

```

# -*- coding: utf-8 -*-
"""
Created on Mon Oct 20 12:41:46 2025

@author: coled_agkeohi
"""

# MM1 HW8 Problem 2
# 2.2: Projection of  $\sin(4t)$  onto  $t, t^2, \tan(t)$  space in  $L_2$ 

import numpy as np
from numpy.polynomial.legendre import leggauss
import matplotlib.pyplot as plt

# ---- inner product on  $L_2[0,1]$  via Gauss-Legendre quadrature ----
def inner(f, g, N=200):
    x, w = leggauss(N)          # nodes/weights on  $[-1,1]$ 
    t = 0.5*(x + 1.0)          # map to  $[0,1]$ 
    wt = 0.5*w
    return np.sum(wt * f(t) * g(t))

# ---- target and bases ----
x = lambda t: np.sin(4*t)

# Part A basis:  $\{t, t^2\}$ 
u1 = lambda t: t
u2 = lambda t: t**2
U1 = [u1, u2]

# Part B basis:  $\{t, t^2, \tan t\}$ 
u3 = lambda t: np.tan(t)        # finite on  $[0,1]$ 
U2 = [u1, u2, u3]

def projection_coeffs(U):
    m = len(U)
    G = np.empty((m, m))
    b = np.empty(m)
    for i, ui in enumerate(U):
        b[i] = inner(x, ui)
        for j, uj in enumerate(U):
            G[i, j] = inner(ui, uj)
    # <-- solve AFTER filling G and b
    c = np.linalg.solve(G, b)
    return c, G, b

# ---- compute coefficients ----
c1, G1, b1 = projection_coeffs(U1)
c2, G2, b2 = projection_coeffs(U2)

def p(t, c, U):
    return sum(ci * ui(t) for ci, ui in zip(c, U))

print("Coefficients part 2 ( $t, t^2, \tan$ ):", c2)

# ---- Taylor expansions about  $t_0 = 0.5$  ----

```

```

t0 = 0.5
si, co = np.sin(2.0), np.cos(2.0)

def T2(t):
    dt = t - t0
    return si + 4.0*co*dt - 8.0*si*dt**2

def T3(t):
    dt = t - t0
    return si + 4.0*co*dt - 8.0*si*dt**2 - (32.0/3.0)*co*dt**3 # <-- dt**3

# ---- plot ----
tt = np.linspace(0.0, 1.0, 600)
p1 = p(tt, c1, U1)
p2 = p(tt, c2, U2)

plt.figure(figsize=(7,5))
plt.plot(tt, x(tt), label='x(t)=sin(4t)')
plt.plot(tt, p1, label='Projection on {t, t^2}')
plt.plot(tt, p2, label='Projection on {t, t^2, tan t}')
plt.plot(tt, T2(tt), label='Quadratic Taylor @0.5')
plt.plot(tt, T3(tt), label='Cubic Taylor @0.5')
plt.xlabel('t'); plt.ylabel('value'); plt.title('Projection & Taylor Approximations on [0,1]')
plt.grid(True); plt.legend()
plt.show()

```