

Heat Equation (FTCS Matrix Method, Dirichlet BCs)

```
import numpy as np import matplotlib.pyplot as plt
```


Parameters

```
alpha = 1.0 Nx = 50 T = 0.1 L, R = 0.0, 0.0 # boundary values
```


Grid and stability

```
dx = 1.0/(Nx-1) dt_max = dx*dx/(2alpha) s = 0.4 dt = s*dt_max Nt = int(T/dt)
dt = T/Nt r = alphadt/dx**2
```


Initial condition

```
x = np.linspace(0.0, 1.0, Nx) u0 = np.sin(np.pi * x)
```


Build matrix A and boundary vector b

```
m = Nx - 2 main = (1 - 2r) np.ones(m) off = r * np.ones(m-1) A = np.diag(main)  
+ np.diag(off,1) + np.diag(off,-1)  
b = np.zeros(m) b[0], b[-1] = rL, rR
```


Time-stepping

```
u = u0.copy() u_in = u[1:-1].copy() snapshots = [] snap_every = max(1, Nt//12)
for j in range(Nt): u_in = A @ u_in + b u[1:-1] = u_in if j % snap_every ==
0 or j == Nt-1: snapshots.append((j*dt, u.copy()))
```


Plot snapshots

```
plt.figure(figsize=(8,4)) for t_here, u_snap in snapshots: plt.plot(x, u_snap,
label=f"t={t_here:.4f}") plt.plot(x, u0, 'k-', label='initial', linewidth=1.2)
plt.xlabel('x'); plt.ylabel('u(x,t)') plt.title('Heat equation (FTCS matrix form)')
plt.legend(fontsize='small') plt.grid(True) plt.show()
```

