Lecture 26: Numerical Integration

- This course focuses on analytic methods for finding the response of a system due to a given input. However numerical methods for simulating the response are useful; i.e. SPICE.

  a) as a check on analytic results

  b) when input is very complicated, or random

  c) when system is non-linear and the techniques we have discussed do Not apply, except locally. Often linear systems interact with nonlinear systems and simulation is a good approach to understanding the global solution.

- So if simulation is so great, why do we teach the analytic methods?
  - Only gives approximate numerical results
  - must be re-run for any change in input
  - gives no real insight into issues of stability, ect.
  - gives no design approaches.

- Numerical Methods could easily consume an entire course. We will focus on just the basic Methods:
  - Forward Euler
  - Backward Euler
  - Runge-Kutta

- The basic problem formulation in general is, given a system with states $q \in \mathbb{R}^+ \times \mathbb{R}^n$ and $f: \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^n$

$$\dot{q} = f(t, q(t)) \quad \text{with I.C.} \quad q(0) = q_0 \qquad f = \text{"Flow" function.}$$

Simulate the solution $q(t)$ for arbitrary $t \geq 0$.

★ This includes $f(t, q(t)) = A\,q(t) + B\,x(t)$ our CT State Space representation.

– A simple example to get us started. Consider the 1st order system $\frac{dy}{dt}(t) + 3y(t) = x(t)$ for $t \geq 0$

where $x(t)$ is an arbitrary function for $t \geq 0$.

e.g. $x(t) = e^{-\frac{1}{8}(t-5)^2} u(t) + \frac{1}{t+1}\cos^2(5t)u(t)$

- Lets rewrite the ODE as $\frac{dy}{dt}(t) = -3y(t) + x(t)$

and approximate the derivative using a forward difference

$$\frac{dy}{dt}(t_0) \approx \frac{y(t_0+T) - y(t_0)}{T}$$ for some small T. at time t_0

- Substituting we get $\frac{y(t_0+T) - y(t_0)}{T} = -3y(t_0) + x(t_0)$

- Rearranging we arrive at $y(t_0+T) = (1-3T)y(t_0) + x(t_0)$ which tells us if we know $x(t_0)$ and $y(t_0)$ we can predict the future value at $t_0 + T$.

- Given $t_0 = 0$ and $x(t_0) = x_0$ $y(t_0) = y_0$ we can start predicting forward to times $t_0 + T$, $t_0 + 2T$, $t_0 + 3T$, etc.

  AND each of the operations is easy on a computer since we are just evaluating pure functions and combining terms.

- The above scheme is called a <u>Forward Euler Method</u> since it uses a forward Approximation

$\boxed{\text{Demo: simple C++ program solving the Above}}$

- How accurate this is depends obviously on T and is related to the Nyquist Criteria.

– The nice thing is, this scales to large, possibly nonlinear systems, with many states (in fact infinite in PDE case)

- Given $\dot{q} = f(t, q(t))$ and $q(t_0) = q_0$

  The forward Euler update is $q(t_0+T) = q(t_0) + T f(t_0, q(t_0))$

- Example: Van der Pohl system. $\dot{q}_1 = q_2$, $\dot{q}_2 = \mu(1-q_1^2)q_2 - q_1$

  with $q(t_0) = \begin{bmatrix} q_1(t_0) \\ q_2(t_0) \end{bmatrix}$

  and no input. $f(t, q(t)) = \begin{bmatrix} q_2(t) \\ \mu(1-q_1(t)^2)q_2(t) - q_1(t) \end{bmatrix}$

  $\boxed{\text{DEMO: FEtime}}$

– We could also approximate the derivative using a backward difference $\dot{q} = \frac{dq}{dt}(t) \approx \frac{q(t) - q(t-T)}{T}$

- This gives $q(t_0) = q(t_0-T) + T f(t_0, q(t_0))$ a system of equations that when solved give $q(t_0)$, given previous state $q(t_0-T)$.
- This is usually done using an optimization approach, for example

$$\underset{q(t_0)}{\operatorname{argmin}} \left[q(t_0) - q(t_0-T) - T f(t_0, q(t_0)) \right]^2$$

- When solved $q(t_0)$ becomes $q(t_0-T)$ for next step in simulation.

- This approach is called Backward Euler Method or an implicit method, since it requires solving an implicit equation at each step.

- While slower to compute, the implicit method gives more stable/accurate results for a given T compared to Forward Euler.

- By far the most popular method is called Runge-Kutta specifically 4^{th} order Runge-Kutta, or RK4.

  - Again given $\dot{q} = f(t, q(t))$ and $q(t_0) = q_0 \in \mathbb{R}^n$ and a step size T, the next step is given by

$$q(t_0+T) = q(t_0) + \frac{T}{6}\left[k_1 + 2k_2 + 2k_3 + k_4\right]$$

where
$$k_1 = f(t_0, q_0)$$
$$k_2 = f\left(t_0 + \frac{T}{2},\ q_0 + T\frac{k_1}{2}\right)$$
$$k_3 = f\left(t_0 + \frac{T}{2},\ q_0 + T\frac{k_2}{2}\right)$$
$$k_4 = f(t_0 + T,\ q_0 + T\cdot k_3)$$

  - It uses a weighted average of four different estimates of the update to improve accuracy and stability.

- Lets apply these schemes to simulate the response of a LTI system with reasonable input, and compare to the analytic solution.

Example: $H(s) = \dfrac{24}{(s+2)(s+3)(s+4)}$ $\quad x(t) = e^{-t}\cos(2t)\,u(t)$

  - Analysis: $Y(s) = H(s)\,X(s)$

$$= \frac{24(s+1)}{(s+2)(s+3)(s+4)(s^2+2s+5)}$$

Doing PFE

$$Y(s) = \frac{K_1}{s+2} + \frac{K_2}{s+3} + \frac{K_3}{s+4} + \frac{K_4 s + K_5}{s^2+2s+5}$$

$$K_1 = \frac{-12}{5} \quad K_2 = 6 \quad K_3 = -\frac{36}{13} \quad K_4 = \frac{-54}{65} \quad K_5 = \frac{30}{65}$$

— Example cont. Using table of transforms

$$y(t) = \left[-\frac{12}{5} e^{-2t} + 6e^{-3t} - \frac{36}{13} e^{-4t} + \frac{54}{65} e^{-t}\cos(2t) + \frac{42}{65} e^{-t}\sin(2t) \right] u(t)$$

- Now lets write in state-space form

$$H(s) = \frac{Y(s)}{X(s)} = \frac{24}{s^3 + 9s^2 + 26s + 24}$$

$$s^3 Y(s) + 9s^2 Y(s) + 26 s Y(s) + 24 Y(s) = 24 X(s)$$

$$\frac{d^3y}{dt^3} + 9\frac{d^2y}{dt^2} + 26\frac{dy}{dt} + 24y = 24x(t)$$

Let $q_1 = y$ $\quad q_2 = \frac{dy}{dt}$, $\quad q_3 = \frac{d^2y}{dt^2}$

Then

$$\dot{q} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -24 & -26 & -9 \end{bmatrix} q(t) + \begin{bmatrix} 0 \\ 0 \\ 24 \end{bmatrix} x(t)$$

$$= f(q(t))$$

with $y(t) = q_1(t)$

- Using this form we can use RK4 and compare to analytic result.

$\boxed{\text{see DEMO}}$