ME751 Advanced Computational Multibody Dynamics

Solving Initial Value Problems

Basic Concepts

March 23, 2010

“I have bad reflexes. I was once run over by a car being pushed by two guys.“
Woody Allen

Before we get started…

- Last Time:
  - Super briefly talked about the EOM when using Euler Angles
  - Discussed two classes of forces likely to be encountered in Engineering Apps
  - Inverse Dynamics Analysis
  - Equilibrium Analysis

- Today:
  - Discuss about the solution of differential equations

- Final Project:
  - Attach the one pager proposal to your Th HW

- I'll need a head count in the next two weeks regarding people who plan to make the trip to John Deere and NADS

Dynamics Analysis

- Dynamics Analysis, Framework:
  - The state of a mechanical system (position, velocity) changes in time under the influence of internal and external **forces and/or prescribed motions**
  - The goal is to determine how the state of the system changes in time
  - Almost always you will only be able to determine the state of the mechanical system at a collection of grid points in time
    - That is, not everywhere, yet you can have as many grid points you wish (and afford)
  - Time evolution is obtained as the solution of the EOM (Newton-Euler equations derived before)

Dynamics vs. Kinematics

- Kinematics Analysis
  - Prescribed motions exclusively determine how the system changes in time
    - The concept of force/torque does not factor in anywhere
  - For a Kinematics Analysis to be possible, the NDOF should be zero
  - Its solution provided at each time step by a sequence of 3 **algebraic** problems:
    - Nonlinear system of equations provides the position at each time step
    - Linear system of equations provides the velocity configuration at each time step
    - Linear system of equations provides the acceleration configuration at each time step

- Dynamics Analysis
  - External forces/torques dictate how the system evolves in time
  - It is more general than Kinematics:
    - A Kinematics problem can be solved using the methods of Dynamics, but not the other way around
  - Its solution obtained at each time step by numerical integration (solving a differential equation)

Dealing With Dynamic Systems

- What have we done in ME751 so far?
  - We posed a dynamics problem
    - Discussed the elements of a mechanical system
    - Understood what their presence entails
    - Formulated a set of second order differential equations that governs the time evolution of the system
  - What's left: solving the differential equation to obtain the dynamics of the mechanical system
    - Four lectures: discuss solution techniques that produce an approximation of the solution of the differential problem

30,000 Feet Perspective

- 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, in ME751 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 **q**
- Problem is reduced to a set of first order differential equations by introducing a helper variable **v** (the velocity):

$$\dot{\mathbf{q}} = \mathbf{v}$$

- With this, the original second order differential problem becomes:

$$\dot{\mathbf{y}} = \mathbf{F}(\mathbf{y}, t) \quad \text{where} \quad \mathbf{y} = \begin{bmatrix} \mathbf{q} \\ \mathbf{v} \end{bmatrix} \quad \text{and} \quad \mathbf{F}(\mathbf{y}, t) \equiv \begin{bmatrix} \mathbf{v} \\ \mathbf{f}(\mathbf{v}, \mathbf{q}, t) \end{bmatrix}$$

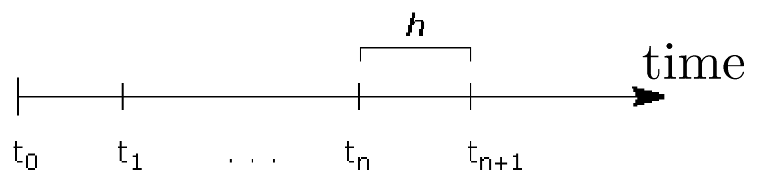
Numerical Integration ~The Problem ~

- Initial Value Problem: (IVP)

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

- IVP: Stating the Problem
  - 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 equation above)
  - In other words, you are given the derivative of a function, can you tell what the function is?

- In ME751, 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 <u>approximation</u> of the value of the unknown function y(t) at the each grid point. That is, the numerical algorithm produces an approximation for $y(t_1)$, $y(t_2)$, $y(t_3)$, etc.; i.e., y_1, y_2, y_3, etc.



Road Map (Tu & Th)

- Difference between ODE and IVP
- Basic Concepts in Numerical Integration
- Basic Methods for Numerical Integration
  - Runge-Kutta
  - AB & AM Methods
  - BDF Methods
- Text used:
  - Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations, by U. Ascher and L. Petzold, SIAM, 1998
  - On reserve at Wendt Library

ODE vs. IVP

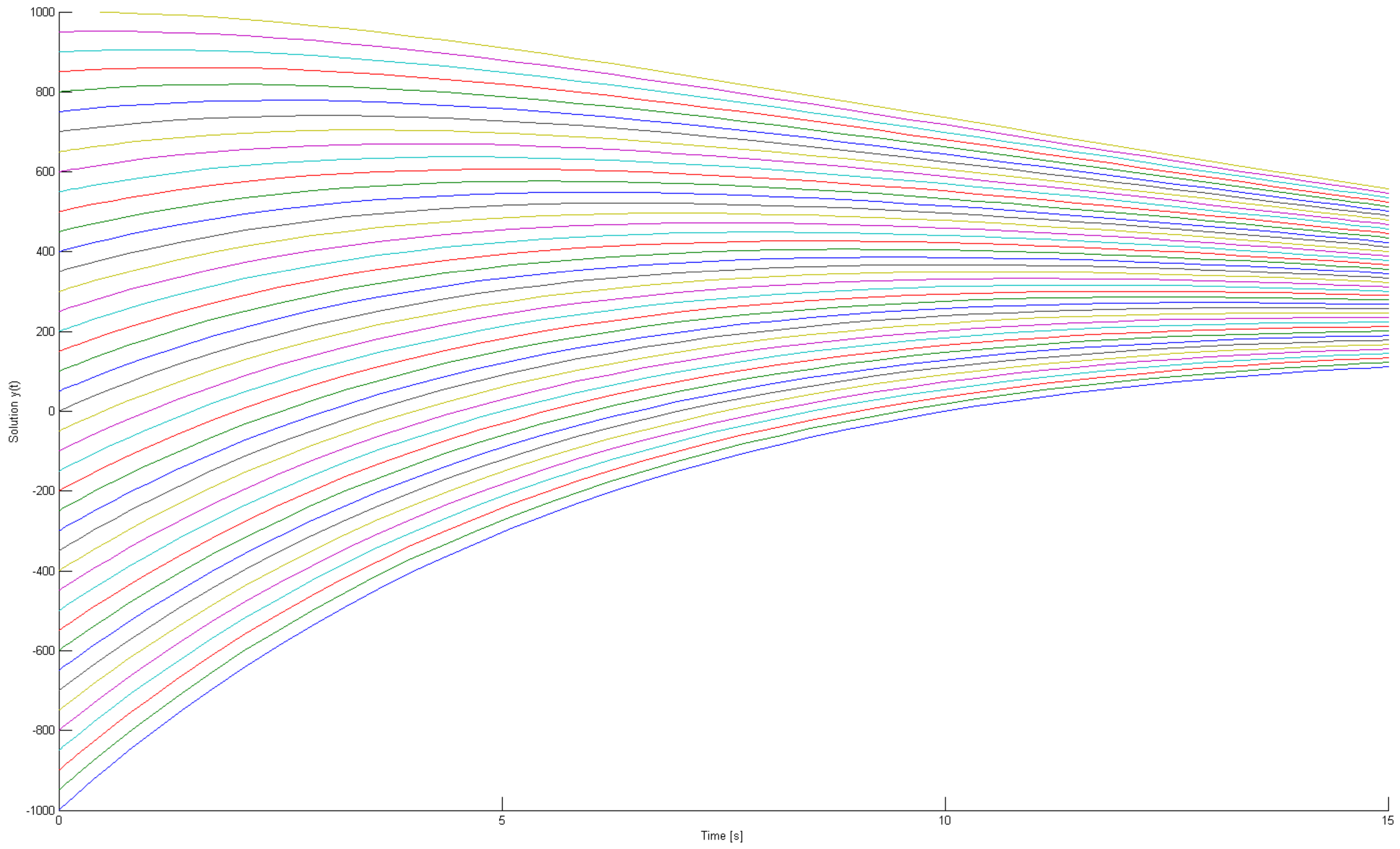
- Difference between ODE and IVP - Often a source of confusion

- Ordinary Differential Equation (ODE)
  - Typically, has an infinite number of solutions

- Initial Value Problem (IVP)
  - Is an ODE **plus** an initial condition (IC):
    - The IC: The unknown function assumes at time T=0 a certain prescribed value
  - The solution for the IVP's that we'll deal with is **UNIQUE**
    - We'll assume that f(t,y) is well behaved (Lipschitz continuous)

ODE: Infinite Number of Solutions

- ODE Problem: $\dot{y}(t) = -0.1y(t) + 100e^{-0.1\,t}$
- A range of Initial Conditions (ICs) is specified: $y_0 = [-1000 : 50 : 1000]$



ODE vs. IVP
[Cntd.]

- Remember:
  1. **IVP = ODE + IC**
  2. **IVP has a UNIQUE solution (unlike on ODE)**

- Why is observation above important?
  - In general, when we start working on a numerical solution to a problem we better know that the problem we are trying to solve is well posed (has a solution and it is unique)

- Turns out that in the Dynamics Analysis we are dealing with a well posed problem (an IVP)
  - Focus then on finding a way to approximate its solution by using the computer
    - The computer will produce numbers that at each node of the time grid will approximate the value of the generalized coordinates and their velocity

Initial Value Problems: Basic Concepts

- Truncation Error
- Accuracy
- Convergence
- 0-stability
- Order of a method
- Local Error
- Stability
  - Absolute stability
  - A-stable Integration Methods
  - L-Stable Integration Methods (Methods with Stiff Decay)

Framework

- Interested in finding a function y(t) over an interval [0,b]
- This function must satisfy the following IVP:

$$\begin{cases} \dot{y} = f(t, y) \\ y(0) = c \end{cases} \qquad t \in [0, b]$$

- We assume that f is bounded and smooth, so that y exists, is unique, and smooth

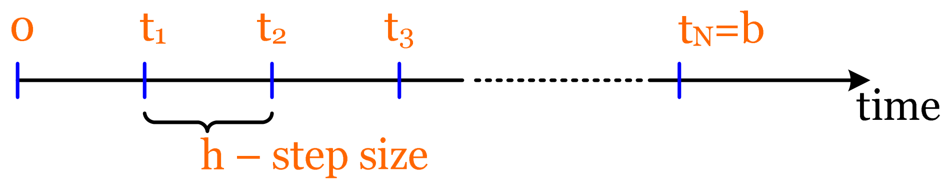
- Given to you:
  - The constants c and b
  - The function f(t,y).

Framework [Cntd.]

- The expression of the given function f(t,y) is typically far from being simple
  - Case 1: f(t,y) is simple, in very rare cases you can find y(t) analytically; i.e., find the exact solution of the IVP (the pen and paper case)
  - Case 2: f(t,y) is complex but nonetheless you have direct access to it.
    - Producing an exact solution is not possible, resort to numerical integration, solution approximated using the computer
  - Case 3: f(t,y) is so complex that you don't even have an expression for it. Instead you have another application (program) evaluate f(t,y) based on the value of t and y that you provide
    - Producing an exact solution is not possible, resort to numerical integration, solution approximated using several computer programs
    - This is what happens typically in multi-discipline engineering analysis

Framework [Cntd.]

- Implications (in relation to the complexity of f(t,y)):
  - Most likely you can't find the function y(t) analytically (using pen and paper)
  - You'll have to use a numerical approximation method to come up with approximations of the unknown function y(t)
    - What you'll produce is an approximation of the value of y(t) at 0, t_1, t_2, etc.
    - You are working on a grid of of N time points, starting at 0 and ending at b
    - For simplicity, we assume that the grid points are equally spaced by a value h
    - This value is called the "integration step-size", usually denoted by h or Δt



Framework [Cntd.]

- Remember this: we'll approximate the value of $y(t_n)$ by a value y_n that we'll learn how to obtain

$$\underbrace{y_n}_{\text{An approximation of the value of the unknown function at time } t_n} \approx \overbrace{y(t_n)}^{\text{The unknown function y evaluated at } t_n}$$

- Notation: I'll use y^h to denote the set of values y_0, y_1, …, y_N that I come up with in my effort to approximate the unknown function y(t) at the grid points t_0, t_1, …, t_N:

$$y^h = \{y_0, y_1, \ldots, y_N\}$$

- Quick Remark: a professional grade method for finding the approximate solution does not use an equally spaced grid of points 0, t_1, t_2, t_3…
  - We'll stick with this assumption of equally spaced points though

Basic Concepts: Truncation Error
[Preliminaries]

- We have our standard IVP:
$$\begin{cases} \dot{y} &= f(t,y) \\ y(0) &= c \end{cases}$$

- Before talking about Truncation Error, let's introduce the simplest numerical integration scheme: Forward Euler
- Invoke a Taylor series expansion of function y(t) around t_{n-1}:

$$y(t_n) = y(t_{n-1}) + h\dot{y}(t_{n-1}) + \frac{1}{2}h^2\ddot{y}(t_{n-1}) + \dots$$

- Drop terms with powers of h of order 2 and higher
- Note that I won't use $y(t_n)$ anymore, but I will now use y_n
  - Make sure you understand this distinction between the exact value and approximate value…

[Basic Concepts]

Truncation Error

[Preliminaries]

- I had this:

$$y(t_n) = y(t_{n-1}) + h\dot{y}(t_{n-1}) + \frac{1}{2}h^2\ddot{y}(t_{n-1}) + \dots$$

Truncation error

- But compute my solution like this

$$y_n = y_{n-1} + h\dot{y}_{n-1}$$

- Or equivalently,

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

- It's fair to expect that there is a slight difference between the exact value and numerical approximation:

[Basic Concepts]

Truncation Error

[Definition]

- Consider how the solution is obtained:

$$\frac{y_n - y_{n-1}}{h} - f(t_{n-1}, y_{n-1}) = 0$$

- Note that in general, if you stick the actual solution in the equation above it is not going to be satisfied:

$$\frac{y(t_n) - y(t_{n-1})}{h} - f(t_{n-1}, y(t_{n-1})) \neq 0$$

- By <u>definition</u>, the quantity above is called the truncation error and is denoted by

$$\mathcal{N}(y, t_n, h) = \frac{y(t_n) - y(t_{n-1})}{h} - f(t_{n-1}, y(t_{n-1}))$$

- Note that this depends on the function (y), the point where you care to evaluate the truncation error (t_n), and the step size used (h)

[Basic Concepts]

Accuracy Order

- Important remark:
  - Note that the truncation error depends on the integration scheme used to compute y^h:

$$\mathcal{N}(y, t_n, h)_{F.Euler} \neq \mathcal{N}(y, t_n, h)_{Runge-Kutta}$$

- Definition: an integration scheme is said to be accurate (or consistent) of order p for a positive p if

$$\mathcal{N}(y, t_n, h) = \mathcal{O}(h^p)$$

- By the way, a quantity "d" is order h^p if there is a constant C so that when h is small enough, one has that

$$|d| \leq C\, h^p$$

[Basic Concepts]

Accuracy Order

- Exercise:
  - Prove that Forward Euler is order 1 accurate, that is,

$$\mathcal{N}(y, t_n, h)_{F.\ Euler} = \mathcal{O}(h)$$

[Basic Concepts]

Convergence

- Important question:
  - Does the numerical solution y^h actually converge to the unique solution of the IVP?
  - "Converge" means as h gets smaller and smaller, do you see the numerical solution at each t_n approaching the exact solution?

- More formal way to frame the convergence concept
  - Define

$$e_n = |y_n - y(t_n)|, \qquad n = 1, 2, ..., N$$

  - Note that N·h=b (as h goes to zero, N keeps growing…)

- The numerical integration is said to be convergent of order p if

$$e_n = \mathcal{O}(h^p), \qquad n = 1, 2, ..., N$$

Zero-stability (0-stability)

- There is a relationship between Accuracy and Convergence
- To capture this relationship we need to introduce the concept of zero-stability
- Definition: the numerical discretization scheme is 0-stable if there are positive constants h_0 and K such that for any solutions x^h and z^h, obtained for $h<h_0$, one has that

$$|x_n - z_n| \leq K \left(|x_0 - z_0| + \max_{1\leq j\leq N} |\mathcal{N}(x, t_j, h) - \mathcal{N}(z, t_j, h)| \right), \qquad 1 \leq n \leq N$$

Exercise

- Prove that Forward Euler is a 0-stable discretization scheme

Order "p" Convergence

- Theorem:

$$\text{Consistency} \quad + \quad \text{0-stability} \quad \Rightarrow \quad \text{Convergence}$$

- Some more specifics:
    - If the method is accurate of order p and 0-stable, then it is convergent of order p:

$$e_n = \mathcal{O}(h^p), \qquad n = 1, 2, ..., N$$

[Basic Concept]
Local Error

- Imagine it as being the error that is noticed in *one* integration step

- Specifically, you start at time step t_{n-1}, from the point y_{n-1}
  - The numerical solution finds for t_n the approximate value y_n (no surprise here)
  - The analytical solution that passes through the initial value y_{n-1} produces at t_n the value $\bar{y}_n$:

$$\begin{cases} \dot{\bar{y}} & = & f(t, \bar{y}) \\ \bar{y}(t_{n-1}) & = & y_{n-1} \end{cases}$$

- The local error is the difference

$$l_n = \bar{y}(t_n) - y_n$$

- Note the relationship that exists between the local error and local truncation error:

$$h \cdot |\mathcal{N}(\bar{y}, t_n, h)| = |l_n| \cdot (1 + \mathcal{O}(h))$$

[Basic Concept]
Absolute Stability

- Convergence is good, but it tells me what happens if I work with smaller and smaller step-sizes h

- In reality, I want to operate with values of h that are large
  - Recall that if h is small you have to take a very large number of integration steps to cover the interval [0,b]

- The relevant question: How large can I consider h yet know for a fact that I don't get garbage approximations of the solution?

  - The answer to this question is posed to each numerical discretization scheme
  - Moreover, it is posed in conjunction with a test problem:

$$\left\{ \begin{array}{rcl} \dot{y} & = & \lambda y \\ y(0) & = & 1 \end{array} \right.$$

[Basic Concept, Short Detour]

Absolute Stability

- Example: mass-spring-damper oscillation

$$\begin{aligned}&\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = 0\\ &x(0) = x_0\\ &\dot{x}(0) = \dot{x}_0\end{aligned}$$

- Case 1: underdamped system

$$\lambda_1 = -\omega_n\zeta - j\omega_d \qquad \lambda_2 = -\omega_n\zeta + j\omega_d$$

- Case 2: overdamped system

$$\lambda_1 = -\omega_n\zeta - \omega_n\sqrt{\zeta^2 - 1} \qquad \lambda_2 = -\omega_n\zeta + \omega_n\sqrt{\zeta^2 - 1}$$

- Case 3: critically damped system

$$\lambda_1 = \lambda_2 = -\omega_n$$

[Basic Concept]

Absolute Stability

- Example, the concluding remark:
  - In mechanical engineering, λ assumes values for which typically

$$\Re(\lambda) < 0$$

- It better be that our numerical discretization scheme leads to numerical solutions that can handle the test IVP for negative values of λ (or its real part, when dealing with complex values…)

- It turns out that this is not trivial
  - Take Forward Euler for a spin to see how problems crop up…

Example:
(λ=-100)

$$\left.\begin{array}{c}\dot{y} = -100y \\ y(0) = 1\end{array}\right\} \Rightarrow y(t) = e^{-100\,t}$$

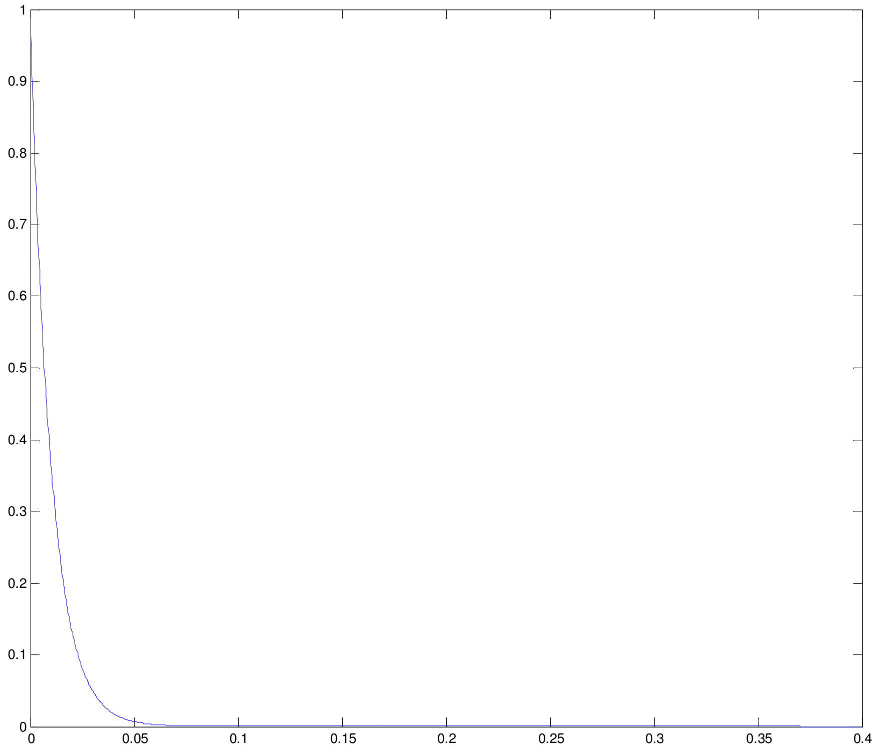
!

- Integrate 5 steps using <u>forward Euler</u> formula: Δt=0.002, Δt=0.01, Δt=0.03
- Compare the errors between numerical and analytical solutions

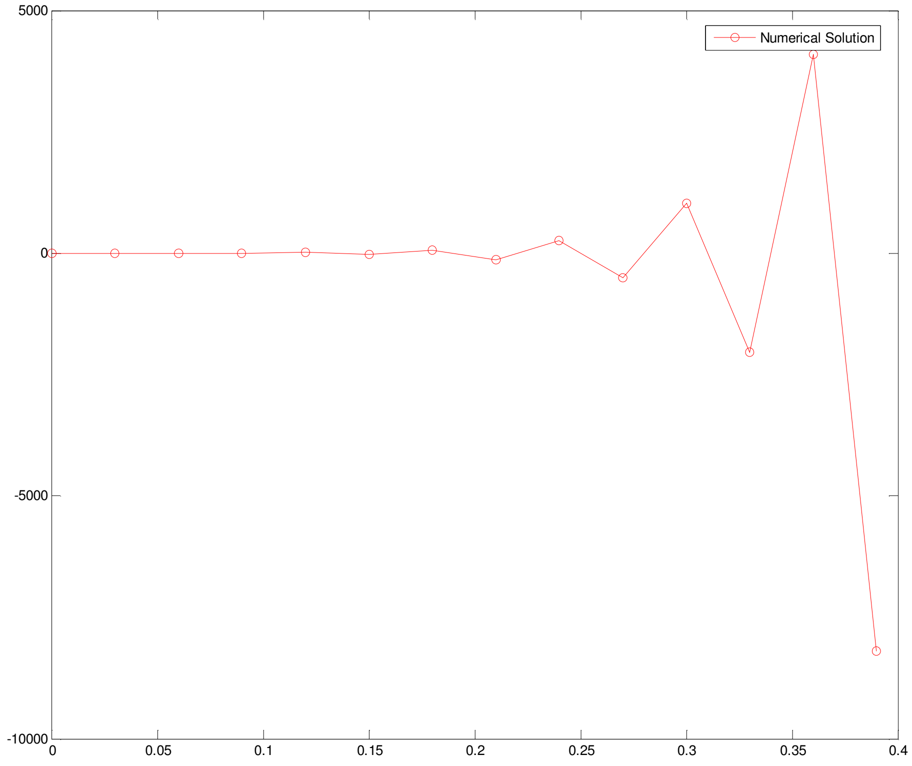
Δt=0.002: Error	Δt=0.01: Error	Δt=0.03: Error
0	0	0
0.01873075307798	0.367879441171 44	2.04978706836786
0.03032004603564	0.13533528323661	-3.99752124782333
0.03681163609403	0.04978706836786	8.00012340980409
0.03972896411722	0.01831563888873	-15.99999385578765
0.04019944117144	0.00673794699909	32.00000030590232 !

Example:
(λ=-100)

$$\left.\begin{aligned}\dot{y} &= -100y \\ y(0) &= 1\end{aligned}\right\} \Rightarrow y(t) = e^{-100\,t}$$



Analytical Solution



Forward Euler

(Δt=0.03)

Basic Concept: Absolute Stability

- One quick way to see that things went south
- First note that
  - if $\Re(\lambda) < 0$, then $0 < y(t_n) < y(t_{n-1}) < \ldots < y(t_1) < y(t_0) = 1$
- You'd expect that the numerical solution will mirror this behavior:

$$0 < y_n < y_{n-1} < \ldots < y_1 < y_0 = 1$$

- Use Forward Euler to express y_n as a function of y_{n-1} and require that the condition $y_n<y_{n-1}$ to obtain that

$$|1 + h\lambda| < 1$$

Basic Concept: Absolute Stability

- Recall what happened
  - We started with the test problem
  - We required that for the test problem, the numerical approximation should behave like the solution. That is, we required that

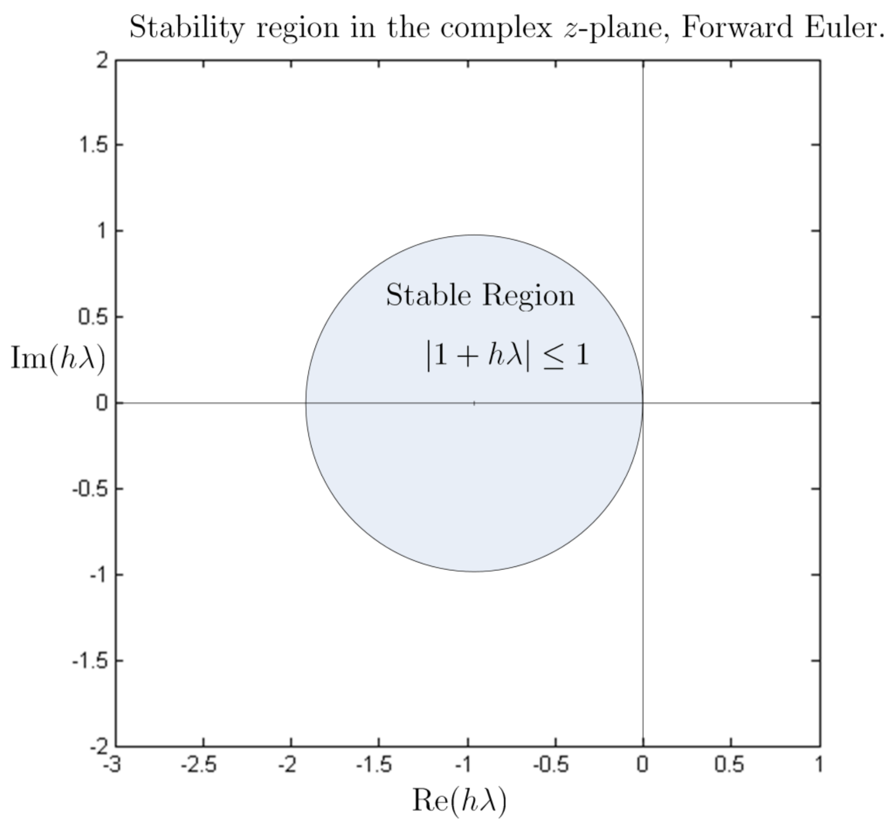
$$y_n \leq y_{n-1}$$

  - We used the discretization scheme (Forward Euler, in our case) to express how y_n is related to y_{n-1}
  - This led to a condition that the step size should satisfy, specifically, for Forward Euler, we obtained that

$$|1 + h\lambda| < 1$$

  - What we just did was to determine the region of absolute stability of Forward Euler
  - The KEY point: to get the absolute stability region for other integration methods, you have to use that integration method to express how y_n is related to y_{n-1}

Basic Concept: Absolute Stability



- The step size h should be such that hλ lands into the shaded circle
- Note that a very negative value for λ will require a very small value of h so that the product hλ is inside the circle

Accuracy vs. Stability

- Note that Forward Euler is accurate to order 1
  - That is, locally,

$$\mathcal{N}(y, t_n, h)_{F.\ Euler} = \mathcal{O}(h)$$

- This is an asymptotic and *local* result, which holds better as h gets smaller
- For the test IVP, the local error compounds due to the particular form of the problem that we work with (the test IVP)
- This compounding and the fact that h does not assume small values (you try to work w/ large h) leads to the phenomenon of loss of stability

- To conclude, there is no contradiction here (the numerical scheme being order 1 accurate yet losing stability for large values of h)

Accuracy vs. Stability: The Concept of Stiffness

- Recall that the size/shape of the stability region (SR) is specific to each discretization scheme

- For some discretization schemes the SR is ridiculously small
  - Forward Euler is one of them

- A small SR to start with, combined with a problem for which λ is very negative leads to unreasonably small values of h
  - Such a problem is called a “stiff” IVP

- In this case it is *not* the accuracy concerns that restrict the value of the step-size h, but rather the stability issue prevails

- Note that it can be the case that if you change the integration method, the stiff problem is just fine (if the stability region is generous)

Example:

- Use Forward Euler to find an approximation of the solution of the following IVP:

$$\begin{cases} \dot{y} = -100y + \sin(t) \\ y(0) = 0 \end{cases} \qquad t \in [0, 8]$$

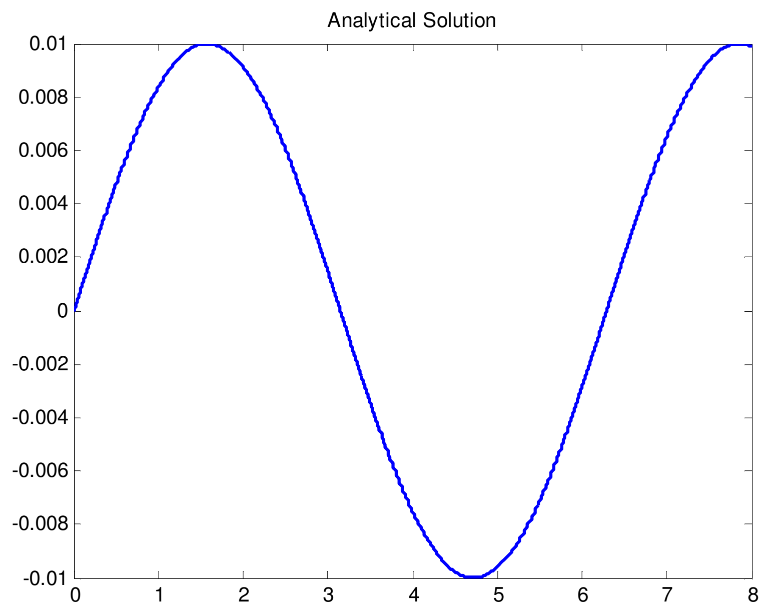
Example [Cntd.]

- Easy to figure out that the exact solution is

$$y(t) = \frac{1}{10001}\left(100\sin(t) - \cos(t) + e^{-100t}\right)$$

- For all purposes, the solution can be rewritten as

$$y(t) \approx 0.01\sin(t-\phi_0) \qquad \text{where} \qquad \tan\phi_0 = \frac{1}{100}$$

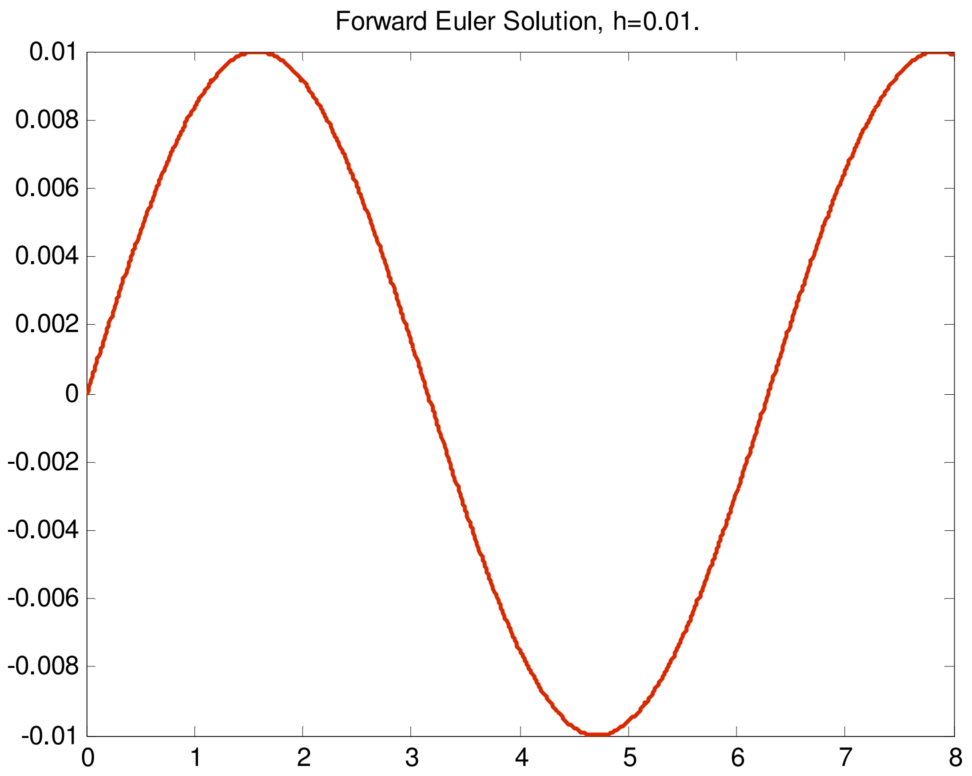


Example [Cntd.]

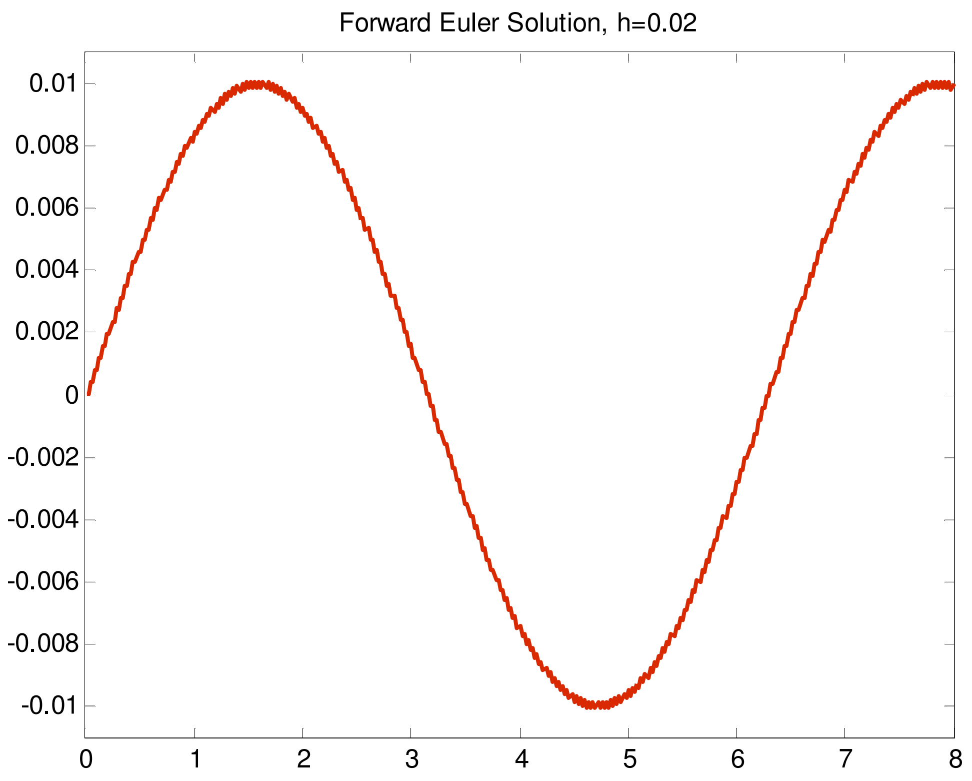
- Note that for the given IVP, λ=-100, which suggests we'll have to work with small step-sizes…
- Note in the plot that the contribution of the exponential is not even seen in the MATLAB plot
- You'd expect that since the exponential component of the solution goes away so quickly one could use Forward Euler and have no difficulties, which is not true… (see next slide)

```
% Solves the IVP y_dot = -100*y + sin(t) and y(0)=0
% yExact is the analytical solution
% yFE is the solution obtained with Forward Euler
% yBE is the solution obtained with Backward Euler
%
% Input: the integration step-size

h = input('Input step size h:');
tend = 8;
tm=0:h:tend;

yExact = 1/10001*(100*sin(tm)-cos(tm)+exp(-100*tm));

yFE = zeros(size(tm'));
yBE = zeros(size(tm'));

for i=2:1:length(yFE)
   yFE(i) = yFE(i-1)*(1-100*h) + h*sin(tm(i-1));
end

dummyINV = 1/(1+100*h);
for i=2:1:length(yBE)
   yBE(i) = yBE(i-1)*dummyINV + h*dummyINV*sin(tm(i));
end
```

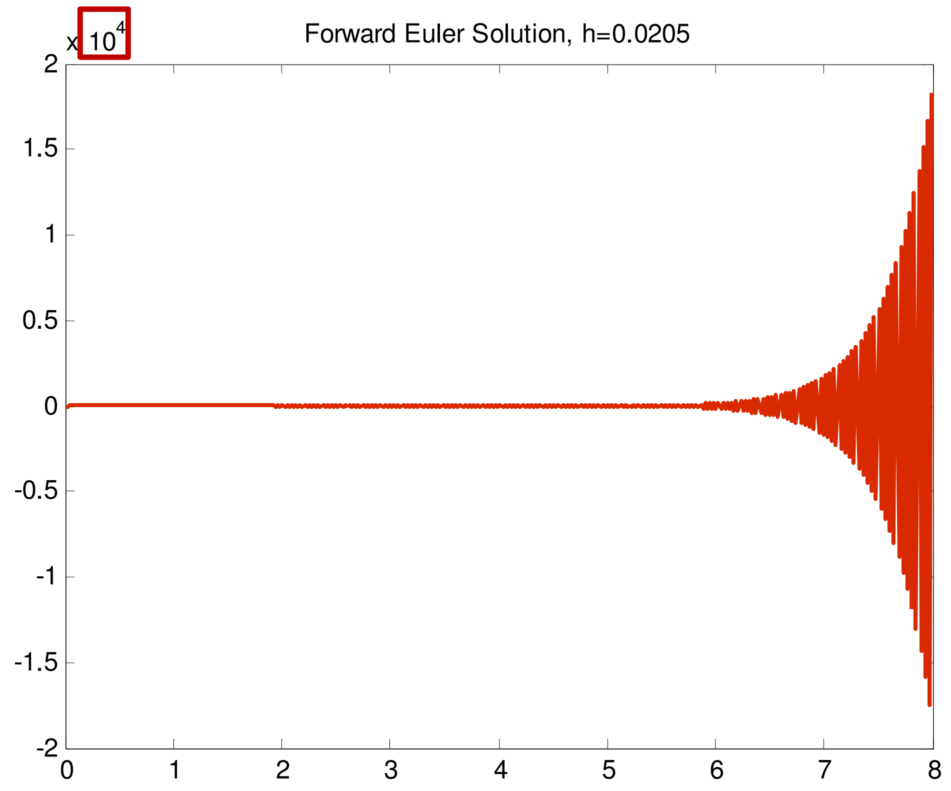
Example, Approached with Forward Euler: h=0.01 s



Example, Approached with Forward Euler : h=0.02 s



Example, Approached with Forward Euler: h=0.0205 s



- As soon as you go beyond the limit value h=0.02 (that goes hand in hand for Forward Euler with \lambda=-100), you run into trouble
- Note that this happens even though the contribution of the exponential goes away very fast…

Example, Approached with Forward Euler

- Conclusion
  - For this type of problem with very negative λ, Forward Euler is bad
    - The step size is significantly limited on stability grounds
- Qualitative definition:
  - An IVP where Forward Euler behaves bad is called STIFF IVP