Solver for Delayed Timoshenko Beam Transmission Problem

1. Problem Formulation

We solve a Timoshenko beam transmission problem on two subdomains $I_1 = (0, L_0)$ and $I_2 = (L_0, L)$, with transverse displacement $\phi_i(x,t)$ and shear angle $\psi_i(x,t)$, $i = 1, 2$, together with an auxiliary transport–delay variable $z(x,t)$ on I_1.

1.1 Governing PDEs on I_i

For $i = 1, 2$, and $x \in I_i$,

$$\rho_i \, \phi_{i,tt} - \kappa_i \, (\phi_{i,x} + \psi_i)_x + f_1(\phi_i, \psi_i) = g_i(x), \tag{1}$$
$$\sigma_i \, \psi_{i,tt} - \lambda_i \, \psi_{i,xx} + \kappa_i \, (\phi_{i,x} + \psi_i) + f_2(\phi_i, \psi_i) = h_i(x). \tag{2}$$

The nonlinear terms are

$$f_1(\phi, \psi) = 4(\phi + \psi)^3 - 2(\phi + \psi) + 2\phi \, \psi^2, \qquad f_2(\phi, \psi) = 4(\phi + \psi)^3 - 2(\phi + \psi) + 2\phi^2 \, \psi.$$

The (time-independent) load functions are

$$g_1(x) = \sin x, \; h_1(x) = x, \quad g_2(x) = \cos x, \; h_2(x) = x + 1.$$

1.2 Transport–Delay PDE on I_1

The auxiliary variable $z(x,t)$ satisfies

$$\tau \, z_t + z_x = 0, \quad x \in [0, L_0].$$

Its boundary values couple to ψ_1:

$$z(0,t) = \psi_{1,t}(0,t), \qquad z(L_0,t) = \psi_{1,t}\big(0, t - \tau L_0\big).$$

1.3 Transmission Conditions at $x = L_0$

Continuity of fields:

$$\phi_1(L_0,t) = \phi_2(L_0,t), \quad \psi_1(L_0,t) = \psi_2(L_0,t).$$

Continuity of fluxes:

$$\kappa_1(\phi_{1,x} + \psi_1) = \kappa_2(\phi_{2,x} + \psi_2), \quad \lambda_1 \psi_{1,x} = \lambda_2 \psi_{2,x}, \quad x = L_0.$$

1.4 Boundary Conditions with Delay

At $x = 0$ (left end of I_1):

$$\phi_1(0,t) = 0, \quad \lambda_1 \psi_{1,x}(0,t) = \gamma \, z(0,t) + \mu \, z(L_0,t).$$

At $x = L$ (right end of I_2):

$$\phi_2(L,t) = 0, \quad \psi_2(0,t) = 0.$$

1.5 Initial Conditions

For x in each subdomain,

$$\begin{aligned}
&\phi_i(x,0) = \phi_i^0(x), \quad \phi_{i,t}(x,0) = \phi_i^1(x),\\
&\psi_i(x,0) = \psi_i^0(x), \quad \psi_{i,t}(x,0) = \psi_i^1(x),\\
&z(x,0) = z_0(x) = \tfrac{13}{4}x + 1.
\end{aligned}$$

1.6 Parameter Values

The constants are set to:

$\gamma = 1$, $\rho_1 = 1, \rho_2 = 2$, $\sigma_1 = 2, \sigma_2 = 1$, $\kappa_1 = 4, \kappa_2 = 1$, $\lambda_1 = 8, \lambda_2 = 4$, $L_0 = 4$, $L = 10$, $\mu = \frac{1}{2}$, $\tau = 1$.

2. Code Overview

Each code block is documented with its purpose.

2.1 Imports and Damping

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# small linear damping to stabilize
beta_phi1,beta_psi1=0.01,0.01
beta_phi2,beta_psi2=0.01,0.01
```

Sets up array operations, plotting, and small artificial damping coefficients.

2.2 Parameter and Grid Setup

```
# Physical & delay PDE parameters
gamma=1.0;rho1,rho2=1.0,2.0;sigma1,sigma2=2.0,1.0
kappa1,kappa2=4.0,1.0;lambda1,lambda2=8.0,4.0
L0,L=4.0,10.0;mu=0.5;tau_delay=1.0

# Discretization
T=10.0;n1,n2=50,50;m=2000
dt=T/(m-1);h1=L0/(n1-1);h2=(L-L0)/(n2-1)

t=np.linspace(0,T,m)
x1=np.linspace(0,L0,n1)
x2=np.linspace(L0,L,n2)
delay_steps=max(1,int(tau_delay*L0/dt))
```

2.3 Solution Arrays Allocation

```
phi1=np.zeros((n1,m));psi1=np.zeros((n1,m))
phi2=np.zeros((n2,m));psi2=np.zeros((n2,m))
z   =np.zeros((n1,m))
```

Preallocate fields on each domain and the transport variable z.

2.4 Loads and Nonlinearities

```
g1=lambda x:np.sin(x)
h1_fun=lambda x:x
g2=lambda x:np.cos(x)
h2_fun=lambda x:x+1
f1=lambda phi,psi:4*(phi+psi)**3-2*(phi+psi)+2*phi*psi**2
f2=lambda phi,psi:4*(phi+psi)**3-2*(phi+psi)+2*phi**2*psi
```

Defines external loads and the cubic nonlinear terms.

2.5 Initial Conditions

```
# phi1^0,psi1^0 on [0,L0]
phi1[:,0]=-19/16*x1**2+200/32*x1
psi1[:,0]=x1
# phi2^0 continuous ramp; psi2^0 given
phi2[:,0]=phi1[-1,0]*(L-x2)/(L-L0)
psi2[:,0]=x2**3-(166/9)*x2**2+(914/9)*x2-1540/9

# First time step from phi_i^1,psi_i^1
dphi1_0=x1
psi1_0=10-x1
dphi2_0=(2/3)*(10-x2)
dpsi2_0=(5/6)*(10-x2)
phi1[:,1]=phi1[:,0]+dt*dphi1_0
psi1[:,1]=psi1[:,0]+dt*psi1_0
phi2[:,1]=phi2[:,0]+dt*dphi2_0
psi2[:,1]=psi2[:,0]+dt*dpsi2_0

# z(x,0) and first upwind step
z[:,0]=13/4*x1+1
for j in range(1,n1):
    z[j,1]=z[j,0]-dt/(tau_delay*h1)*(z[j,0]-z[j-1,0])
```

2.6 Time-Stepping Loop

Pseudocode structure:

1. Update interior of I_1 and I_2 with 2nd-order finite differences, explicit in time, plus damping.
2. Upwind transport update for z.
3. Enforce Dirichlet BCs: $\phi_1(0) = 0$, $\phi_2(L) = 0$, $\psi_2(0) = 0$.
4. Transmission at $x = L_0$: average adjacent nodes to enforce continuity.
5. Delayed-feedback BC for $\psi_1(0)$: compute and clamp derivatives, then solve for boundary value.

2.7 Visualization

```
X=np.concatenate((x1,x2))
Phi=np.vstack((phi1,phi2))
```

```
fig=plt.figure()
ax=fig.add_subplot(111,projection='3d')
Tm,Xm=np.meshgrid(t,X)
ax.plot_surface(Tm,Xm,Phi,rstride=10,cstride=10,cmap='viridis')
ax.set(xlabel='t',ylabel='x',zlabel='phi(x,t)')
plt.show()
```

Plots the concatenated solution surface $\phi(x,t)$.