ME451
Kinematics and Dynamics of Machine Systems

Introduction to Dynamics

Numerical Integration Techniques

December 2, 2014

- Quote of the day: "Success is going from failure to failure without loss of enthusiasm." -- Winston Churchill

Dan Negrut
ME451, Fall 2014
University of Wisconsin-Madison

Before we get started…

- Last time
  - Setting up initial conditions for the dynamics analysis
  - Revisit the computation of reaction forces
- Today
  - Numerical Integration
  - [Begin] Solving the constrained equations of motion using the Newmark integration formulas
- **NOTE**: Bring this deck of slides to class next Tu
- New assignment, last of the class – due on December 9
  - MATLAB 9 – posted online
  - ADAMS 6 – posted online
- Midterm Exam on Th December 4, open everything
  - Review session on Wd, December 3 at 7:15 PM (same format like last time)
    - Review session room: 1163ME (next door room)
- Project 2 due on 12/16 at 11:59 PM

Before we get started…

- Final Exam: content
  - Part 1: Pen and paper
    - You'll have to generate a pair of acf/adm files but you don't have to use these files unless you go for the bonus
  - Part 2: Bonus (extra credit)
    - You'll have to use simEngine2D and the pair of acf/adm files
  - Score cannot exceed 100%

- Final Exam: logistics
  - Tuesday, December 16, 2014
  - 2:45 PM - 4:45 PM
  - Room: 2109ME (computer lab)
  - MATLAB access – one of two choices:
    - Bring your own laptop
    - Use CAE machine

- Final Project
  - Due on Friday, December 19 at 11:59 PM

- Dropped HW policies
  - Lowest 5 scores amongst the MATLAB, pen-and-paper, and ADAMS assignments will be dropped

Numerical Integration
[i.e., solving an IVP using a numerical method]

Numerical Method

(also called Numerical Algorithm, or simply Algorithm)

- Represents a recipe, a succession of steps that one takes to find an approximation of the solution of a problem that otherwise does not admit an analytical solution
  - Analytical solution: sometimes called "closed form" or "exact" solution
  - The approximate solution obtained with the numerical method is also called "numerical solution"

- Examples:
  - Evaluate the integral $I = \int_0^3 e^{-x^2} dx$
  - Solve the equation $e^x + \sin(\frac{2}{x}) + \sqrt{x} = 2$
  - Many, many others
    - Actually very seldom can you find the exact solution of a problem that originates in a real world engineering application

Where/How are Numerical Methods Used?

- Powerful and inexpensive computers have revolutionized the use of numerical methods and their impact
  - Simulation of a car crash in minute detail
  - Formation of galaxies
  - Motion of atoms
  - Finding the charge density around nuclei in a nanostructure
  - Solve the partial differential equations that govern combustion in a cylinder

- Numerical methods enable the concept of “simulation-based engineering”
  - You use computer simulation to understand how your system behaves, how it can be modified and controlled

Numerical Methods in ME451

- In regards to ME451, one would use numerical method to solve the dynamics problem (the resulting set of differential equations that capture Newton's second law)
  - The particular class of numerical methods used to solve differential equations is typically called "**numerical integrators**", or "integration formulas"
  - A numerical integrator generates a numerical solution at **discrete time points** (also called grid points, station points, nodes)
    - This is in fact just like in Kinematics, where the solution is computed on a time grid
  - Different numerical integrators generate **different solutions**, but the solutions are typically very close together, and [hopefully] closed to the actual solution of our problem
  - Putting things in perspective: In 99% of the cases, the use of numerical integrators is **the only alternative** for solving complicated systems described by non-linear differential equations

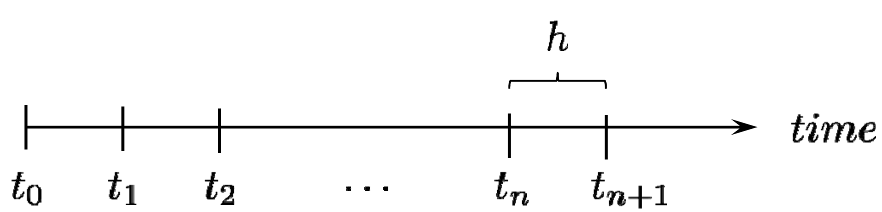
Numerical Integration ~Basic Concepts~

- Initial Value Problem: (IVP)

$$\begin{cases} \dot{y} = f(t, y) \\ y(t_0) = y_0 \end{cases}$$

- The general framework
  - You are looking for a function $y(t)$ that depends on time (changes in time), whose time derivative is equal to a function $f(t, y)$ that is given to you (see IVP above)
  - In other words, I give you the derivative of a function, can you tell me what the function is?
  - Known quantities: both y_0 and the function f. Unknown quantity: $y(t)$.

- In ME451, the best you can hope for is to find an approximation of the unknown function $y(t)$ at a sequence of discrete points (as many of them as you wish)
  - The numerical algorithm produces an approximation of the value of the unknown function $y(t)$ at the each grid point. That is, the numerical algorithm produces $y(t_1)$, $y(t_2)$, $y(t_3)$, etc.



Relation to ME451

- When carrying out Dynamics Analysis, what you can compute is the acceleration of each part in the model

- Acceleration represents the second time derivative of your coordinates

- Somewhat oversimplifying the problem to make the point across, in ME451 you get the second time derivate

$$\ddot{\mathbf{q}} = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, t)$$

- This represents a second order differential equation since it has two time derivatives taken on the position **q**

Numerical Integration: Euler's Method

Find solution of this Initial Value Problem:

$$\begin{cases} \dot{y}(t) = f(y,t) \\ y(t_0) = y_0 \end{cases}$$

- The idea: at each grid point t_k, **turn the differential problem into an algebraic problem** by approximating the value of the time derivative:

$$\dot{y}(t_k) \approx \frac{y(t_{k+1}) - y(t_k)}{t_{k+1} - t_k} = \frac{y_{k+1} - y_k}{\Delta t} \quad \Rightarrow \quad y_{k+1} = y_k + \Delta t\, \dot{y}(t_k)$$

This step is called "discretization". It transforms the problem from a continuous **ODE** problem into a discrete **algebraic** problem

Euler's Method (Δt is the step size):

$$y_{k+1} = y_k + \Delta t\, f(t_k, y_k)$$

Forward Euler (FE): Example

- Solve the IVP

$$\dot{y} = -10y$$
$$y(0) = 1$$

using Forward Euler (FE) with a step-size $h = 0.01$

- Compare to the exact solution

FE integration with $h = 0.01$, $f(t, y) = -10y$

$$\begin{array}{llllll}
n=0 & y_0 = 1.0 & & & & = 1.0 \\
n=1 & y_1 = y_0 + h \cdot f(t_0, y_0) & = 1.0 & + & 0.01 \cdot (-10 * 1.0) & = 0.9 \\
n=2 & y_2 = y_1 + h \cdot f(t_1, y_1) & = 0.9 & + & 0.01 \cdot (-10 * 0.9) & = 0.81 \\
n=3 & y_3 = y_2 + h \cdot f(t_2, y_2) & = 0.81 & + & 0.01 \cdot (-10 * 0.81) & = 0.729 \\
n=4 & y_4 = y_3 + h \cdot f(t_3, y_3) & = 0.729 & + & 0.01 \cdot (-10 * 0.729) & = 0.6561 \\
n=5 & y_5 = y_4 + h \cdot f(t_4, y_4) & = 0.6561 & + & 0.01 \cdot (-10 * 0.6561) & = 0.5905
\end{array}$$

Exact solution: $y(t) = e^{-10t}$

$$\begin{array}{lcl}
y(0) & = & 1.0 \\
y(1) & = & 0.9048 \\
y(2) & = & 0.8187 \\
y(3) & = & 0.7408 \\
y(4) & = & 0.6703 \\
y(5) & = & 0.6065
\end{array}$$

Forward Euler: Effect of Step-Size

IVP: $\dot{y} = -0.1y + \sin(t)$, $y(0) = 0$

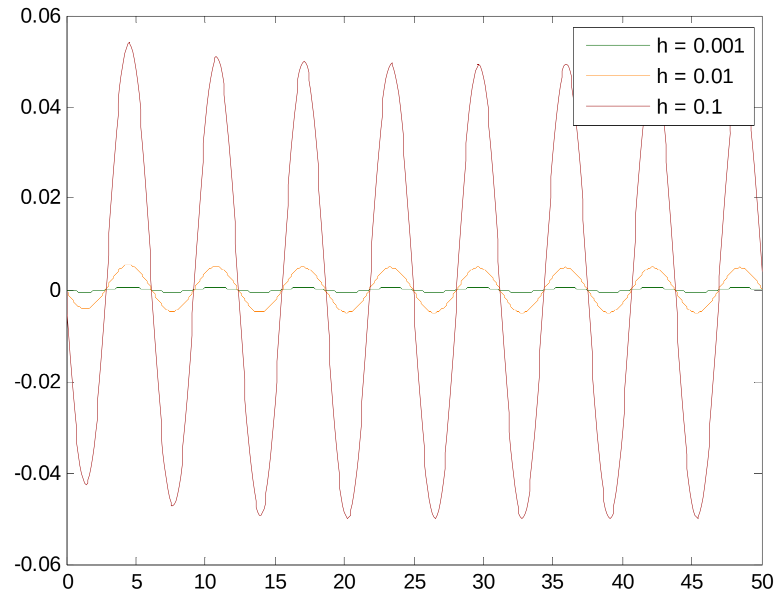
$$y_{exact}(t) = \frac{1}{1.01}\left(e^{-0.1t} + 0.1\sin(t) - \cos(t)\right)$$

```
% IVP (RHS + IC)
f = @(t,y) -0.1*y + sin(t);
y0 = 0;
tend = 50;

% Analytical solution
y_an = @(t) (0.1*sin(t) - cos(t) + exp(-0.1*t)) / (1+0.1^2);

% Loop over the various step-size values and plot errors
colors = [[0, 0.4, 0]; [1, 0.5, 0]; [0.6, 0, 0]];
Figure, hold on, box on
h = [0.001 0.01 0.1];
for ih = 1:length(h)
    tspan = 0:h(ih):tend;
    y = zeros(size(tspan));  err = zeros(size(tspan));
    y(1) = y0;               err(1) = 0;
    for i = 2:length(tspan)
        y(i) = y(i-1) + h(ih) * f(tspan(i-1), y(i-1));
        err(i) = y(i) - y_an(tspan(i));
    end
    plot(tspan, err, 'color', colors(ih,:));
end
legend('h = 0.001', 'h = 0.01', 'h = 0.1');
```

FE errors for different values of the step-size $h = 0.001, 0.01, 0.1$



Stiff Differential Equations.
Explicit vs. Implicit Numerical Integration Formulas

A Simple IVP Example

- Consider the IVP

$$y' = -100y$$
$$y(0) = 1$$

whose analytical solution is

$$y(t) = e^{-100t}$$

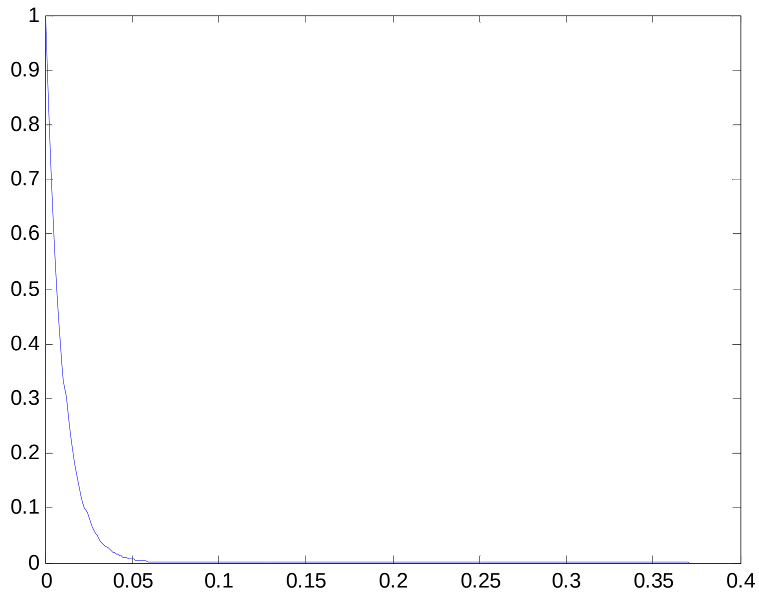
- 5 integration steps with Forward Euler formula, $h = 0.002$, $h = 0.01$, $h = 0.03$
- Compare the global error (difference between analytical and numerical solution)

Error for $h = 0.002$	Error for $h = 0.01$	Error for $h = 0.03$
0	0	0
0.01873075307798	0.36787944117144	2.04978706836786
0.03032004603564	0.13533528323661	-3.99752124782333
0.03681163609403	0.04978706836786	8.00012340980409
0.03972896411722	0.01831563888873	-15.99999385578765
0.04019944117144	0.00673794699909	32.00000030590232

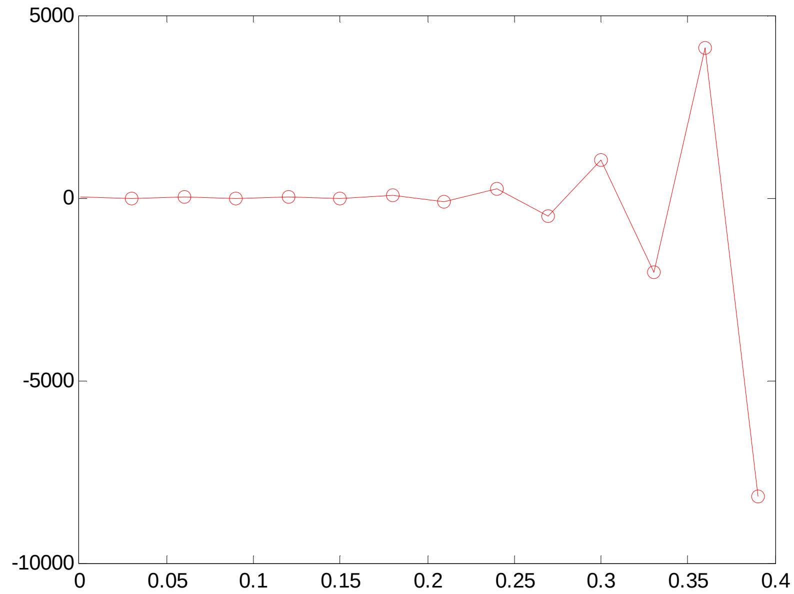
A Simple IVP Example

$$y' = -100y$$
$$y(0) = 1$$

Analytical
Solution



Forward Euler
$h = 0.03$



Stiff Differential Equations

- Problems for which explicit integration methods (such as Forward Euler) do not work well
  - Other explicit formulas: Runge-Kutta (RK4), DOPRI5, Adams-Bashforth, etc.

- Stiff problems require a different class of integration methods: implicit formulas
  - The simplest implicit integration formula: Backward Euler (BE)

$$y_{n+1} = y_n + h \cdot f(t_{n+1}, y_{n+1})$$

Forward Euler: Rationale Behind It

- Starting from the IVP

$$\dot{y} = f(t, y)$$
$$y(t_0) = y_0$$

- Look back to approximation the derivative

$$\dot{y}(t_{n+1}) \approx \frac{y(t_{n+1}) - y(t_n)}{t_{n+1} - t_n} = \frac{y_{n+1} - y_n}{t_{n+1} - t_n} = \frac{y_{n+1} - y_n}{h}$$

- Rewrite the above as

$$y_{n+1} = y_n + h\dot{y}(t_{n+1})$$

and use ODE to obtain

$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$$

Backward Euler Method with constant step-size h

BE: Geometrical Interpretation

- IVP

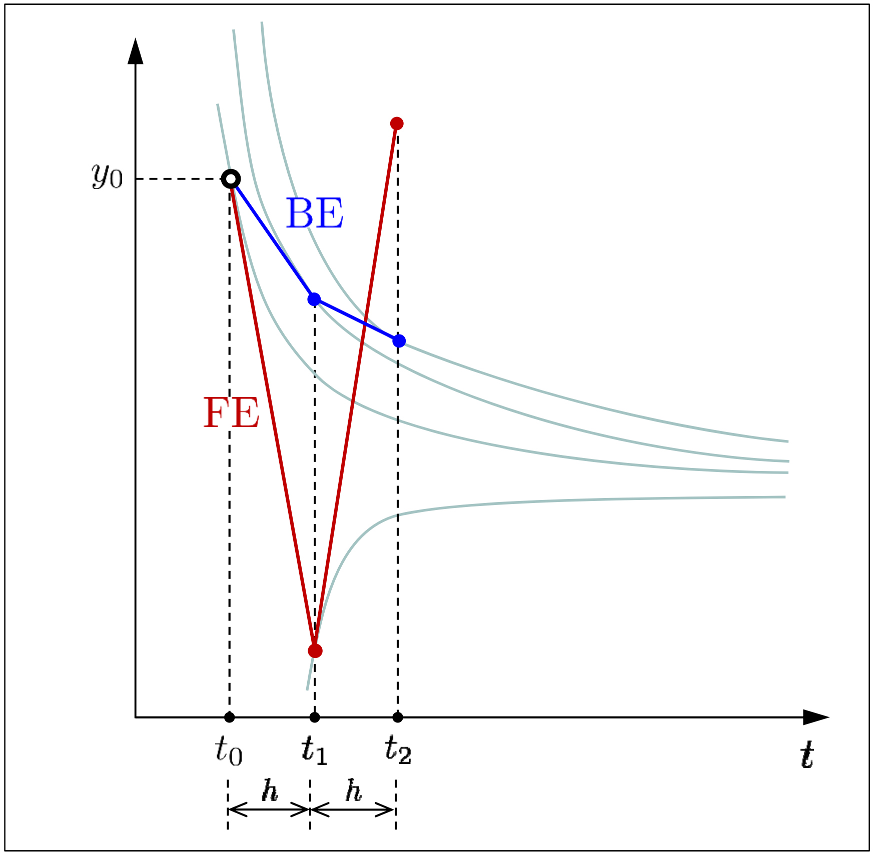
$$\dot{y} = f(t, y)$$
$$y(t_0) = y_0$$

- Forward Euler integration formula

$$y_{n+1} = y_n + hf(t_n, y_n)$$

- Backward Euler integration formula

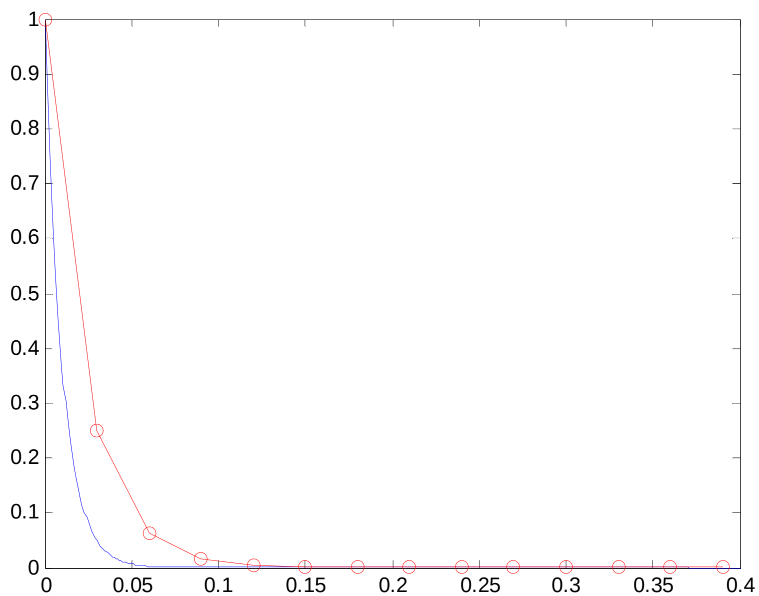
$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$$



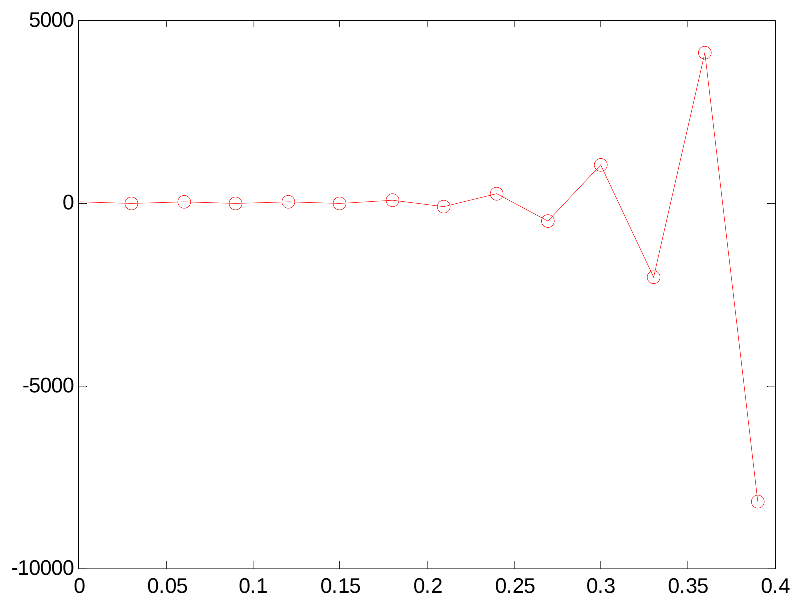
A Simple IVP Example

$$y' = -100y$$
$$y(0) = 1$$

Analytical Solution
Backward Euler $h = 0.03$



Forward Euler $h = 0.03$



Forward Euler vs. Backward Euler

Initial Value Problem (IVP)

$$\dot{y} = f(t, y)$$
$$y(t_0) = y_0$$

Forward Euler

$$y_{n+1} = y_n + h\dot{y}_n$$
$$\dot{y}_n \equiv f(t_n, y_n)$$

$$y_{n+1} = y_n + h \cdot f(t_n, y_n)$$

Backward Euler

$$y_{n+1} = y_n + h\dot{y}_{n+1}$$
$$\dot{y}_{n+1} \equiv f(t_{n+1}, y_{n+1})$$

$$y_{n+1} = y_n + h \cdot f(t_{n+1}, y_{n+1})$$

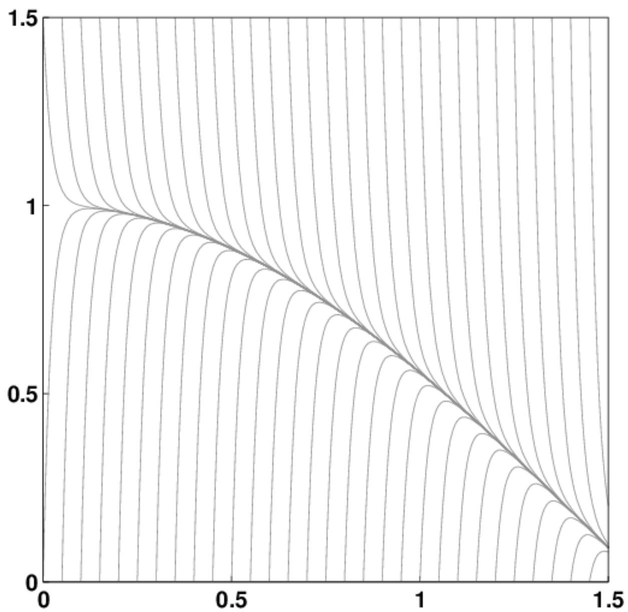
Backward-Difference Formulas (BDF)

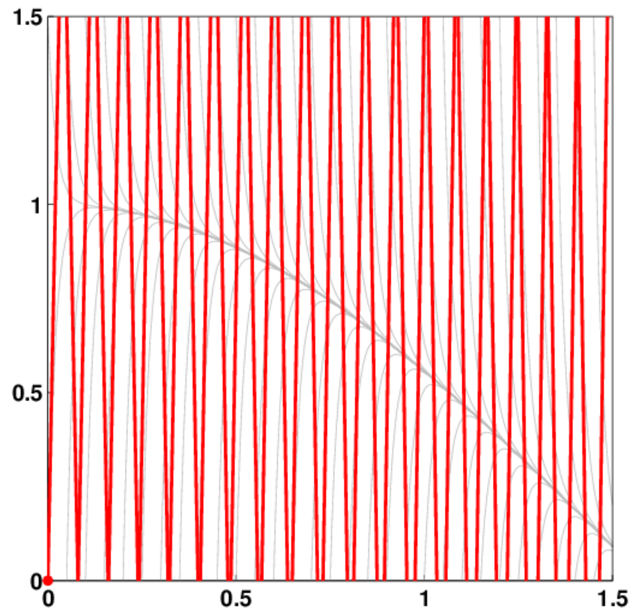
- IVP $\dot{y} = f(t, y)$
  $y(t_0) = y_0$
- Also known as Gear method (DIFSUB – 1971)
- Family of implicit linear multi-step formulas
  - BDF of 1st order: $y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$
  - BDF of 2nd order: $y_{n+1} = \frac{4}{3}y_n - \frac{1}{3}y_{n-1} + \frac{2}{3}hf(t_{n+1}, y_{n+1})$
  - BDF of 3rd order: $y_{n+1} = \frac{18}{11}y_n - \frac{9}{11}y_{n-1} + \frac{2}{11}y_{n-2} + \frac{6}{11}hf(t_{n+1}, y_{n+1})$
  - BDF of 4th order: $y_{n+1} = \frac{48}{25}y_n - \frac{36}{25}y_{n-1} + \frac{16}{25}y_{n-2} - \frac{3}{25}y_{n-3} + \frac{12}{25}hf(t_{n+1}, y_{n+1})$
  - BDF of 5th order: $y_{n+1} = \frac{300}{137}y_n - \frac{300}{137}y_{n-1} + \frac{200}{137}y_{n-2} - \frac{75}{137}y_{n-3} + \frac{12}{137}y_{n-4} + \frac{60}{137}hf(t_{n+1}, y_{n+1})$

C.W. (Bill) Gear
b. 1935

Curtiss & Hirschfelder Example

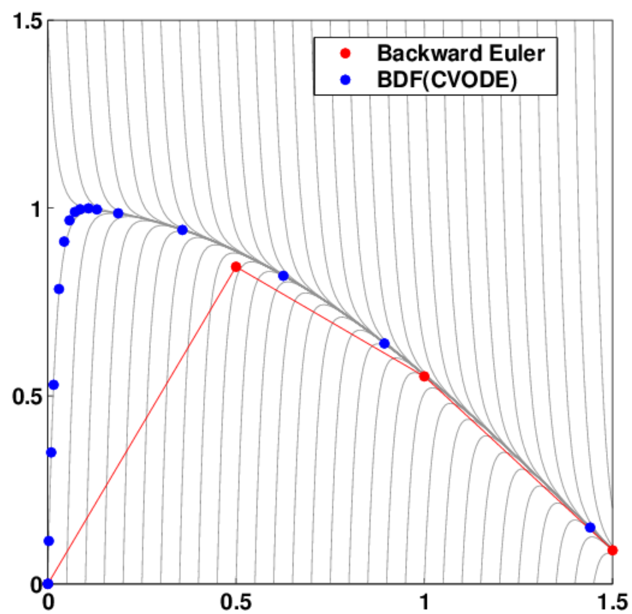
$$\dot{y} = -50\,(y - \cos(t))$$
$$y(0) = 0$$





Forward Euler

$h = 2.01/50$



Backward Euler

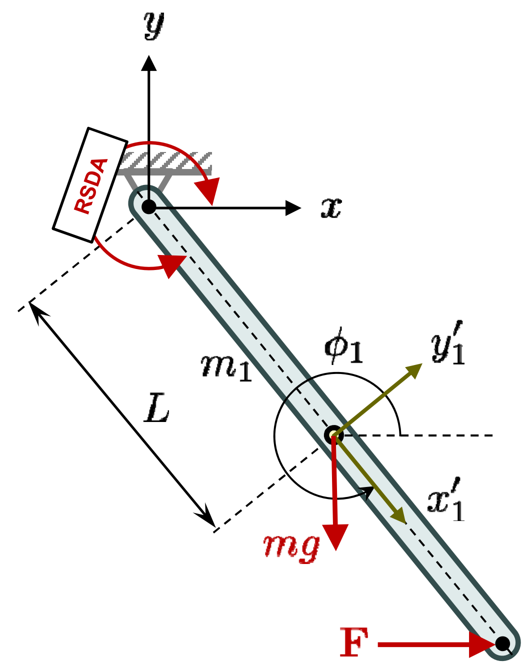
$h = 25/50$

C.F. Curtiss and J.O. Hirschfelder – "Integration of Stiff Equations"
Proc. Nat. Acad. Sci, USA (1952)

Back to ME451

Sample Problem
Find the time evolution of the pendulum

- Simple Pendulum:
  - Mass $m = 20\ kg$
  - Half-length $L = 2\ m$
  - Force acting at tip of pendulum
    - $F = 30\sin(2\pi t)\ \ [N]$
  - RSDA element in revolute joint
    - $\phi_0 = 3\pi/2$
    - $k = 45\ Nm/rad$
    - $c = 10\ \ N/s$
  - ICs: hanging down, starting from rest



Dynamics Analysis:
The 30,000 Feet Perspective

- Three Steps for Dynamic Analysis:
  1. Derive constrained equations of motion (you get a DAE problem)
  2. Specify initial conditions (ICs)
  3. Use numerical integration algorithm to discretize DAE problem and turn it into an algebraic problem

Lagrange Multiplier Form of the Constrained Equations of Motion

- Equations of Motion $\mathbf{M}\ddot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^T \lambda = \mathbf{Q}^A$
- Position Constraint Equations $\mathbf{\Phi}(\mathbf{q}, t) = \mathbf{0}$
- Velocity Constraint Equations $\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} = -\mathbf{\Phi}_t \triangleq \nu$
- Acceleration Constraint Equations $\mathbf{\Phi}_{\mathbf{q}}\ddot{\mathbf{q}} = -\left(\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}\right)_{\mathbf{q}}\dot{\mathbf{q}} - 2\mathbf{\Phi}_{\mathbf{q}t}\dot{\mathbf{q}} - \mathbf{\Phi}_{tt} \triangleq \gamma$

Variables in the DAE Problem

- Generalized accelerations: $\ddot{\mathbf{q}}$
- Generalized velocities: $\dot{\mathbf{q}}$
- Generalized positions: $\mathbf{q}$
- Lagrange multipliers: $\boldsymbol{\lambda}$

All these quantities are functions of time (they change in time)

What's Special About the EOM of Constrained Planar Systems?

- There are three things that make the ME451 dynamics problem challenging:
  - The problem is **not in standard form** $\dot{y} = f(t, y)$
  - Moreover, the problem is **not a first order** ODE
    - The EOM contain the second time derivative of the positions
  - In fact, the problem is **not even an ODE**
    - The unknown function **q**(t), that is the position of the mechanism, is the solution of a second order differential equation but it must also satisfy a set of kinematic constraints at position, velocity, and acceleration levels, formulated as a set of algebraic equations
      - We have solve a set of differential-algebraic equations (DAEs)
      - DAEs are much harder to solve than ODEs

Linda R. Petzold – "Differential/Algebraic Equations Are Not ODEs"
SIAM J. of Scientific and Statistical Computing (1982)

Numerical Solution of the DAEs in Constrained Multibody Dynamics

- This is a research topic in itself
- We cover one of the simplest algorithms possible
  - We will use Newmark's formulas to discretize the index-3 DAEs of constrained multibody dynamics
  - Note that the textbook does not discuss this method

Nathan M. Newmark (1910 – 1981)

Solution Strategy

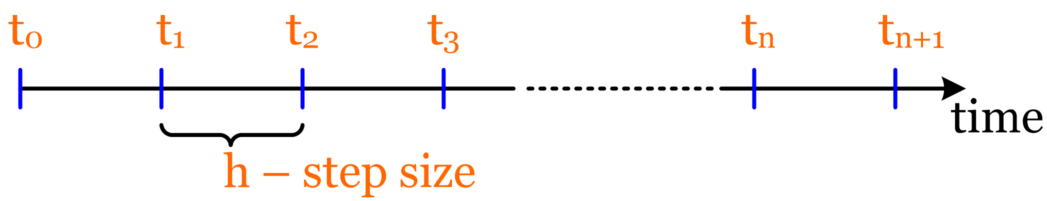
[Step 3 of the "Three Steps for Dynamics Analysis", see Slide 25]

The numerical solution; i.e., an approximation of the actual solution of the dynamics problem, is produced in the following three stages:

- **Stage 1**: the Newmark numerical integration (discretization) formulas are used to **express the positions and velocities as functions of accelerations**
- **Stage 2**: everywhere in the constrained EOM, the positions and velocities are **replaced using the discretization formulas** and expressed in terms of the acceleration
  - This is the most important step, since through this "discretization" the **differential problem is transformed into an algebraic problem**
- **Stage 3**: the acceleration and Lagrange multipliers are obtained by **solving a nonlinear system**

Newmark Integration Formulas (1/2)

- Goal: find the positions, velocities, accelerations and Lagrange multipliers on a grid of time points; i.e., at $t_0, t_1, t_2, \ldots$



- <u>Stage 1/3</u> – Newmark's formulas relate *position to acceleration* and *velocity to acceleration*:

$$\mathbf{q}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\left[(1-2\beta)\ddot{\mathbf{q}}_n + 2\beta\ddot{\mathbf{q}}_{n+1}\right] \equiv \mathbf{p}(\ddot{\mathbf{q}}_{n+1})$$

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + h\left[(1-\gamma)\ddot{\mathbf{q}}_n + \gamma\ddot{\mathbf{q}}_{n+1}\right] \equiv \mathbf{v}(\ddot{\mathbf{q}}_{n+1})$$

- <u>Stage 2/3</u> – Newmark's method (1957) discretizes the second order EOM:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{F}(t) \quad \Leftrightarrow \quad \mathbf{M}\ddot{\mathbf{q}}_{n+1} + \mathbf{C}\dot{\mathbf{q}}_{n+1} + \mathbf{K}\mathbf{q}_{n+1} = \mathbf{F}(t_{n+1})$$

Newmark Integration Formulas (2/2)

- Newmark Method
  - Initially introduced to deal with linear transient Finite Element Analysis
  - Accuracy: 1st Order
  - Stability: Very good stability properties
  - Choose values for the two parameters controlling the behavior of the method: $\beta = 0.3025$ and $\gamma = 0.6$

- Write the EOM at each time t_{n+1}

$$\mathbf{M}\ddot{\mathbf{q}}_{n+1} + \mathbf{C}\dot{\mathbf{q}}_{n+1} + \mathbf{K}\mathbf{q}_{n+1} = \mathbf{F}(t_{n+1})$$

- Use the discretization formulas to replace $\mathbf{q}_{n+1}$ and $\dot{\mathbf{q}}_{n+1}$ in terms of the accelerations $\ddot{\mathbf{q}}_{n+1}$ using formulas on previous slide:

$$\mathbf{q}_{n+1} = \mathbf{p}(\ddot{\mathbf{q}}_{n+1}) \quad \text{and} \quad \dot{\mathbf{q}}_{n+1} = \mathbf{v}(\ddot{\mathbf{q}}_{n+1})$$

- Obtain algebraic problem in which the unknown is the acceleration (denoted here by $\boldsymbol{x}$):

$$\mathbf{M} \cdot \mathbf{x} + \mathbf{C} \cdot \mathbf{v}(\mathbf{x}) + \mathbf{K} \cdot \mathbf{p}(\mathbf{x}) = \mathbf{F}(t_{n+1})$$