Stochastic Processes:
Data Analysis and Computer Simulation

Python programming for beginners

3. The Euler method for numerical integration

3.1. Ordinary differential equations (ODE)

1st order ODE

- Consider the following 1st order differential equation.

$$\frac{dy(t)}{dt} = f(y(t), t) \tag{A1}$$

- Assume that the initial conditions are $y = y_0$ at time $t = t_0$.
- We need to determine $y(t)$, for any $t \geq t_0$.

Formal solution

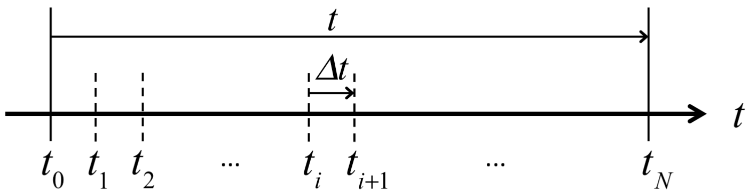
- Integrate Eq.(A1) over time, from $0 \rightarrow t$, to obtain the formal solution for $y(t)$

$$y(t) = y_0 + \int_{t_0}^{t} dt' f(y(t'), t') \tag{A2}$$

3.2. Numerical calculation

Discretization

- Divide the total time span $t_0 \rightarrow t$ into N equally spaced segments, each describing a time increment Δt.



$$t_i \equiv t_0 + i\Delta t, \quad y_i \equiv y(t_i), \quad f_i \equiv f(t_i, y(t_i)) \tag{A3}$$

Advancing the solution forward a small step Δt

- Integrate Eq.(A1) over a small time interval, from $t_i \rightarrow t_{i+1} (= t_i + \Delta t)$,

$$\begin{aligned}
y_{i+1} &= y_i + \int_{t_i}^{t_{i+1}} dt' f(y(t'), t') && \text{(A3)}\\
&= y_i + \int_0^{\Delta t} d\tau f(y(t_i + \tau), t_i + \tau) \qquad (\tau \equiv t' - t_i) && \text{(A4)}\\
&= y_i + \int_0^{\Delta t} d\tau \left[f_i + \mathcal{O}(\tau) + \mathcal{O}(\tau^2) + \cdots\right] && \text{(A5)}\\
&= y_i + \left[\tau f_i + \mathcal{O}(\tau^2) + \mathcal{O}(\tau^3) + \cdots\right]_0^{\Delta t} && \text{(A6)}\\
&= y_i + \Delta t f_i + \mathcal{O}(\Delta t^2) + \mathcal{O}(\Delta t^3) + \cdots && \text{(A7)}
\end{aligned}$$

Euler method

- Difference equation $\rightarrow$ 1st order in Δt

$$y_{i+1} = y_i + \int_{t_i}^{t_{i+1}} dt' f(y(t'), t') \simeq y_i + \Delta t f_i \tag{A8}$$

- Simulation procedure $\rightarrow$ Explicit method

$$y_0, f_0 \xrightarrow[\Delta t]{\text{Eq.(A8)}} y_1, f_1 \xrightarrow[\Delta t]{\text{Eq.(A8)}} \cdots y_i, f_i \cdots \xrightarrow[\Delta t]{\text{Eq.(A8)}} y_N, f_N \tag{A9}$$

- Forward difference approximation (1st order)

$$\left.\frac{dy(t)}{dt}\right|_{t=t_i} \simeq \frac{y_{i+1} - y_i}{\Delta t} \tag{A10}$$

Leapfrog method

- Central difference approximation

$$\left.\frac{dy(t)}{dt}\right|_{t=t_i} \simeq \frac{y_{i+1} - y_{i-1}}{2\Delta t} \tag{A11}$$

- Difference equation, Substitute Eq.(A11) in Eq.(A1)

$$y_{i+1} = y_{i-1} + \int_{t_{i-1}}^{t_{i+1}} dt' f(y(t'), t') \simeq y_{i-1} + 2\Delta t f_i \tag{A12}$$

- Simulation procedure → Explicit method

$$y_{-1}, f_0 \xrightarrow[2\Delta t]{\text{Eq.(A11)}} y_1, f_2 \xrightarrow[2\Delta t]{\text{Eq.(A11)}} \cdots y_i, f_{i+1} \cdots \xrightarrow[2\Delta t]{\text{Eq.(A11)}} y_N \tag{A13}$$

$$y_0, f_1 \xrightarrow[2\Delta t]{\text{Eq.(A11)}} y_2, f_3 \cdots \xrightarrow[2\Delta t]{\text{Eq.(A11)}} y_{N-1}, f_N \tag{A14}$$

Runge-Kutta (2nd)

- Difference equation

$$y_{i+1} = y_i + \int_{t_i}^{t_{i+1}} dt' f(y(t'), t') \simeq y_i + \Delta t f'_{i+\frac{1}{2}} \qquad \text{(Leapfrog)} \tag{A15}$$

with

$$y'_{i+\frac{1}{2}} = y_i + \frac{1}{2}\Delta t f_i, \qquad f'_{i+\frac{1}{2}} = f(y'_{i+\frac{1}{2}}, t_{i+\frac{1}{2}}) \qquad \text{(Euler)} \tag{A16}$$

- Simulation procedure → Explicit method

$$\begin{gathered} y_0, f_0 \xrightarrow[\frac{1}{2}\Delta t]{\text{Eq.(A16)}} y'_{\frac{1}{2}}, f'_{\frac{1}{2}} \qquad y_i, f_i \xrightarrow[\frac{1}{2}\Delta t]{\text{Eq.(A16)}} y'_{i+\frac{1}{2}}, f'_{i+\frac{1}{2}} \\ y_1, f_1 \xrightarrow[\frac{1}{2}\Delta t]{\text{Eq.(A16)}} y'_{1+\frac{1}{2}}, f'_{1+\frac{1}{2}} \qquad \cdots \xrightarrow[\frac{1}{2}\Delta t]{\text{Eq.(A16)}} y'_{N-\frac{1}{2}}, f'_{N-\frac{1}{2}} \\ y_0, f'_{\frac{1}{2}} \xrightarrow[\Delta t]{\text{Eq.(A15)}} y_1, f'_{1+\frac{1}{2}} \xrightarrow[\Delta t]{\text{Eq.(A15)}} \cdots y_i, f'_{i+\frac{1}{2}} \cdots \xrightarrow[\Delta t]{\text{Eq.(A15)}} y_N \end{gathered} \tag{A17}$$

Runge-Kutta (4th)

- Difference equation

$$\begin{aligned} y_{i+1} &= y_i + \int_{t_i}^{t_{i+1}} dt' f(y(t'), t') \\ &\simeq y_i + \frac{1}{6}\Delta t \left[f_i + 2f'_{i+\frac{1}{2}} + 2f''_{i+\frac{1}{2}} + f'''_{i+1} \right] \end{aligned} \tag{A18}$$

with

$$y'_{i+\frac{1}{2}} = y_i + \frac{\Delta t}{2} f_i, \qquad f'_{i+\frac{1}{2}} = f(y'_{i+\frac{1}{2}}, t_{i+\frac{1}{2}}) \tag{A19}$$

$$y''_{i+\frac{1}{2}} = y_i + \frac{\Delta t}{2} f'_{i+\frac{1}{2}}, \qquad f''_{i+\frac{1}{2}} = f(y''_{i+\frac{1}{2}}, t_{i+\frac{1}{2}}) \tag{A20}$$

$$y'''_{i+1} = y_i + \Delta t f''_{i+\frac{1}{2}}, \qquad f'''_{i+1} = f(y'''_{i+1}, t_{i+1}) \tag{A21}$$

3.3. Try the Euler method using Python

A very simple problem

- Numerically solve the following differential equation and determine $y(t)$ for $0 \le t \le 10$ with the initial condition $y = 1$ at $t = 0$. Then compare it with the analytical solution $y = \exp(-t)$.

$$\frac{dy(t)}{dt} = -y(t) \tag{A22}$$

In [1]:
```
% matplotlib inline
import numpy as np                 # import numpy library as np
import matplotlib.pyplot as plt # import pyplot library as plt
plt.style.use('ggplot')           # use "ggplot" style for graphs
```

In [2]:
```
# Euler method
dt, tmin, tmax = 0.1, 0.0, 10.0 # set \Delta t,t0,tmax
step=int((tmax-tmin)/dt)
# create array t from tmin to tmax with equal interval dt
t  = np.linspace(tmin,tmax,step)
y  = np.zeros(step) # initialize array y as all 0
ya = np.exp(-t) # analytical solution y=exp(-t)
plt.plot(t,ya,label='Exact',lw=5) # plot y vs. t (analytical)
y[0]=1.0 # initial condition
for i in range(step-1):
    y[i+1]=y[i]-dt*y[i] # Euler method Eq.(A8)
plt.plot(t,y,ls='--',lw=3,label='Numerical') # plot y vs t (numerical)
plt.plot(t,y/ya,lw=3,label='Ratio') # plot y/ya vs. t
plt.legend() #display legends
plt.show() #display plots
```