Julia コード

$$y^2 - x^3 + 4x = 0, \qquad \left(\frac{x}{2}\right)^4 + y^4 = 1 \tag{1}$$

連立非線形方程式の数値解を Newton 法で計算する。
上記方程式を関数値ベクトルとして f(x) とし、ヤコビ行列を **Df(x)** として定義する。

```
using LinearAlgebra

# 関数値ベクトル
function f(x)
   return [ x[2]^2-x[1]^3+4*x[1],
            (x[1]/2)^4+x[2]^4-1 ]
end

# ヤコビアン 偏導関数行列
function Df(x)
   return [ -3*x[1]^2+4  2*x[2];
            x[1]^3/4    4*x[2]^3 ]
end

function newton(x0, f, Df)
    maxiter = 100  # 最大反復回数. 適宜調整する.
    tol = 1e-6     # 停止条件の tolerance. 適宜調整.
    x2 = x1 = x0 # 初期値

    for i in 1:maxiter

        # 行列Df が 退化している場合
        if det(Df(x1))==0
            return "degenerate"
        end

```

```
        x2 = x1 - Df(x1)\f(x1)

        # 途中経過の表示
        @show x2

        # 停止条件の判定
        if norm(x2 - x1) < tol
            println("Converged in $i iterations.")
            break
        end

        x1 = x2
    end
    return x2    # 数値解を返却
end

newton([0,1.0],f,Df)
```

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{pmatrix} \tag{2}$$

A の固有値を求め、連立 1 次方程式 $A\boldsymbol{x} = \boldsymbol{b}$, $b = (0, 0, 0, 0, 6)^T$ を CG 法 (共役勾配法) で解け。

```
using LinearAlgebra

A = Float64[2 -1 0 0 0; -1 2 -1 0 0; 0 -1 2 -1 0; 0 0 -1 2 -1; 0 0 0 -1 2]

b = Float64[0, 0, 0, 0, 6]
x = [0, 0, 0, 0, 0]
r = b - A*x
```

```
p = r
tol = 1e-6   # 停止条件のtolerance

# 固有値の確認
@show eigvals(A)

# CG法による反復計算
for i in 1:20
    alpha = r'*p/(p'*A*p)
    xx = x + alpha*p
    # 停止条件
    (norm(xx - x) < tol) && break
    global x = xx
    @show i, Float64.(x)   # 途中経過の表示
    global r = r - alpha*A*p
    beta = r'*A*p/(p'*A*p)
    global p = r - beta*p
end

@show x   # 数値解の表示
```

以下の定積分を複合台形公式にて計算せよ。

$$I = \int_0^{\pi} \sin x \mathrm{d}x \tag{3}$$

```
# 複合台形公式
function comp_trapezoidal_rule(f,a,b, N)
    h = (b-a)/N
    return h/2*sum(f(a + (i-1)*h) + f(a + i*h) for i in 1:N)
end

f(x) = sin(x) # 被積分関数
a = 0         # 区間始点
b = pi        # 区間終点
```

```
N = 80          # 分割数


@show I = comp_trapezoidal_rule(f,a,b,N)

# 誤差表示
@show  1-I
```

常微分方程式の初期値問題を考える。

$$u'(t) = u(t) + 1, \quad u(0) = 1 \tag{4}$$

```
# Runge-Kutta法
# http://www.isc.meiji.ac.jp/~mcelab/jyo_en2/2021/09/index.htm
# f:微分方程式 # t:時刻 # h:刻み幅
# x:関数 # n:繰返し回数
function RK4(f, t, x, h, n)
    for i in 1:n
        k0 = f(t,       x)
        k1 = f(t + h/2, x + k0*h/2)
        k2 = f(t + h/2, x + k1*h/2)
        k3 = f(t + h,   x + k2*h)
        x += (k0 + 2*k1 + 2*k2 + k3)* h/6
        t += h
    end
    return x
end

#####
# 微分方程式 u'(t) = u(t) + 1 の関数定義
function f(t, u)
    return u + 1
end
```