Stokes Second Problem

2020136006 정성원

Problem

무한하게 펼쳐진 평평한 벽이 주기적인 진동을 한다고 가정한다(See Figure 1). 이 때, 점착조건(No-slip condition)으로 벽에서의 속도 $u(0,t) = U_0 cos(nt)$ 를 만족한다.

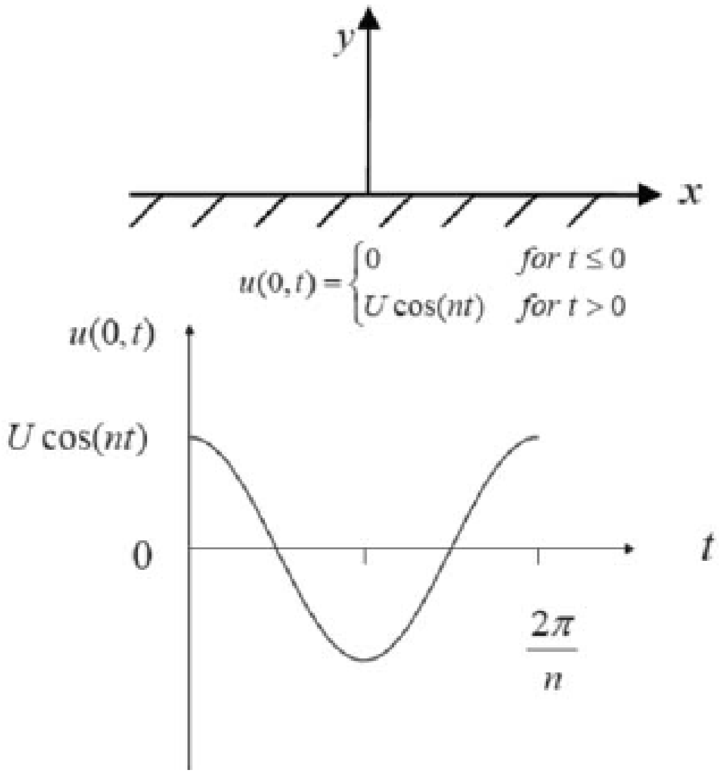
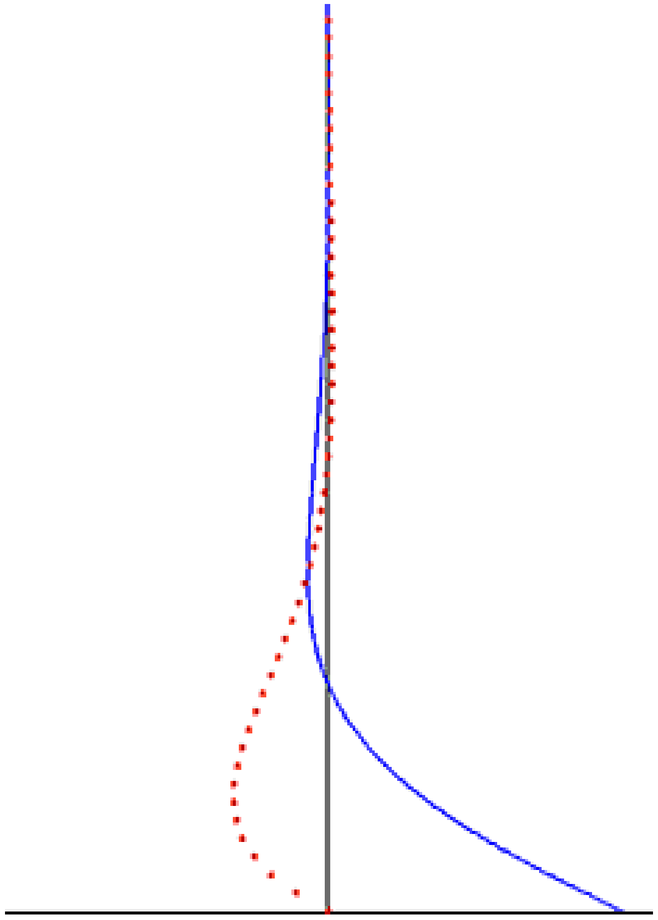


Figure 1 Schematic diagram of Stokes second problem



Problem

1. (Analytic solution)

(1) 3차원 나비에-스톡스 방정식에서, Stokes second problem을 풀기 위한 간소화 된 지배방정식을 유도하시오. 유도 과정에서 사용되는 가정들 또한 서술하시오.

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial y^2}$$

N-S Eq

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

Problem

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

Assumption

$u = u(y, t)$ 이므로 $\frac{\partial u}{\partial x} = 0$ → $\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial y^2}\right)$

$v = 0$ 라고 가정 → $\frac{\partial u}{\partial t} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial y^2}\right)$

압력 구배 $\frac{\partial p}{\partial x} = 0$ 라고 가정 → $\frac{\partial u}{\partial t} = \nu\left(\frac{\partial^2 u}{\partial y^2}\right)$

Problem

(2) 위에 주어진 Stokes second problem의 해가 다음과 같음을 보이시오.

$$u(y,t) = U_0 e^{-\eta_s} \cos(nt - \eta_s)\,,\ where\ \eta_s = \sqrt{\frac{n}{2\nu}}\,y.$$

$$\frac{\partial u}{\partial t} = \nu\left(\frac{\partial^2 u}{\partial y^2}\right)$$

$$u(0,t) = U_0 cosnt$$

방정식과 경계 조건이 모두 선형이므로 속도는 복소 함수의 실수부로 나타낼 수 있음.

$$u = U_0 Re\left(e^{i\omega t} f(y)\right)$$ ($Re(f)$: f의 실수부)

$U_0 e^{i\omega t} f(y)$를 $\frac{\partial u}{\partial t} = \nu\left(\frac{\partial^2 u}{\partial y^2}\right)$에 대입하여 계산하면

$$f''(y) - \frac{in}{\upsilon} f(y) = 0$$

Problem

$$f''(y) - \frac{in}{v} f(y) = 0$$

$$\therefore\ f(y) = Aexp\left(\sqrt{\frac{in}{v}}y\right) + Bexp\left(-\sqrt{\frac{in}{v}}y\right)$$

$f(0) = 1, f(\infty) = 0$를 만족해야 하므로 $f(0) = A + B = 1\,, f(\infty) = A = 0$

$$\therefore\ f(y) = exp\left(-\sqrt{\frac{in}{v}}y\right) = \exp\left(-\sqrt{\frac{n}{v}}y\left(\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}i\right)\right)$$

$$f(y) = \exp\left(-\sqrt{\frac{n}{2v}}y\right) \cdot \exp\left(-\sqrt{\frac{n}{2v}}iy\right)$$

Problem

$$f(y) = \exp\left(-\sqrt{\frac{n}{2v}}y\right) \cdot \exp\left(-\sqrt{\frac{n}{2v}}iy\right)$$

$$u = U_0 Re\left(e^{i\omega t} f(y)\right)$$

$$= U_0 \exp\left(-\sqrt{\frac{n}{2v}}y\right) Re\left(\exp\left(\left(nt - \sqrt{\frac{n}{2v}}y\right)i\right)\right)$$

$$\boxed{u = U_0 \exp\left(-\sqrt{\frac{n}{2v}}y\right) \cos\left(nt - \sqrt{\frac{n}{2v}}y\right)}$$

Problem

2. (Numerical analysis)

무한하게 펼쳐진 두 개의 평판이 각 각 y=0과 y=L에 위치한다고 가정한다. 바닥에 있는 평판(y=0)은 $u(0,t) = \cos(nt)$ 로 진동하는 반면에, 위에 있는 평판(y=L)은 고정되어있다. 주어진 조건 하에서 Stokes second problem의 지배방정식을 사용하여 다음 조건 하에 속도profile u(y,t)를 구하시오. $\nu = 1, n = 2, U_0 = 1,$ 그리고 $L = 10.$

(1) 주어진 방정식을 first-order forward difference in time 그리고 second-order central difference in space(FTCS scheme)으로 계산하시오. 속도 profile은 $nt = 0, \frac{\pi}{2}, \pi, 3\pi/2, 2\pi$ 일 때에 대하여 그리시오. 또한 quasi-steady state velocity profile을 $(nt - T) = 0, \frac{\pi}{2}, \pi, 3\pi/2, 2\pi$ 에 대하여 구하시오. T는 일종의 quasi-steady state 해를 얻기 위한 transient period로서 $T = 10\pi$ 로 주어진다.

Problem

(1) 주어진 방정식을 first-order forward difference in time 그리고 second-order central difference in space(FTCS scheme)으로 계산하시오. 속도 profile은 $\mathrm{nt} = 0, \frac{\pi}{2}, \pi, 3\pi/2, 2\pi$ 일 때에 대하여 그리시오. 또한 quasi-steady state velocity profile을 $(\mathrm{nt} - \mathrm{T}) = 0, \frac{\pi}{2}, \pi, 3\pi/2, 2\pi$ 에 대하여 구하시오. T는 일종의 quasi-steady state 해를 얻기 위한 transient period로서 $\mathrm{T} = 10\pi$ 로 주어진다.

FTCS

$$\frac{\partial u}{\partial t} = \nu\left(\frac{\partial^2 u}{\partial y^2}\right)$$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \upsilon\,\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta y^2}$$

$$u_i^{n+1} = u_i^n + \frac{\upsilon \cdot \Delta t}{\Delta y^2}\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)$$

Problem

$$u_i^{n+1} = u_i^n + \frac{v \cdot \Delta t}{\Delta y^2}\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)$$

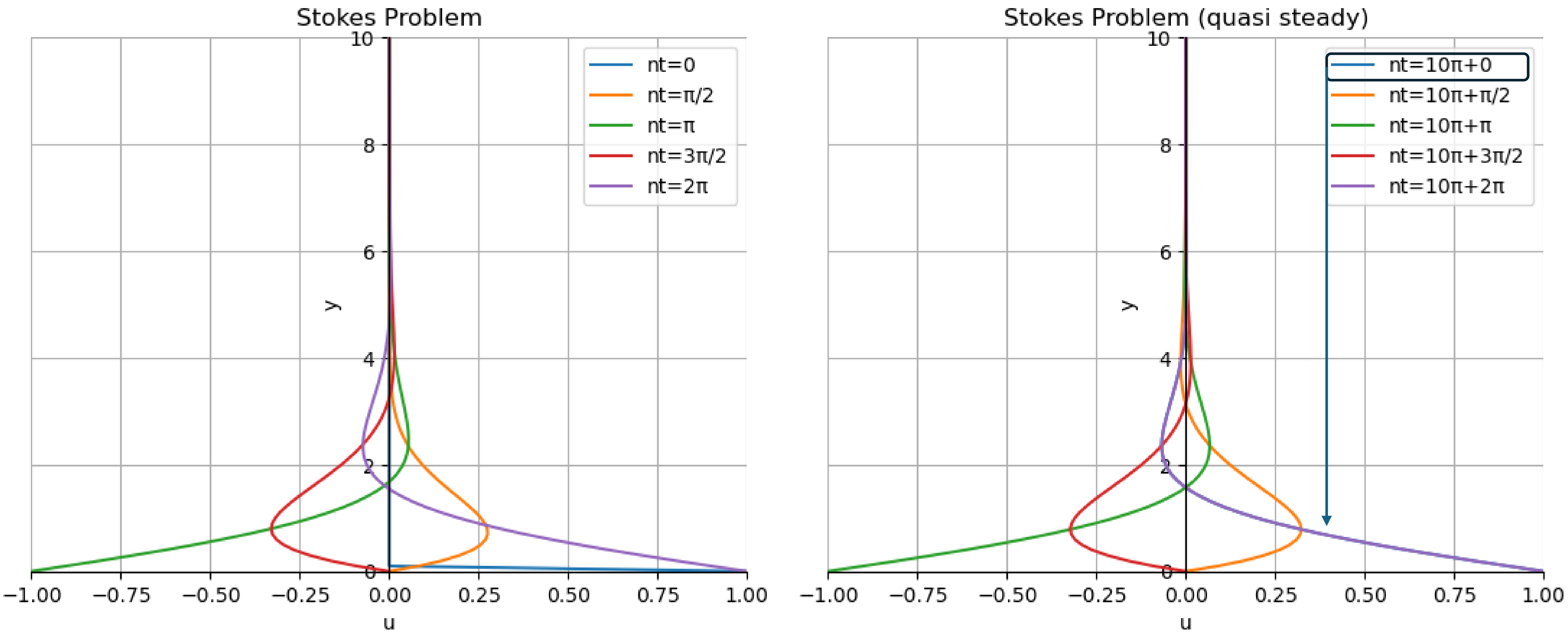
```
v = 1
n = 2
U = 1
L = 10
dy = 0.1
dt = 0.001 * np.pi
t = 0
y_list = np.linspace(0, L, int(L/dy) + 1)
t_list = [0]
u_list = np.zeros(int(L/dy) + 1)
u_list[0] = U * np.cos(n * t)  # Initial condition at t=0

u_t_list_FTCS =[u_list.copy()]
```

Problem

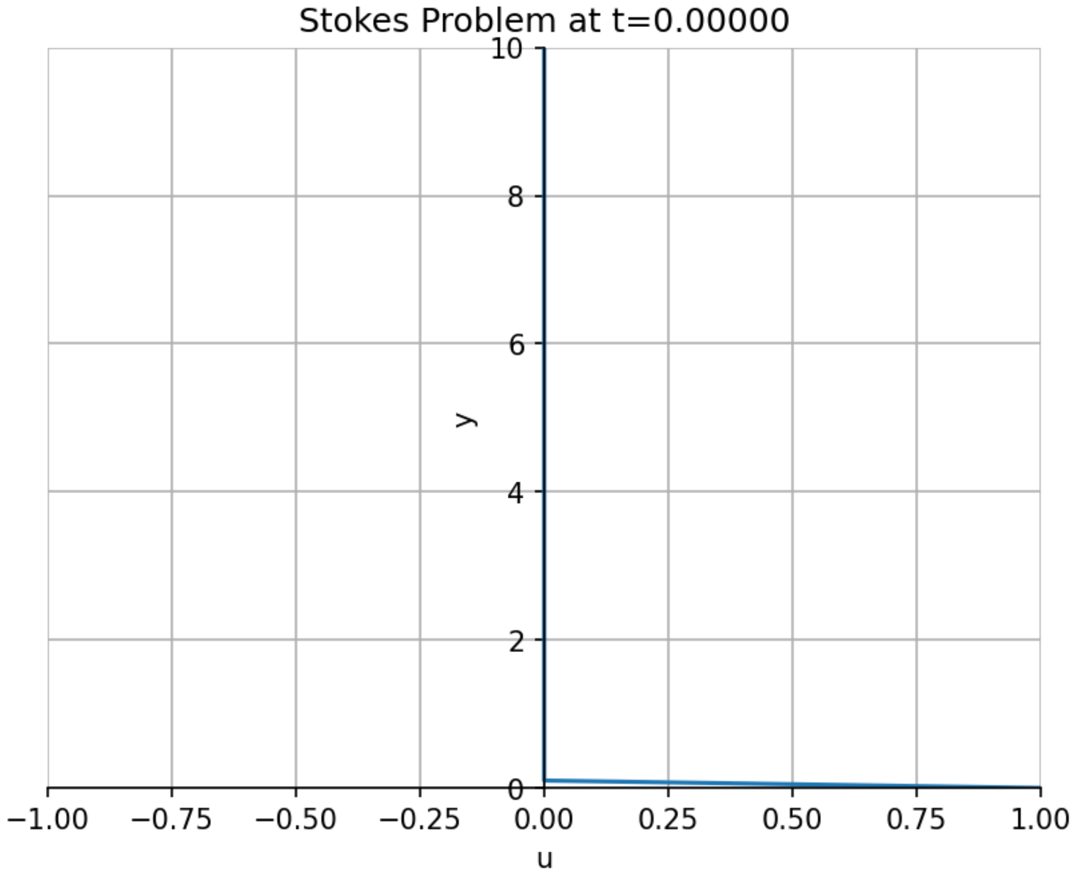
$$u_i^{n+1} = u_i^n + \frac{\upsilon \cdot \Delta t}{\Delta y^2}\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)$$

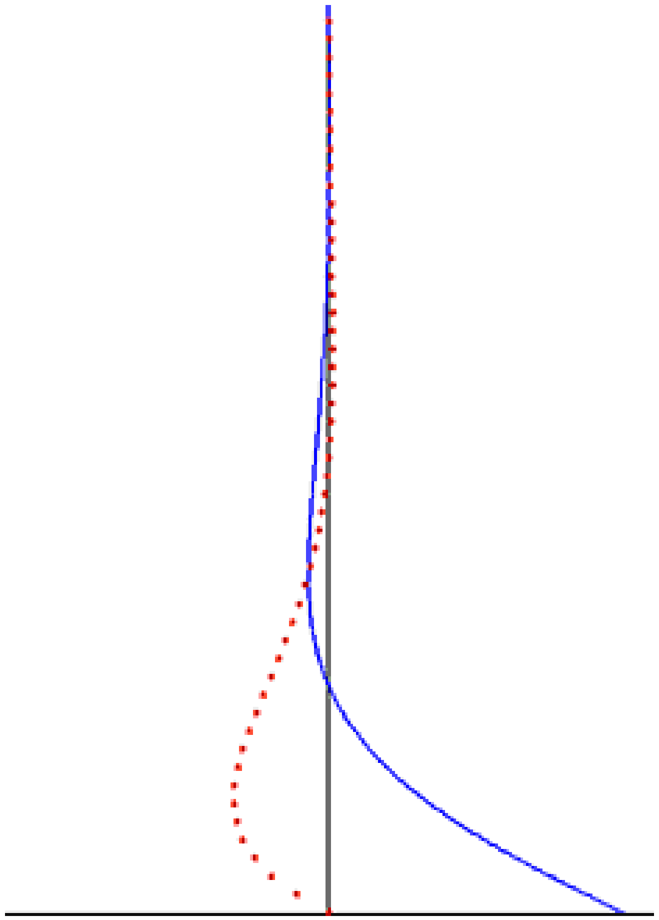
```
for j in range(2000):

    u_list_new = np.zeros(int(L/dy) + 1)
    u_list_new[0] = U * np.cos(n * (t+dt))
    u_list_new[-1] = 0

    t += dt

    for i in range(1,len(u_list)-1):
        u_new = v*((u_list[i+1] - 2*u_list[i] + u_list[i-1]) / dy**2) * dt + u_list[i]

        u_list_new[i] = u_new

    u_list = u_list_new

    t_list.append(t)
    u_t_list_FTCS.append(u_list_new.copy())
```

Problem


Stokes Problem
nt=0
nt=π/2
nt=π
nt=3π/2
nt=2π
y
u
Stokes Problem (quasi steady)
nt=10π+0
nt=10π+π/2
nt=10π+π
nt=10π+3π/2
nt=10π+2π
y
u

Problem


Stokes Problem at t=0.00000
10
8
6
4
2
0
y
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
u



Problem

(2) 위 문제를 시간에 대하여 Crank-Nicolson scheme을 사용하여 계산하시오

Crank-Nicolson Scheme

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \upsilon \frac{(u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}) + (u_{i+1}^n - 2u_i^n + u_{i-1}^n)}{2(\Delta y)^2}$$

$$r = \frac{\upsilon \Delta t}{2(\Delta y)^2}$$

$$-ru_{i+1}^{n+1} + (1 + 2r)u_i^{n+1} - ru_{i-1}^{n+1} = ru_{i+1}^n + (1 - 2r)u_i^n + ru_{i-1}^n$$

Problem

(2) 위 문제를 시간에 대하여 Crank-Nicolson scheme을 사용하여 계산하시오

Crank-Nicolson Scheme

$$-ru_{i+1}^{n+1} + (1+2r)u_i^{n+1} - ru_{i-1}^{n+1} = ru_{i+1}^{n} + (1-2r)u_i^{n} + ru_{i-1}^{n}$$

$$\begin{pmatrix} 1+2r & -r & 0 & \cdots & & 0 \\ -r & 1+2r & -r & 0 & \cdots & 0 \\ & \ddots & \ddots & \ddots & & \\ 0 & \cdots & & -r & 1+2r & -r \\ 0 & \cdots & \cdots & 0 & -r & 1+2r \end{pmatrix} \begin{pmatrix} u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{i-2}^{n+1} \\ u_{i-1}^{n+1} \end{pmatrix} = \begin{pmatrix} ru_3^n + (1-2r)u_2^n + ru_1^n \\ ru_4^n + (1-2r)u_3^n + ru_2^n \\ \vdots \\ ru_i^n + (1-2r)u_{i-1}^n + ru_{i-2}^n \end{pmatrix} + \begin{pmatrix} ru_1^{n+1} \\ 0 \\ \vdots \\ 0 \\ ru_i^{n+1} \end{pmatrix}$$

Problem

(2) 위 문제를 시간에 대하여 Crank-Nicolson scheme을 사용하여 계산하시오

Crank-Nicolson Scheme

```python
for j in range(12000):
    t = t+ dt
    u_list_old = np.zeros((int(L/dy)-1))
    r_list = np.zeros((int(L/dy)-1,int(L/dy)-1))

    for i in range(int(L/dy)-1):
        if i == 0:
            r_list[i,i] = 1 + 2*r
            r_list[i,i+1] = -r

        elif i == int(L/dy)-2:
            r_list[i,i-1] = -r
            r_list[i,i] = 1 + 2*r

        else:
            r_list[i,i-1] = -r
            r_list[i,i] = 1 + 2*r
            r_list[i,i+1] = -r
```

$$
\begin{pmatrix}
1+2r & -r & 0 & \cdots & & 0 \\
-r & 1+2r & -r & 0 & \cdots & 0 \\
 & \ddots & & & & \\
0 & \cdots & -r & 1+2r & -r & \\
0 & \cdots & \cdots & 0 & -r & 1+2r
\end{pmatrix}
\begin{pmatrix}
u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{i-2}^{n+1} \\ u_{i-1}^{n+1}
\end{pmatrix}
$$

Problem

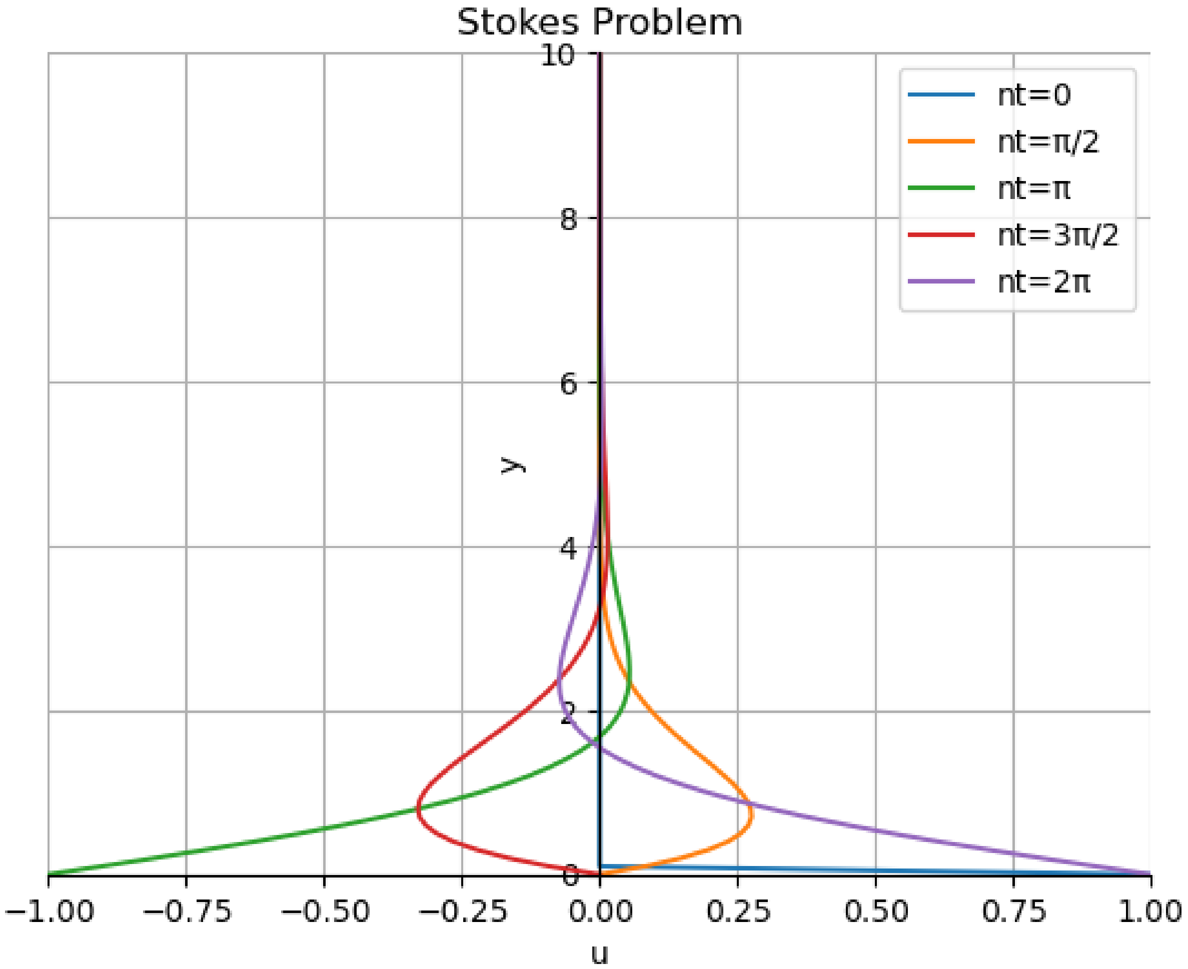
(2) 위 문제를 시간에 대하여 Crank-Nicolson scheme을 사용하여 계산하시오

Crank-Nicolson Scheme

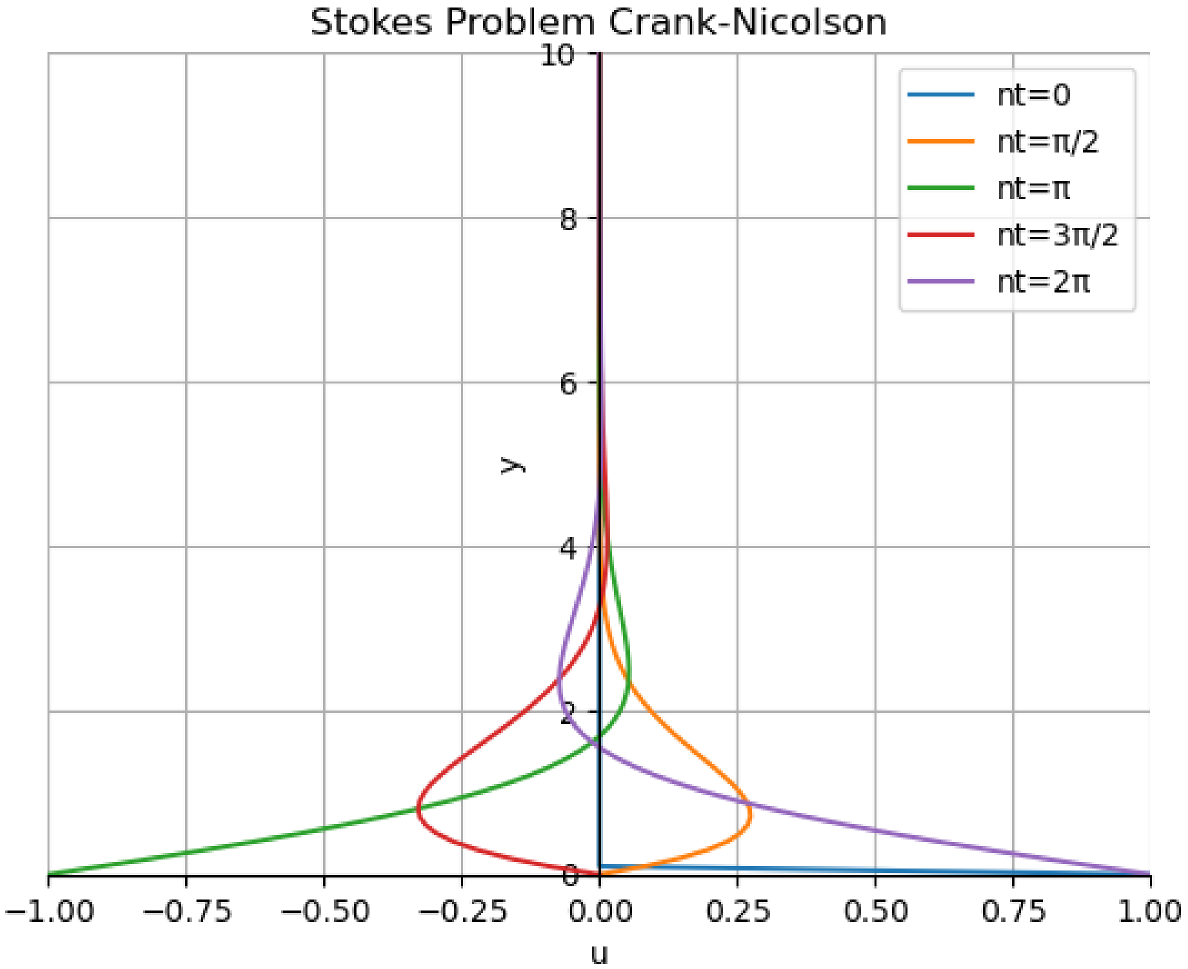
```
for i in range(int(L/dy)-1):
    u_list_old[i] = r * u_t_list_CN[j][i] + (1 - 2*r) * u_t_list_CN[j][i+1] + r * u_t_list_CN[j][i+2]

u_list_old[0] = u_list_old[0] + r* U * np.cos(n * t)

u_list_new = np.linalg.solve(r_list, u_list_old)

# boundary condition
u_list_new = np.insert(u_list_new, 0, U * np.cos(n * t))
u_list_new = np.append(u_list_new, 0)

u_t_list_CN.append(u_list_new.copy())
t_list.append(t)
```

$$\begin{pmatrix} r u_3^n + (1-2r) u_2^n + r u_1^n \\ r u_4^n + (1-2r) u_3^n + r u_2^n \\ \vdots \\ r u_i^n + (1-2r) u_{i-1}^n + r u_{j-2}^n \end{pmatrix} + \begin{pmatrix} r u_1^{n+1} \\ 0 \\ \vdots \\ 0 \\ r u_i^{n+1} \end{pmatrix}$$

Problem

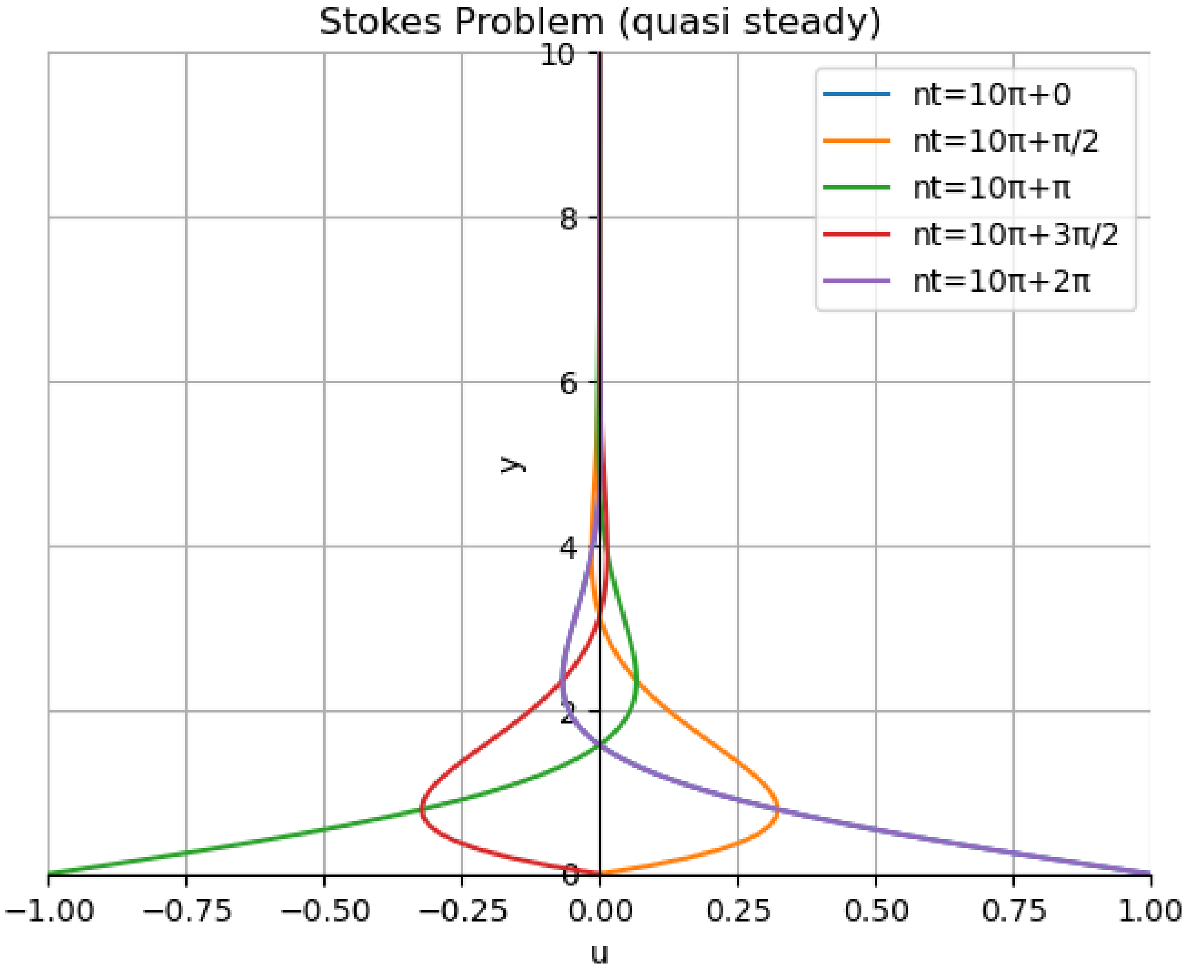


FTCS

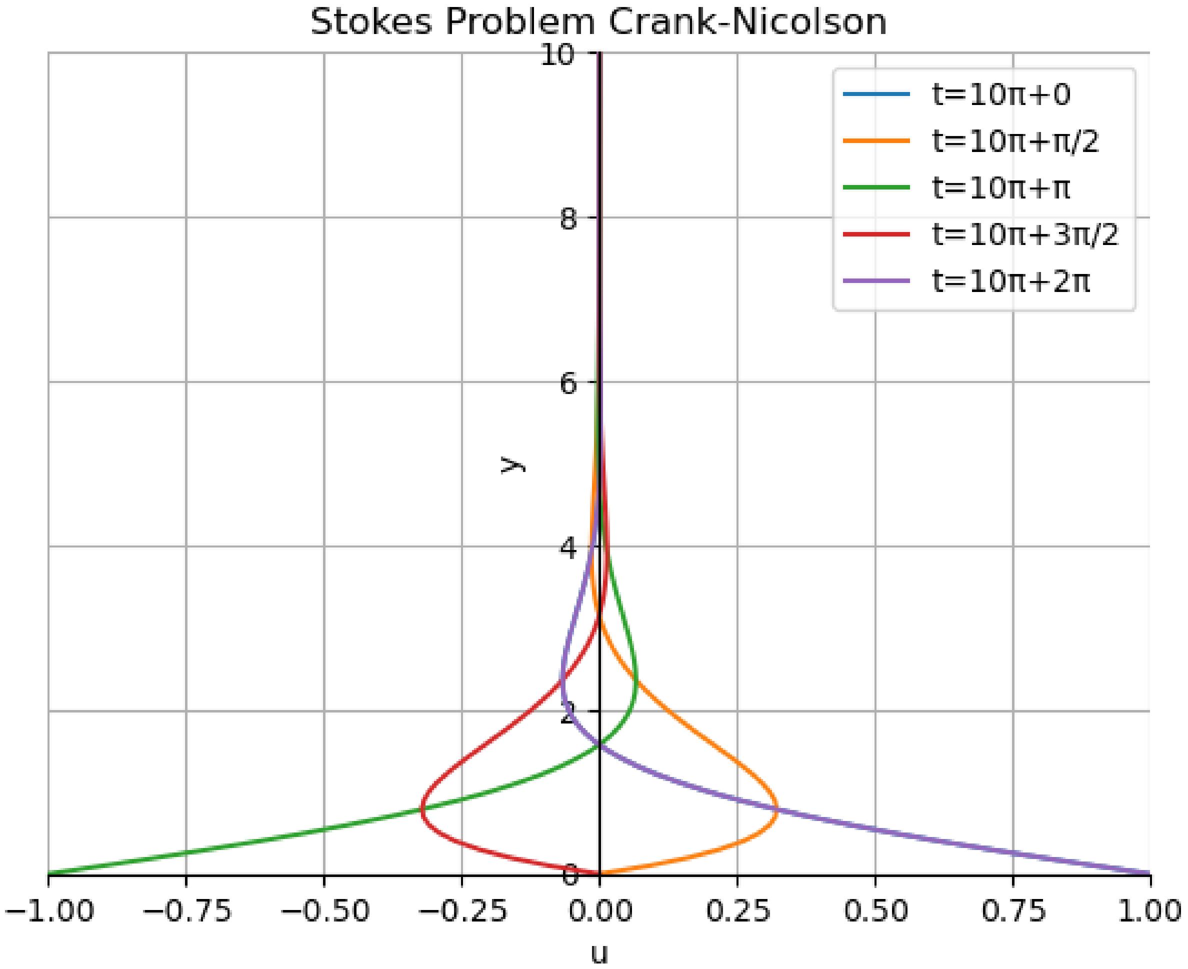


Crank-Nicolson

Problem



FTCS



Crank-Nicolson

Problem

(3) 각각 다른 두개의 scheme에 대하여 시간의 변화에 따른 해의 수렴율을 구하시오 (log(Δt) vs log (l2norm)). FTCS는 시간에 대한 1차, Crank-Nicolson은 2차의 수렴율을 보여야 하며, 오차계산에 사용되는 exact solution은

$$u(y,t) = U_0 e^{-\eta_s} \cos(nt - \eta_s)\,, where\ \eta_s = \sqrt{\frac{n}{2v}}y.$$

을 사용하여 구하시오.

```python
def u_exact(t, y):
    n_s = (n/(v*2))**0.5*y
    u = U * np.exp(-n_s)*np.cos(n*t - n_s)
    return u
```

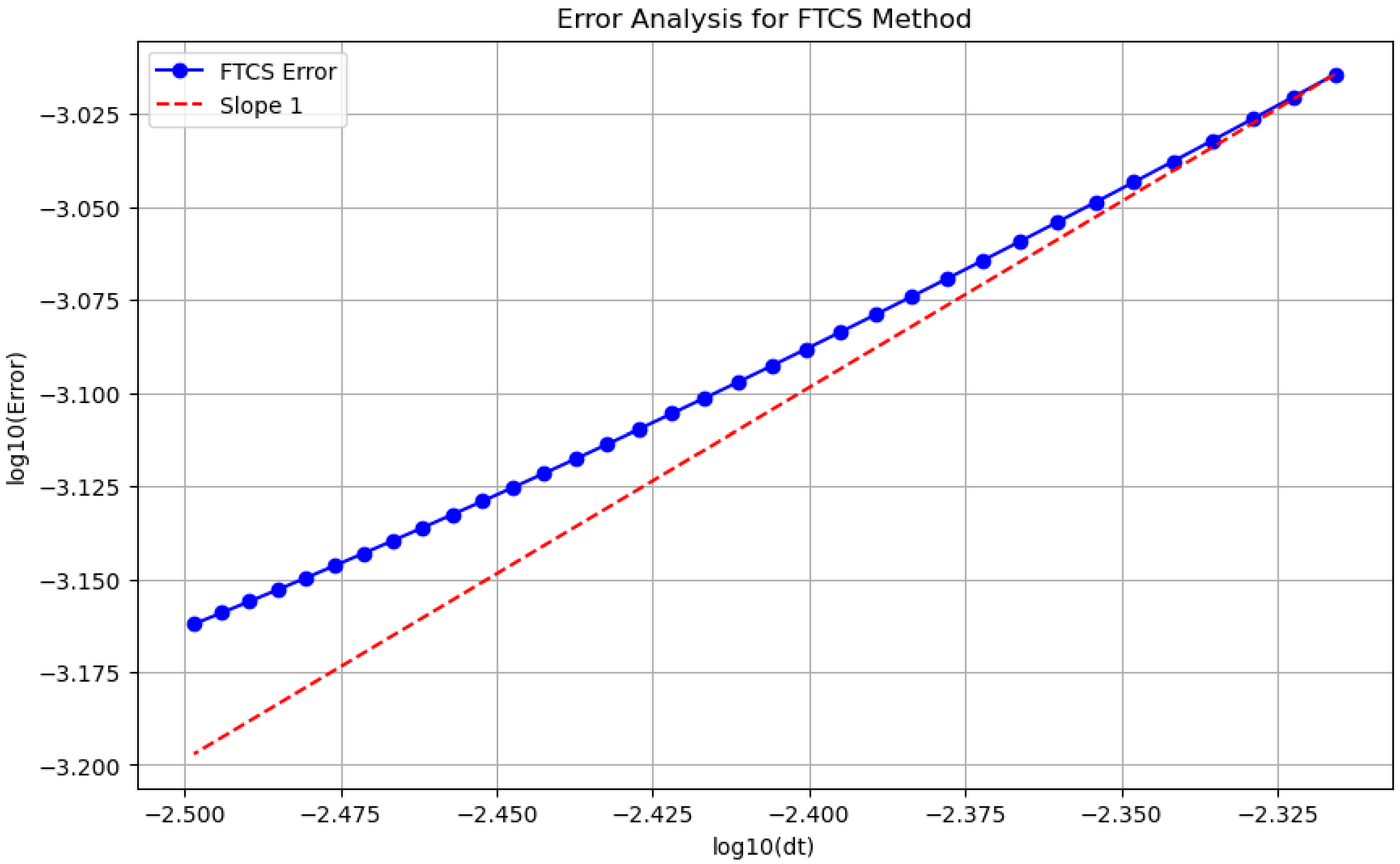
Problem(FTCS)

```
error_FTCS = []
dt_list = []
for k in range(650, 1000, 10):
    v = 1
    n = 2
    U = 1
    L = 10
    dy = 0.1
    dt = np.pi/k
    t = 0
    y_list = np.linspace(0, L, int(L/dy) + 1)
    t_list = [0]
    u_list = np.zeros(int(L/dy) + 1)
    u_list[0] = U * np.cos(n * t)  # Initial condition at t=0

    u_t_list_FTCS =[u_list.copy()]

    u_exact_list = [u_exact(0, y_list)]
```

Problem(FTCS)

```
for j in range(int(6*k)):   nt = 12π

    u_list_new = np.zeros(int(L/dy) + 1)
    u_list_new[0] = U * np.cos(n * (t+dt))
    u_list_new[-1] = 0

    t += dt

    for i in range(1,len(u_list)-1):
        u_new = v*((u_list[i+1] - 2*u_list[i] + u_list[i-1]) / dy**2) * dt + u_list[i]

        u_list_new[i] = u_new

    u_list = u_list_new

    u_t_list_FTCS.append(u_list_new.copy())
    u = u_exact(t, y_list)
    u_exact_list.append(u.copy())

error_FTCS.append(np.linalg.norm(u_t_list_FTCS[int(6*k)] - u_exact_list[int(6*k)], 2)* np.sqrt(dy))
dt_list.append(dt)
```

Problem(FTCS)

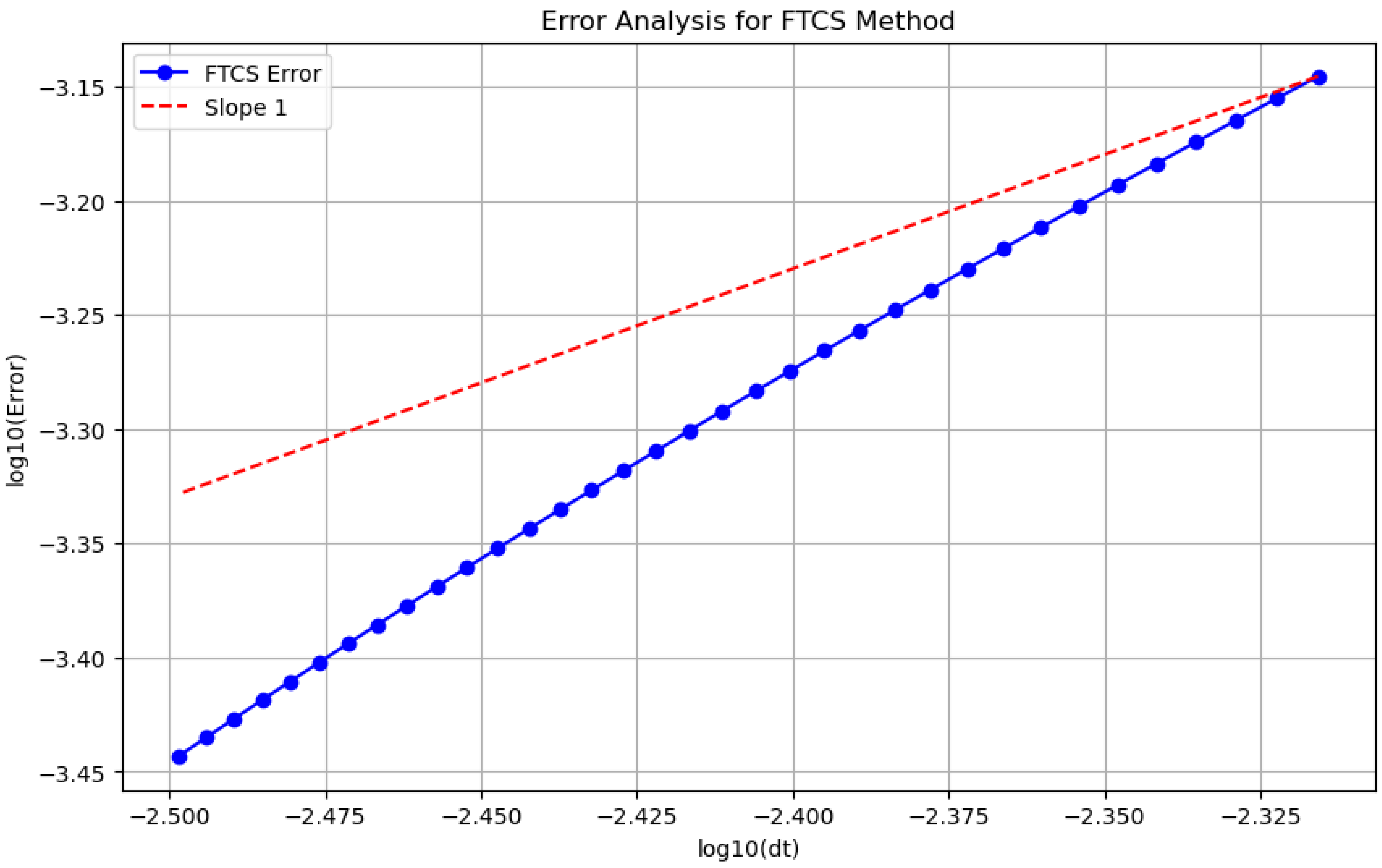


Problem(FTCS)

$nt = 24\pi$

```python
for j in range(int(12*k)):

    u_list_new = np.zeros(int(L/dy) + 1)
    u_list_new[0] = U * np.cos(n * (t+dt))
    u_list_new[-1] = 0

    t += dt

    for i in range(1,len(u_list)-1):
        u_new = v*((u_list[i+1] - 2*u_list[i] + u_list[i-1]) / dy**2) * dt + u_list[i]

        u_list_new[i] = u_new

    u_list = u_list_new

    u_t_list_FTCS.append(u_list_new.copy())
    u = u_exact(t, y_list)
    u_exact_list.append(u.copy())

error_FTCS.append(np.linalg.norm(u_t_list_FTCS[int(12*k)] - u_exact_list[int(12*k)], 2)* np.sqrt(dy))
dt_list.append(dt)
```

Problem(FTCS)



?

Problem(FTCS)

추측

Local truncation error

$$\tau(x,t) = \frac{u(x,t+k) - u(x,t)}{k} - \frac{1}{h^2}(u(x-h,t) - 2u(x,t) + u(x+h,t)).$$

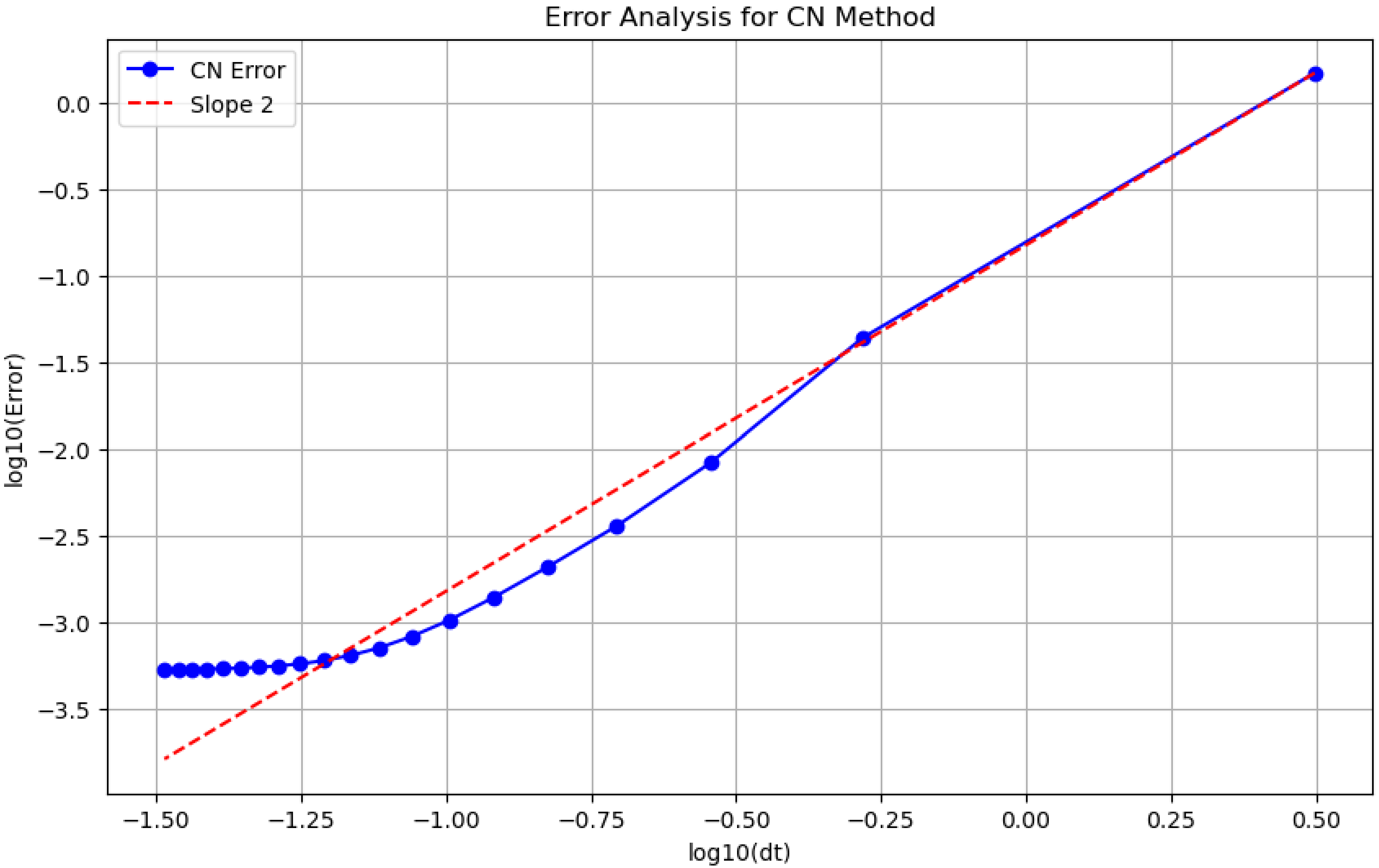
$$\tau(x,t) = \left(u_t + \frac{1}{2}ku_{tt} + \frac{1}{6}k^2u_{ttt} + \cdots\right) - \left(u_{xx} + \frac{1}{12}h^2u_{xxxx} + \cdots\right).$$

Global truncation error

$$\begin{aligned} e_n &= y(t_n) - y_n \\ &= y(t_n) - \Big(y_0 + hA(t_0, y_0, h, f) + hA(t_1, y_1, h, f) + \cdots + hA(t_{n-1}, y_{n-1}, h, f)\Big). \end{aligned}$$

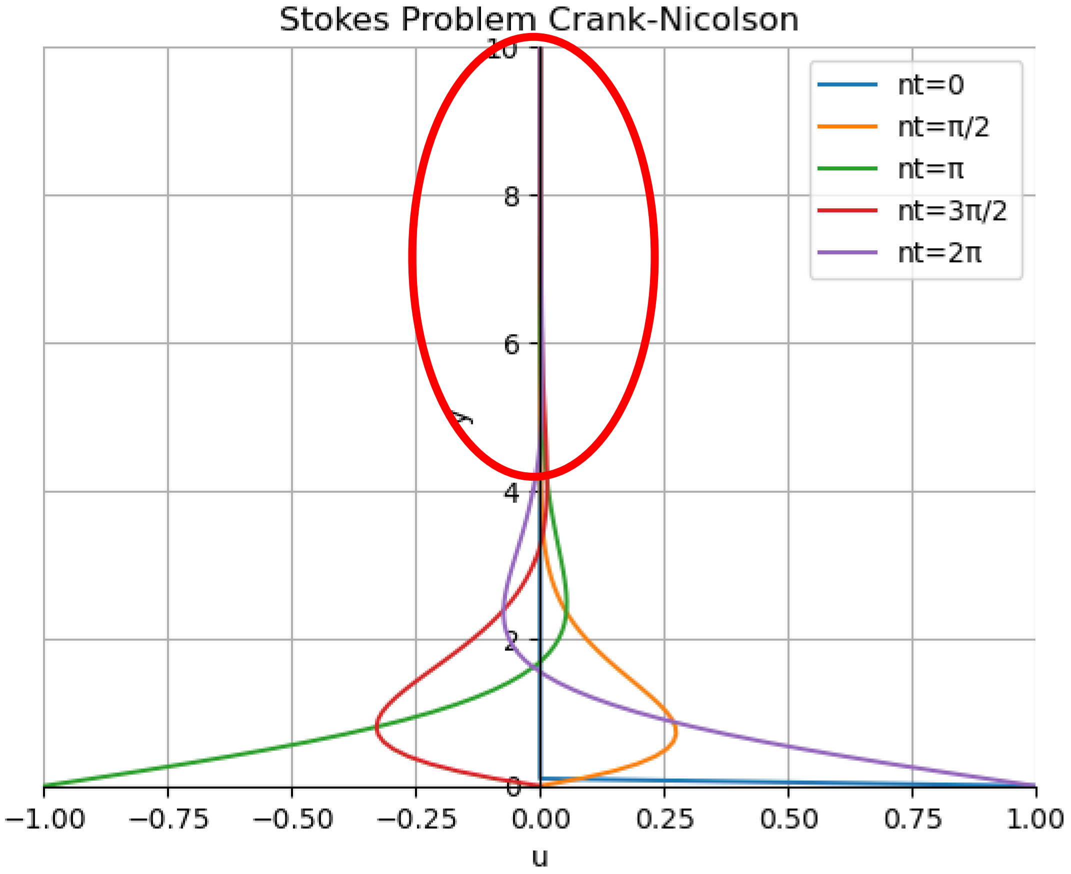
Local truncation error의 차수가 1차이므로 Global truncation error의 차수가 1이 아님

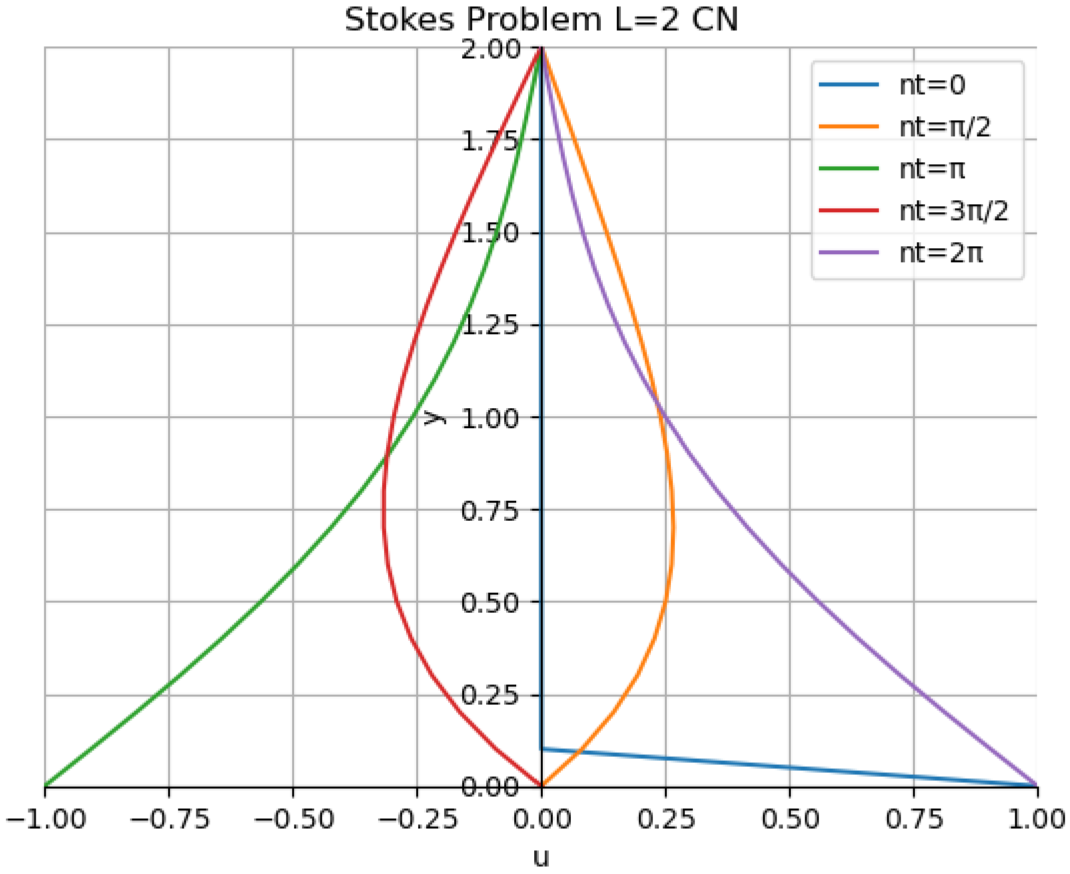
Problem(Crank-Nicolson)



Problem(Crank-Nicolson)

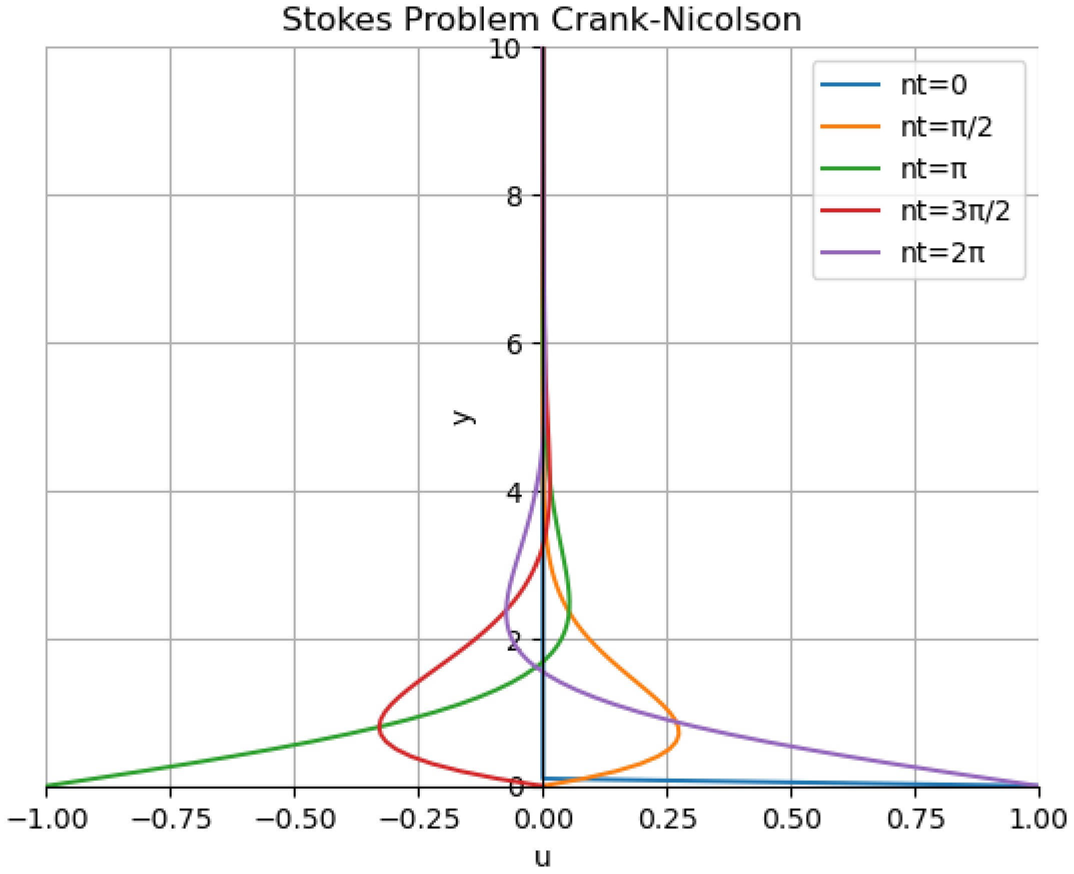
(4) (2)번 문제를 L=2의 조건에 대하여 다시 진행하고, 두 평판의 거리가 속도 profile에 미치는 영향을 서술하시오.

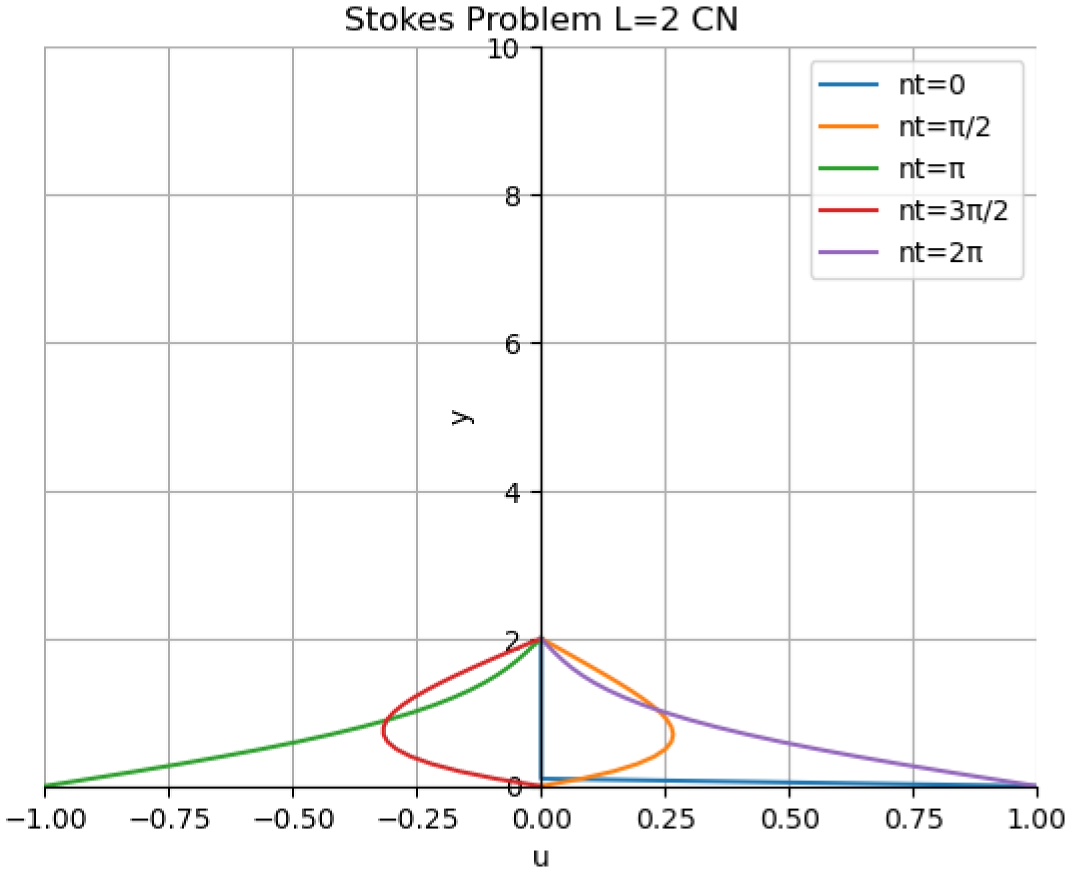




Problem(Crank-Nicolson)

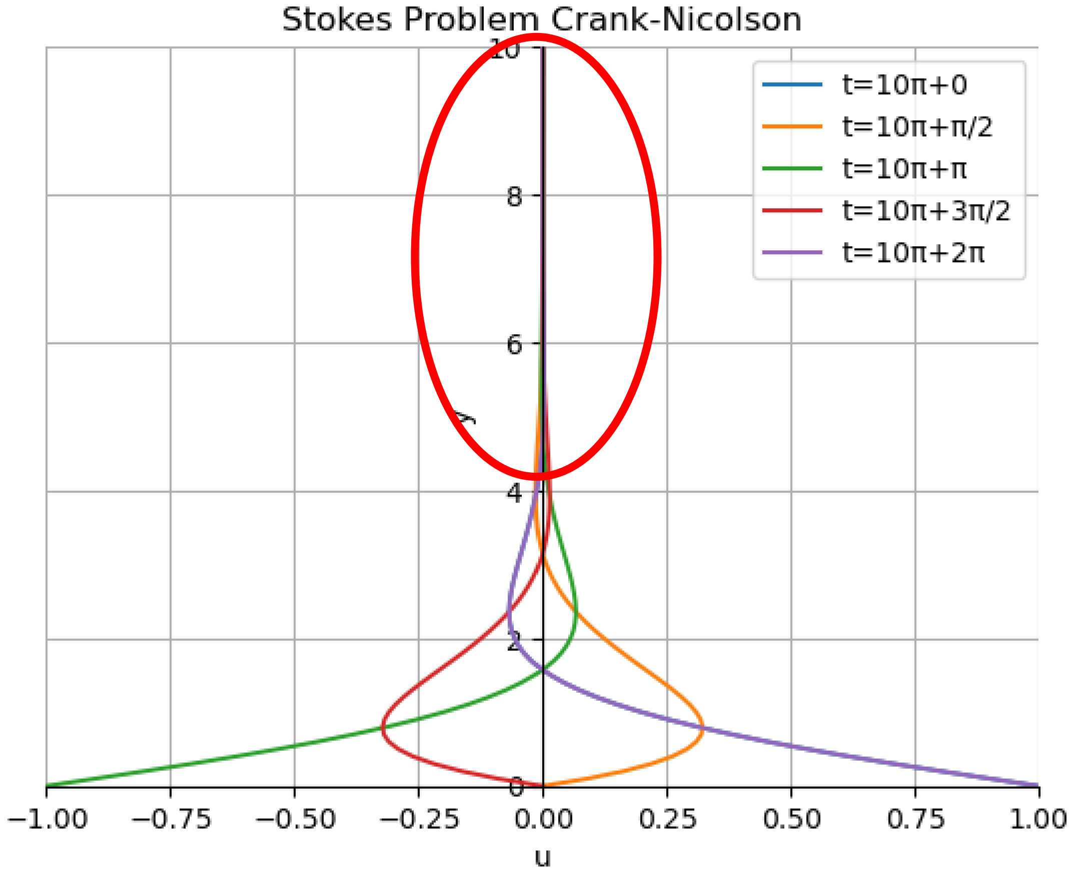
(4) (2)번 문제를 L=2의 조건에 대하여 다시 진행하고, 두 평판의 거리가 속도 profile에 미치는 영향을 서술하시오.

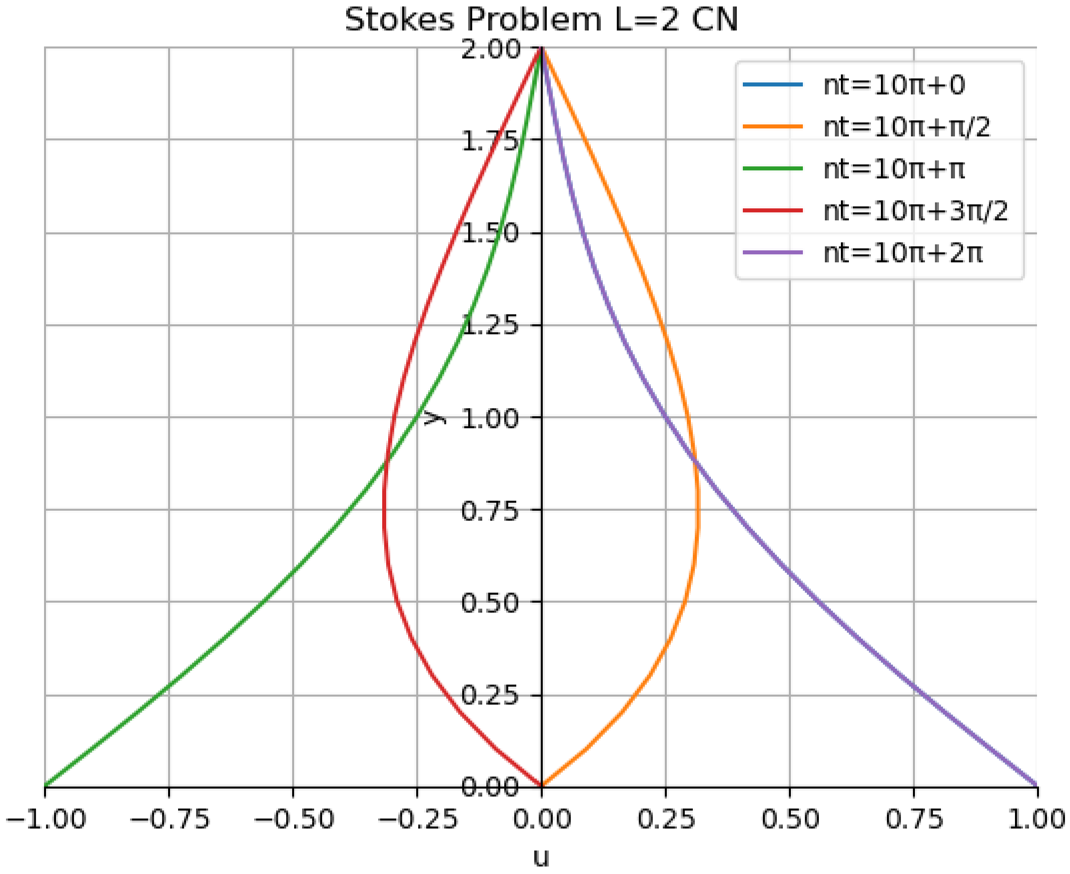




Problem(Crank-Nicolson)

(4) (2)번 문제를 L=2의 조건에 대하여 다시 진행하고, 두 평판의 거리가 속도 profile에 미치는 영향을 서술하시오.





Problem(Crank-Nicolson)

(4) (2)번 문제를 L=2의 조건에 대하여 다시 진행하고, 두 평판의 거리가 속도 profile에 미치는 영향을 서술하시오.