```
# -*- coding: utf-8 -*-
"""
Created on Tue Sep 30 15:37:24 2025

@author: coled
"""

# ICF HW6 Problem 3

import numpy as np
import matplotlib.pyplot as plt


def dm_dx(M, f, x, gamma=1.4):

    c1 = (1 + (gamma-1)*0.5 * M **2) / ( (M**2 - 1))
    c2 = (gamma*M**2) * f / ( np.exp(1/(x+1)))
    c3 = -2 / ((x+1)**2)
    return M * c1 * (c3-c2)

# grid (meters)
dx = 1.0e-3  # m
xmin = 0.0
xmax = 200.0
n_samples = int((xmax - xmin)/dx)

x = np.linspace(xmin, xmax, n_samples)
M = np.zeros_like(x)
M[0] = 6.0

f = 0.005
tol = 1e-3

def find_choke(M1=M, x1=x, f1=f):
    for i in range(len(x) - 1):
        M[i+1] = M[i] + dx * dm_dx(M[i], f, x[i])
    return (x[i], M)

L1, M_array = find_choke()


T0 = 600.0          # K
p0 = 200.0e3        # Pa
R  = 287.0          # J/(kg*K)

x2 = x

u   = np.zeros_like(x2)
A   = np.zeros_like(x2)
T   = np.zeros_like(x2)
rho = np.zeros_like(x2)
p   = np.zeros_like(x2)
p0_array = np.zeros_like(x2)
T0_array = np.zeros_like(x2)
time = 0.0
```

```
def pressure_isen(M, gamma=1.4):   # p/p0
    return (1.0 + 0.5*(gamma-1.0)*M**2) ** (-gamma/(gamma-1.0))

def flow_property_functions(M, T0=T0, p0=p0, R=R, gamma=1.4):
    c1 = 1.0 + 0.5*(gamma-1.0)*M**2
    T = T0 / c1
    u = M * np.sqrt(gamma*R*T)
    return {"temp": T, "u": u}

# mass-flow constant from inlet
props = flow_property_functions(M_array[0])
A_init   = np.pi * np.exp(2.0)
u_init   = props["u"]
p_init   = p0 * pressure_isen(M_array[0])
T_init   = props["temp"]
rho_init = p_init / (R * T_init)
rho_const = rho_init * u_init * A_init

# fields + residence time
for i in range(len(M_array)):
    props = flow_property_functions(M_array[i])
    T[i] = props["temp"]
    T0_array[i] = T0
    u[i] = props["u"]
    A[i] = np.pi * np.exp(2.0 / (x2[i] + 1.0))
    rho[i] = rho_const / (u[i] * A[i])
    p[i] = rho[i] * R * T[i]
    p0_array[i] = p[i] / pressure_isen(M_array[i])
    time += dx / u[i]

# plots
fig1, axs = plt.subplots(2, 3, figsize=(11, 6), constrained_layout=True)

axs[0,0].plot(x2, M_array);        axs[0,0].set_title('Mach Number');          axs[0,0].set_xlabel(
axs[0,1].plot(x2, p);              axs[0,1].set_title('Pressure [Pa]');        axs[0,1].set_xlabel(
axs[0,2].plot(x2, p0_array);       axs[0,2].set_title('Stagnation Pressure');  axs[0,2].set_xlabel(
axs[1,0].plot(x2, T0_array);       axs[1,0].set_title('Stagnation T [K]');     axs[1,0].set_xlabel(
axs[1,1].plot(x2, T);              axs[1,1].set_title('Static T [K]');         axs[1,1].set_xlabel(

plt.show()
```