i, (a[i, j] / a[i, i]), size(a)[1])
    end
```

```
    end
    return l
  end
  end

  using .Task1LU
  using .Task2LU
  function task1_bs(u::Matrix{Float64}, b::Vector{Float64})::
      Vector{Float64}
       #結果の行ベクトル
       result_vec::Vector{Float64} = zeros(Float64, 0)
       u_size::Tuple{Int64,Int64} = size(u)
       #初期値
       x::Float64 = b[1] / u[u_size[1], u_size[2]]
       pushfirst!(result_vec, x)
       for i = 2:(u_size[1])
           term::Float64 = 0.0
           result_vec_counter::Int64 = 1
           for n = (u_size[1]-(i-2)):u_size[1]
               term += (u[(u_size[1]-(i-1)), n] / u[(u_size[1]-(i
                  -1)), (u_size[2]-(i-1))]) * result_vec[
                  result_vec_counter]
               result_vec_counter += 1
           end
           x = (b[u_size[1]-i+1] / u[(u_size[1]-i+1), (u_size[2]-
              i+1)]) - term
           pushfirst!(result_vec, x)
       end
       return result_vec
  end

  function solution_error(a::Matrix{Float64}, x::Vector{Float64
      }, b::Vector{Float64})
    return b - a*x
  end
end
```

数値計算法　演習課題 3　提出日：2024 年 6 月 27 日

202310330　長田悠生

```
using .BackwardSubstitution

a = [
    4.0 3.0 2.0 1.0
    3.0 4.0 3.0 2.0
    2.0 3.0 4.0 3.0
    1.0 2.0 3.0 4.0
]

b = [
    1.0
    1.0
    1.0
    1.0
]

#課題1のパターン
u1::Matrix{Float64} = BackwardSubstitution.Task1LU.make_u(a)
kadai1_solution = BackwardSubstitution.task1_bs(u1, b)
println("課題1の関数を用いて計算したときの解")
println(kadai1_solution)

kadai1_error = BackwardSubstitution.solution_error(a,
    kadai1_solution, b)
println("課題1の関数を用いて計算したときの解の誤差")
println(kadai1_error)


#課題2のパターン
u2::Matrix{Float64} = BackwardSubstitution.Task2LU.make_u(a)
kadai2_solution = BackwardSubstitution.task1_bs(u2, b)
println("課題2の関数を用いて計算したときの解")
println(kadai2_solution)

kadai2_error = BackwardSubstitution.solution_error(a,
    kadai2_solution, b)
println("課題2の関数を用いて計算したときの解の誤差")
```

```
println(kadai2_error)
```

課題 3 のプログラムの実行結果

課題 3 のプログラムの実行結果

```
$ julia --project ./src/3.jl
課題1の関数を用いて計算したときの解
[-0.01704761904761909, 0.11428571428571432, 0.10666666666666663,
    0.512]
課題1の関数を用いて計算したときの解の誤差
[1.1102230246251565e-16, -0.75, -1.2714285714285714,
   -1.5795238095238093]
課題2の関数を用いて計算したときの解
[0.004761904761904745, 0.07142857142857145, 0.08333333333333331,
    0.6]
課題2の関数を用いて計算したときの解の誤差
[1.1102230246251565e-16, 0.0, 0.0, 0.0]
```

課題 4

(4.1)

(4.2)

(4.3)

課題 5

(5.1)

(5.2)