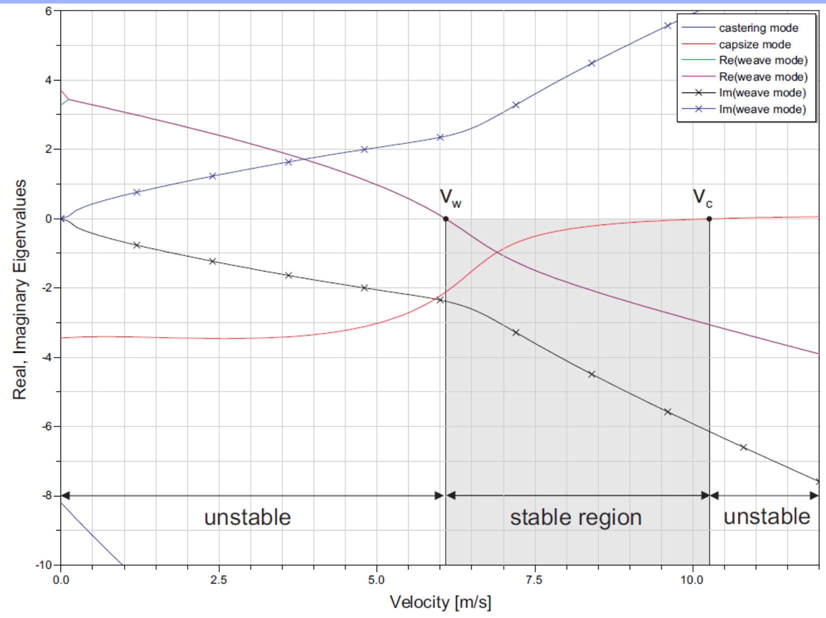
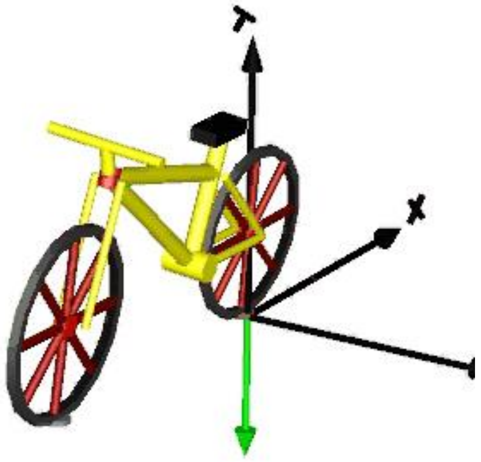
Virtual Physics
Equation-Based Modeling

TUM, January 10, 2023

Numerical Stability and Stiffness





Dr. Dirk Zimmer

German Aerospace Center (DLR), Robotics and Mechatronics Centre

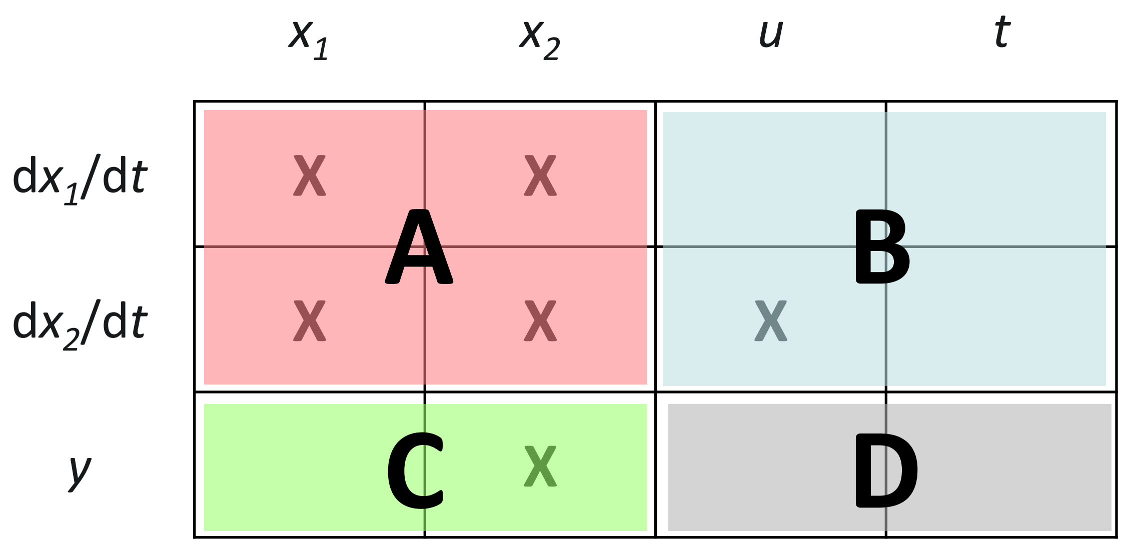
Recap: State-Space Form

Robotics and Mechatronics Centre

- We have learned that for linear systems, the state-space form can be described by four sub-matrices **A**, **B**, **C**, and **D**.

$$d\mathbf{x}/dt = \mathbf{A}\mathbf{x} + \mathbf{B}(\mathbf{u}, t)$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}(\mathbf{u}, t)$$



	x_1	x_2	u	t
dx_1/dt	X	X		
dx_2/dt	X	X	X	
y		X		

Recap: State-Space Form

- For the actual dynamics, the **A** matrix is for primary interest.

$$d\mathbf{x}/dt = \mathbf{A}\mathbf{x}$$

	x_1	x_2	u	t
dx_1/dt	X	X		
dx_2/dt	X	X	X	
y		X		

A

Recap: Stability in 1D Systems

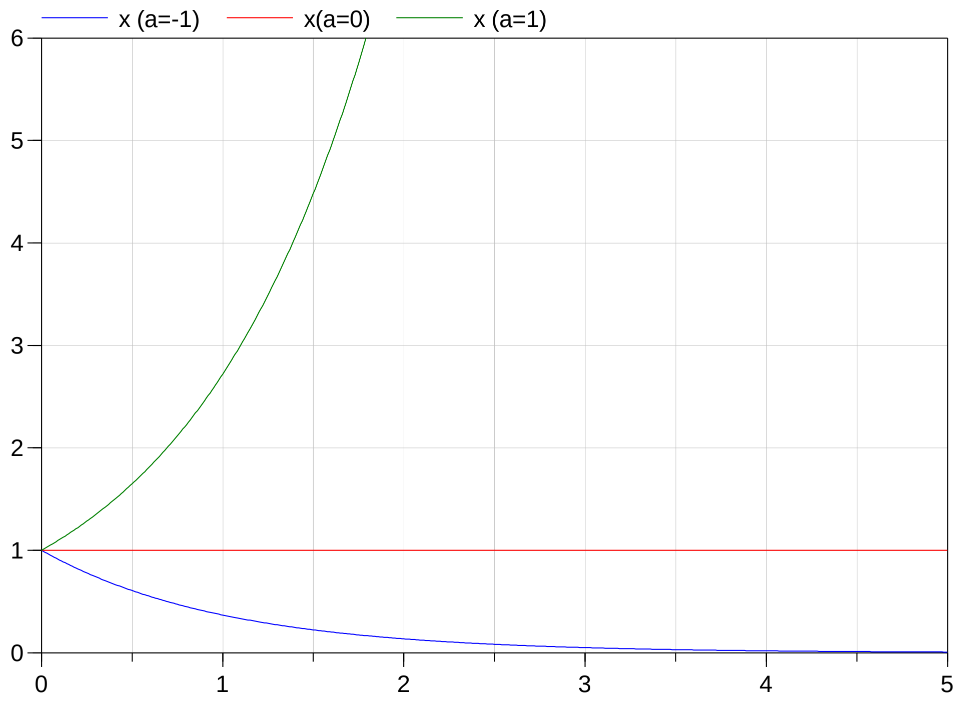
Robotics and Mechatronics Centre

Let us look at the general one-dimensional linear system

$$dx/dt = ax \quad \textit{(with } x_{start} = 1\textit{)}$$

Definition:

- a > 0:
  The system is ***unstable***
- a < 0:
  The system is ***stable***
- a = 0:
  The system is ***marginally stable***



Recap: Stability in *n*D Systems

What about the multi-dimensional case?

$$d\mathbf{x}/dt = \mathbf{A}x$$

- We can perform an eigenvalue decomposition:

  $$\mathbf{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{-1}$$

  where **Q** consists in the eigenvectors and **Λ** is a diagonal matrix containing the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$.

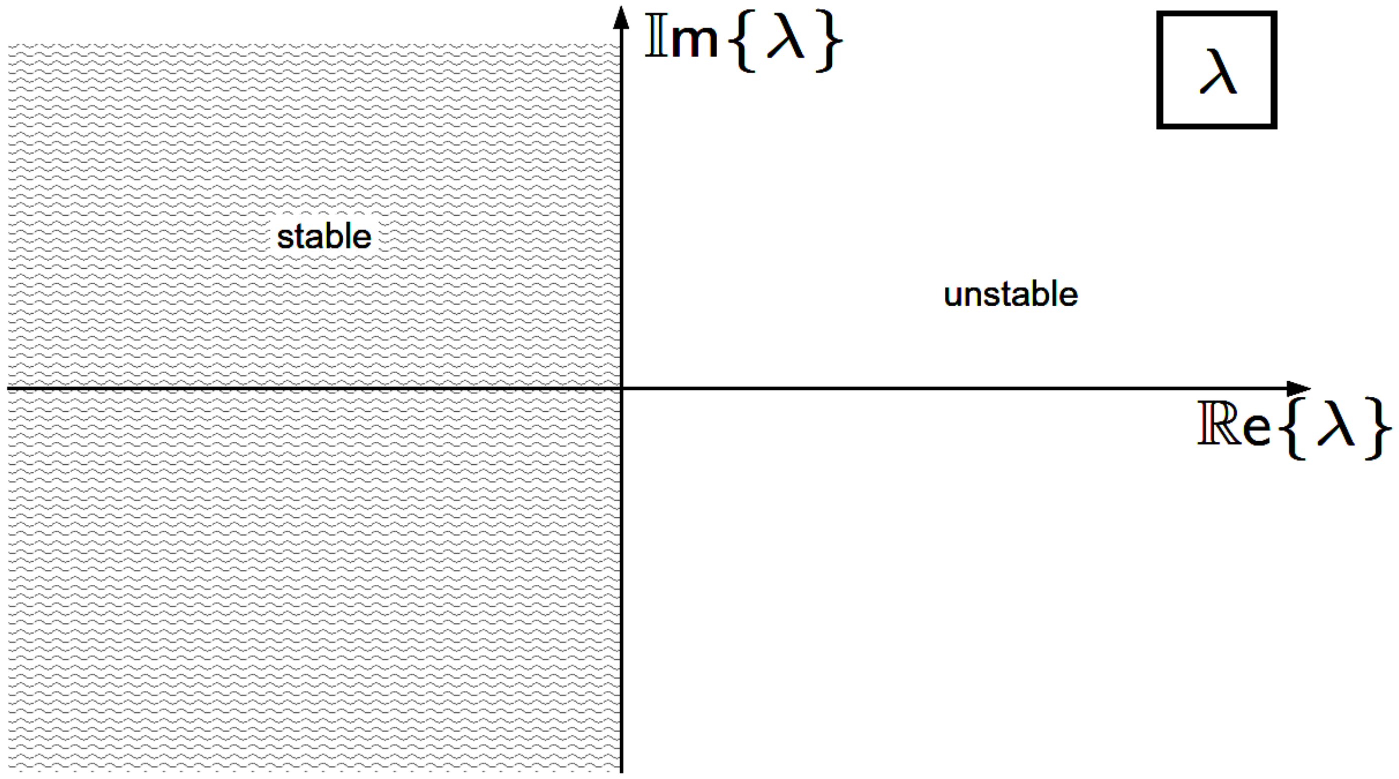
- The system is ***asymptotically stable*** iff all eigenvalues are smaller than 0.

- The system is ***marginaly stable*** if all eigenvalues are smaller or equal than 0.

- The system is ***unstable*** otherwise.

- For complex eigenvalues, the real part is decisive

Domain of Analytical Stability


$\mathbb{I}\mathrm{m}\{\lambda\}$
$\boxed{\lambda}$
stable
unstable
$\mathbb{R}\mathrm{e}\{\lambda\}$

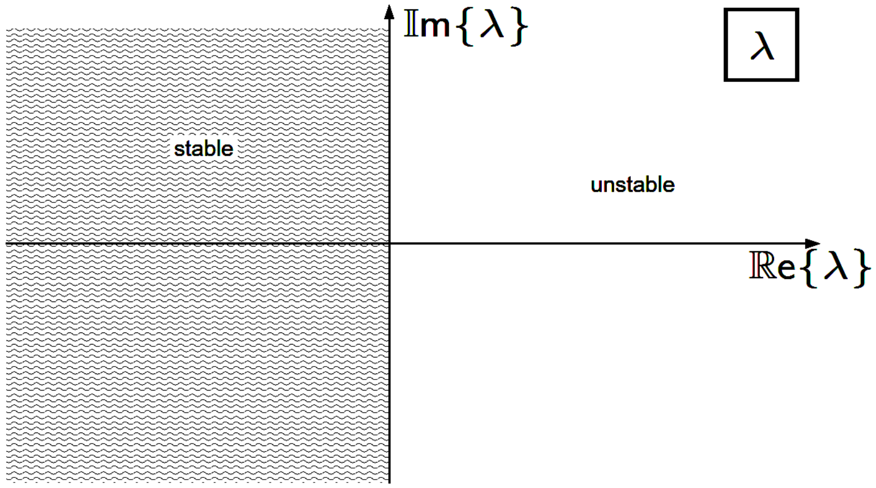
Analytical vs. Numerical Stability

So far, we have examined the analytical stability of our system. But there is a crucial question for the simulation of a system:

Can the applied scheme for time-integration change the stability of the system?

- Let us start with an analysis of Forward Euler (FE):

- Remember:


$\mathbb{I}\mathrm{m}\{\lambda\}$
λ
stable
unstable
$\mathbb{R}\mathrm{e}\{\lambda\}$

Numerical Stability: FE

Let us apply the Forward Euler scheme on the system:

$$d\mathbf{x}/dt = f(\mathbf{x}(t),t) = \mathbf{A}\mathbf{x}(t)$$

- By discretizing time with step-width *h* we get:

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h \cdot f(\mathbf{x}(t),t)$$

$$\Rightarrow \quad \mathbf{x}(t+h) = \mathbf{x}(t) + h \cdot \mathbf{A}\mathbf{x}(t)$$

$$\Rightarrow \quad \mathbf{x}(t+h) = (\mathbf{I} + h\mathbf{A})\,\mathbf{x}(t)$$

Numerical Stability: FE

Robotics and Mechatronics Centre

Let us apply the Forward Euler scheme on the system:

$$d\mathbf{x}/dt = f(\mathbf{x}(t),t) = \mathbf{A}\mathbf{x}(t)$$

- By discretizing time with step-width *h* we get:

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h \cdot f(\mathbf{x}(t),t)$$

$$\Rightarrow \quad \mathbf{x}(t+h) = \mathbf{x}(t) + h \cdot \mathbf{A}\mathbf{x}(t)$$

$$\Rightarrow \quad \mathbf{x}(t+h) = (\mathbf{I} + h\mathbf{A})\,\mathbf{x}(t)$$

- Applying FE transforms the continuous function **x**(t) into a series $\mathbf{x}_k$.

$$\Rightarrow \quad \mathbf{x}_{k+1} = (\mathbf{I} + h\mathbf{A})\,\mathbf{x}_k$$

$$\Rightarrow \quad \mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k \qquad \text{with} \qquad \mathbf{F} = \mathbf{I} + h\mathbf{A}$$

Numerical Stability: FE

When is this series numerically stable?

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k$$

- This is the case when all eigenvalues of **F** are located inside the unit circle of complex numbers.
- Each integration scheme has its own formula for **F**. In case of FE, it is:

  $\mathbf{F} = \mathbf{I} + h\mathbf{A}$

Numerical Stability: FE

When is this series numerically stable?

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k$$

- This is the case when all eigenvalues of **F** are located inside the unit circle of complex numbers.
- Each integration scheme has its own formula for **F**. In case of FE, it is:

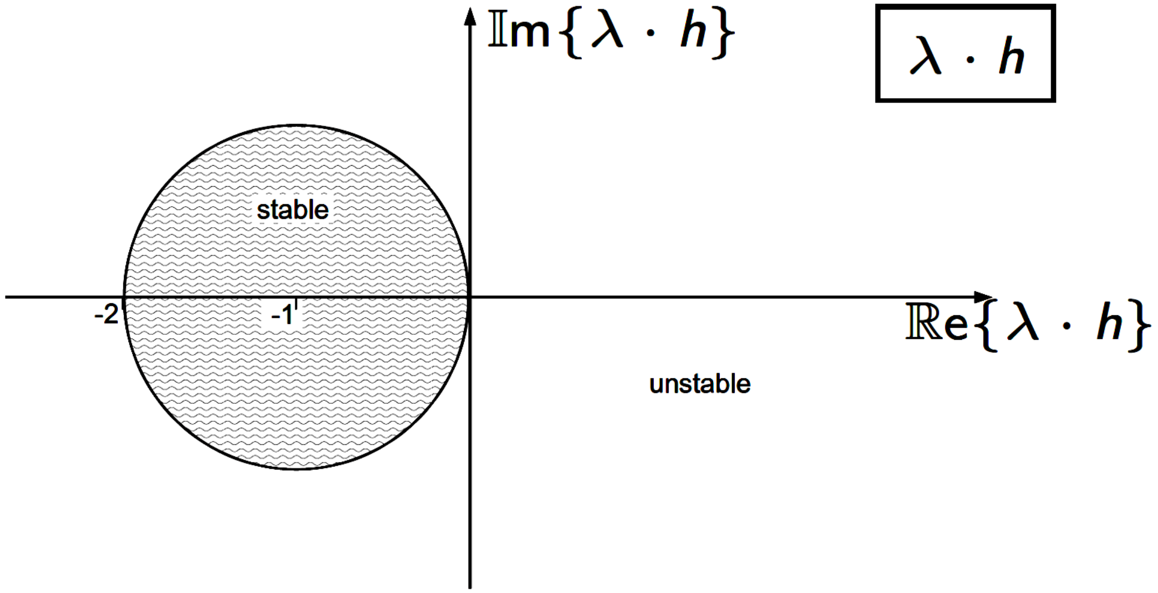
  $\mathbf{F} = \mathbf{I} + h\mathbf{A}$
- We would like to express the eigenvalues of **F** in terms of the eigenvalues of **A**

$$F = \mathbf{I} + h\mathbf{A}$$
$$\Rightarrow \quad F = \mathbf{QIQ}^{-1} + h\mathbf{Q\Lambda Q}^{-1}$$
$$\Rightarrow \quad F = \mathbf{Q}(\mathbf{I} + h\mathbf{\Lambda})\,\mathbf{Q}^{-1}$$

- By assuming h=1, the eigenvalues of **F** are the eigenvalues of A plus 1.0 + 0i.

Numerical Stability: FE

By applying FE, the stability region of the system **A** has been reduced from the half-plane to a circle of radius $1/h$ (in the plot, the axes are scaled by h).

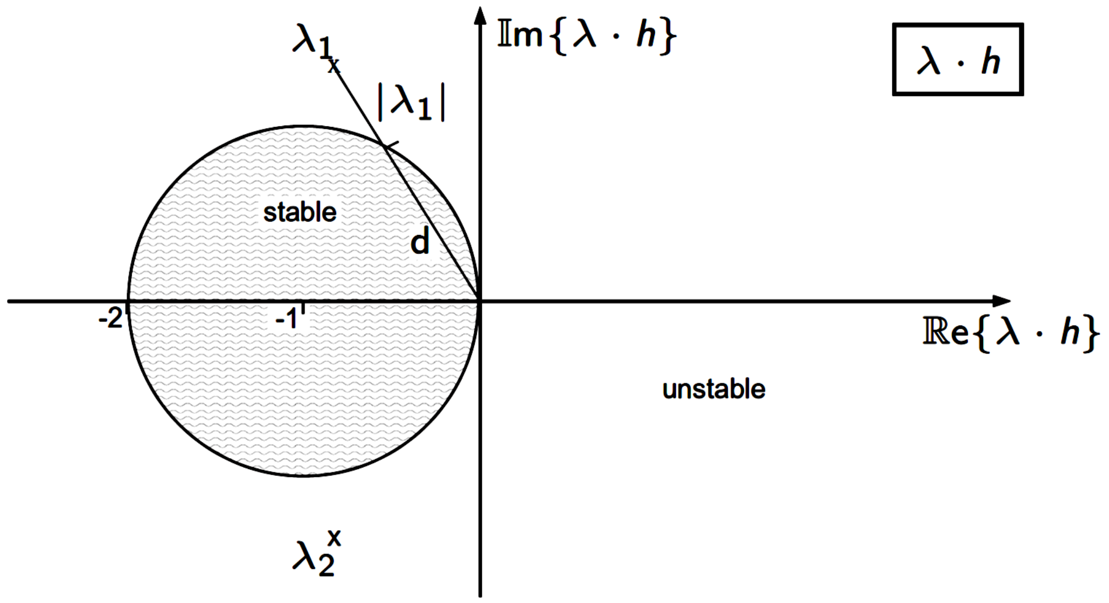


$\mathbb{I}\mathrm{m}\{\lambda \cdot h\}$
$\boxed{\lambda \cdot h}$
stable
-2
-1
$\mathbb{R}\mathrm{e}\{\lambda \cdot h\}$
unstable

- Forward Euler is potentially destabilizing.

Numerical Stability: FE

To get a qualitative correct result for a stable system, we have to choose *h* small enough, so that all eigenvalues are located inside the stable region.


$\mathbb{I}\mathrm{m}\{\lambda \cdot h\}$
$\lambda \cdot h$
λ_1
$|\lambda_1|$
stable
d
-2
-1
$\mathbb{R}\mathrm{e}\{\lambda \cdot h\}$
unstable
λ_2

$$\rightarrow \quad h_{\max} = d/|\lambda_1|$$

Backward Euler

There is an alternative Euler scheme: Backward Euler.

- In contrast to Forward Euler, we take the time-derivative of t+*h*:

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h \cdot f(\mathbf{x}(t+h), t+h)$$

- If we use Backward Euler to integrate forward in time, we have to solve this equation system for **x**(t+*h*) by an iterative solver (Newton's method or similar) with the residual equation:

$$0 = \mathbf{x}(t) + h \cdot f(\mathbf{x}(t+h), t+h) - \mathbf{x}(t+h)$$

- In contrast to Forward Euler, there is no ***explicit*** computational form available. Hence Backward Euler is an ***implicit*** method.
(FE is an explicit method)

Numerical Stability: BE

What is the numerical stability domain of Backward Euler?

- The answer can be derived without needing any formulas.
- If we compute backwards in time, stable systems become unstable and vice versa.
- If we compute backwards in time, BE becomes computational identical to FE (computed forward in time)
- Hence the stability domain is simply flipped to the other side.

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h \cdot f(\mathbf{x}(t+h), t+h)$$

$$\Rightarrow \quad \mathbf{x}(t+h) = \mathbf{x}(t) + h \cdot \mathbf{A}\mathbf{x}(t+h)$$

$$\Rightarrow \quad (\mathbf{I} - h\mathbf{A})\,\mathbf{x}(t+h) = \mathbf{x}(t)$$

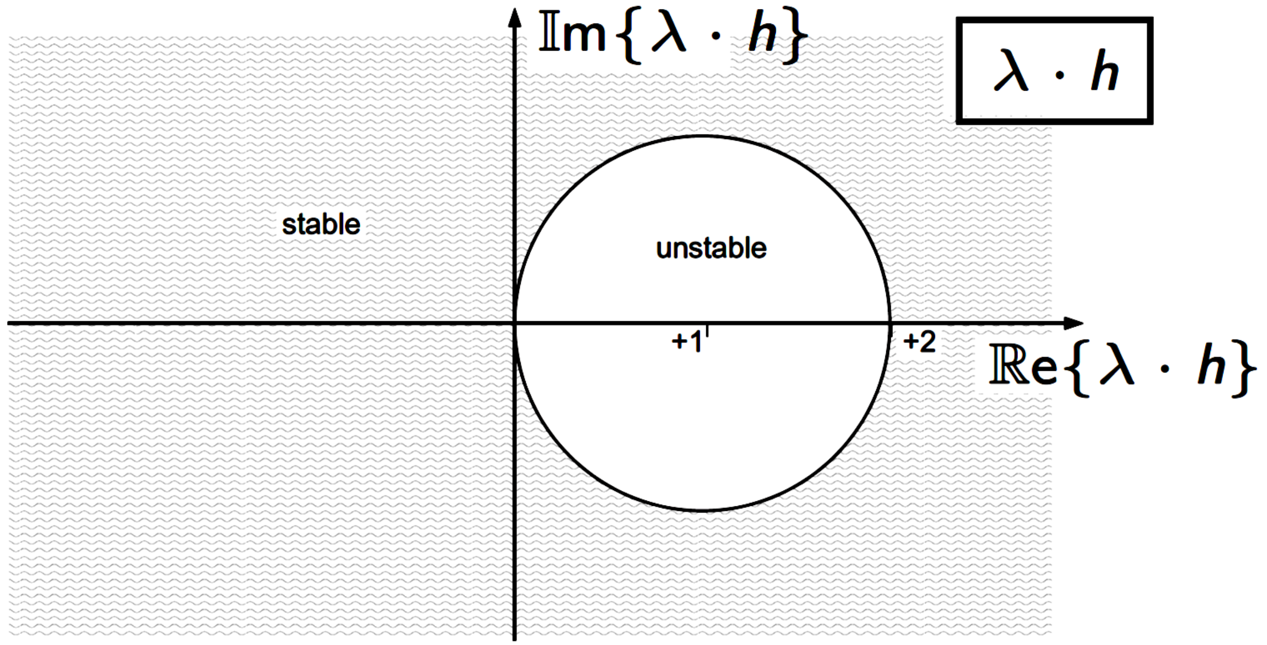
$$\Rightarrow \quad \mathbf{x}(t+h) = (\mathbf{I} - h\mathbf{A})^{-1}\,\mathbf{x}(t)$$

$$\Rightarrow \quad \mathbf{x}_{k+1} = (\mathbf{I} - h\mathbf{A})^{-1}\,\mathbf{x}_k$$

$$\Rightarrow \quad \mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k \qquad \text{with} \qquad \mathbf{F} = (\mathbf{I} - h\mathbf{A})^{-1}$$

Numerical Stability: BE

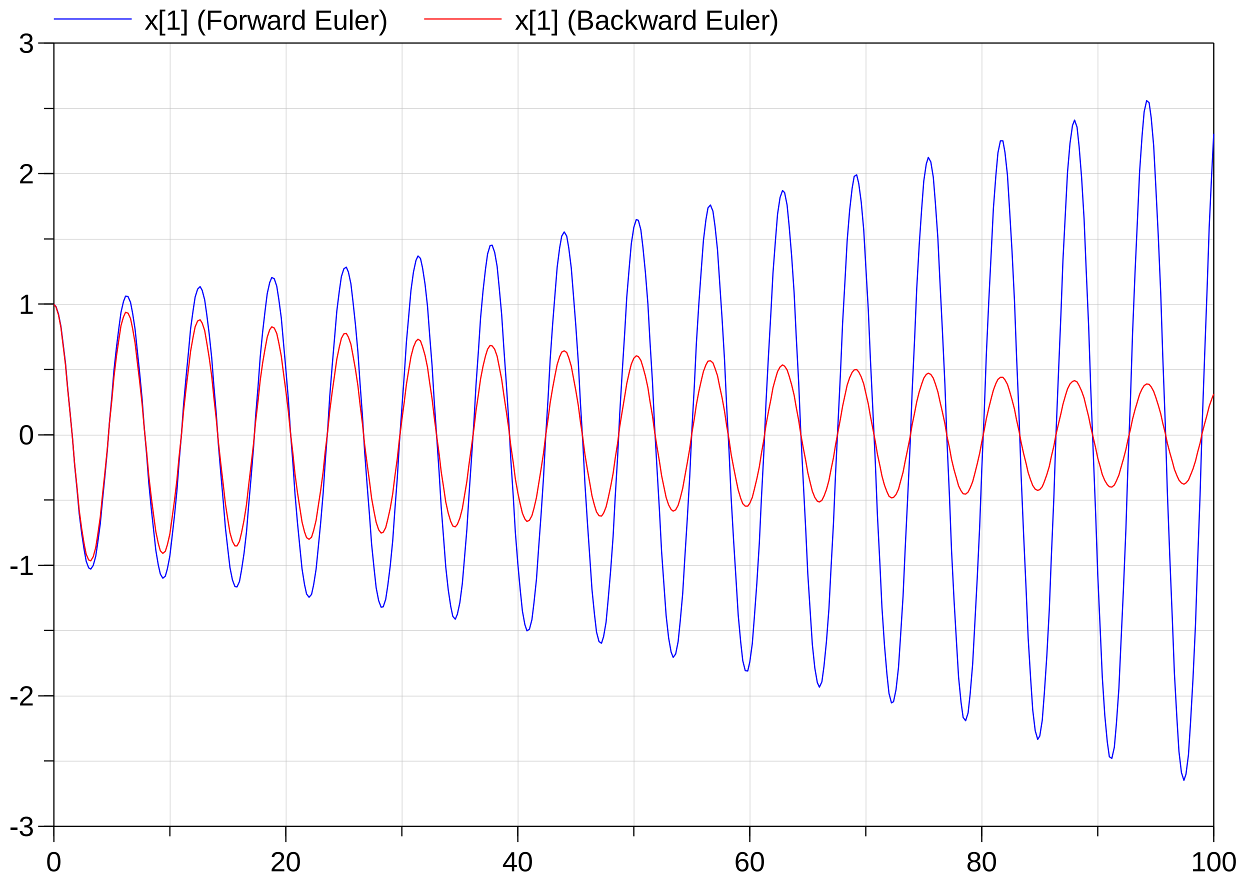
By applying BE, the unstable region in the right-half plane collapses to a circle of radius 1/*h*.


$\mathbb{I}\mathrm{m}\{\lambda \cdot h\}$
$\lambda \cdot h$
stable
unstable
+1
+2
$\mathbb{R}\mathrm{e}\{\lambda \cdot h\}$

- Backward Euler causes a numerical stabilization.

FE vs. BE

FE and BE (h=0.02s) are applied on a marginally stable system.



Fight the Drift

Robotics and Mechatronics Centre

For Real-Time Applications that use FE, it is sometimes possible to fight the destabilizing drift and get better solutions.

- The following system represents a marginally stable spring system:

$$dv/dt = -s \cdot C/M$$
$$ds/dt = v$$

- The total energy in this system is the potential energy in the spring plus the kinetic energy:

$$E = ½ \cdot C \cdot s^2 + ½ \cdot M \cdot v^2$$

Fight the Drift

- We know that energy is conserved hence:

$$E = ½·C·s^2 + ½·M·v^2 = \text{const}$$

- This means that *s* and *v* are always placed on an ellipse. To fight the drift, we can project on this ellipse after each integration step. For this example, it is particularly easy.

- We compute a scaling factor out the current total energy *E* and the initial total energy E0…

$$p = (E0/E)^{½}$$

- …and apply this factor to *s* and *v*

$$s' = s \cdot p$$
$$v' = v \cdot p$$

Fight the Drift

Robotics and Mechatronics Centre

This is the corresponding Python Code

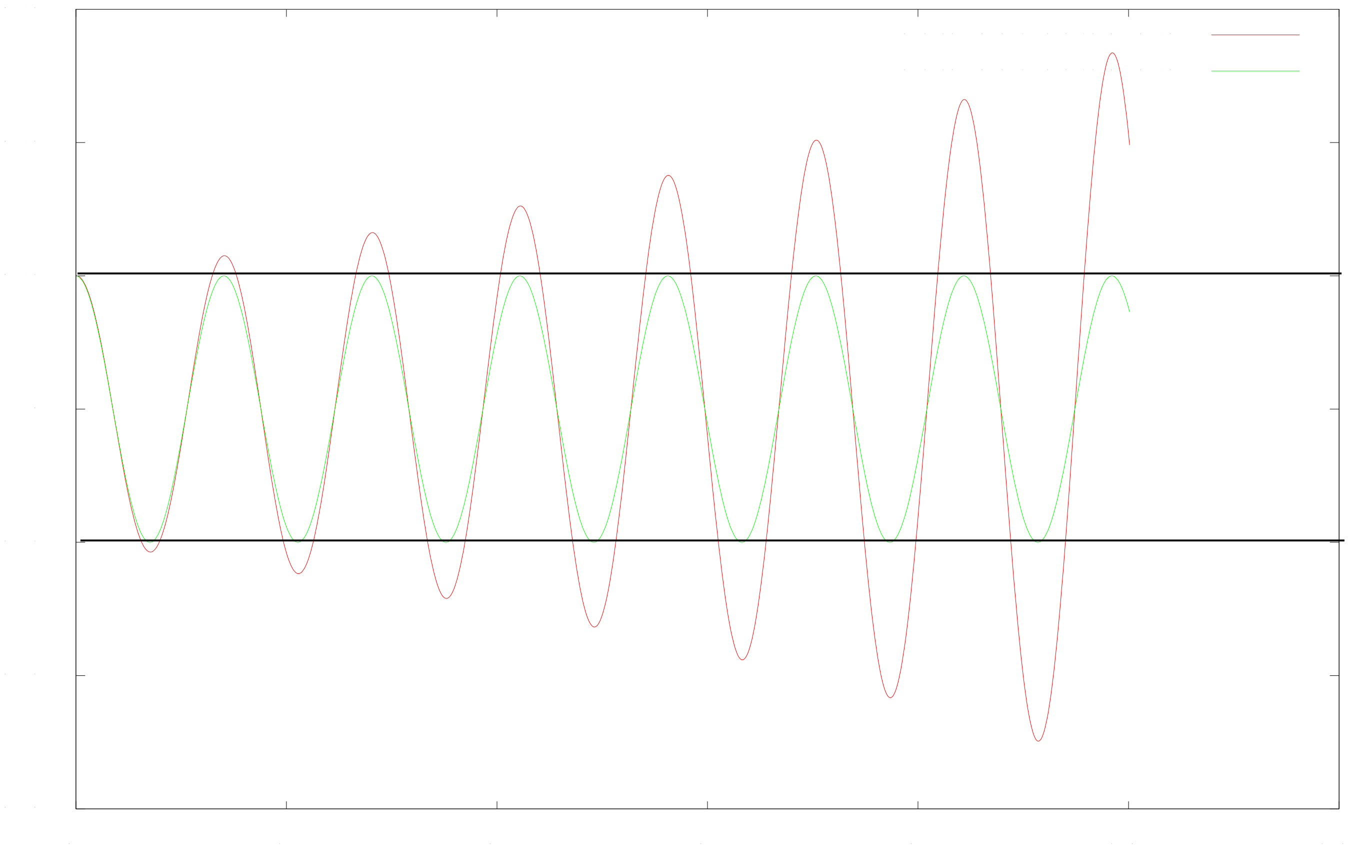
```
open file for ouput
fh = open("out.dat","w")

#perform time-integration
while time < tStop:
      dv_dt = -s*C/M
      ds_dt = v
      v += h*dv_dt
      s += h*ds_dt
      E = 0.5*C*s*s + 0.5*M*v*v
      p = sqrt(E0/E)
      s *= p
      v *= p
      time += h
      print(time,"\t",s,"\t",v,file=fh)

print("See out.dat for simulation result")
fh.close()
```

Fight the Drift

And here the simulation results for FE and FE with projection.



FE vs. BE

Forward Euler and Backward Euler are the origin of all higher-order methods

- All explicit higher-order methods degenerate to FE for $h \rightarrow 0$ (supposing a finite numerical precision).
- All implicit higher-order methods degenerate to BE for $h \rightarrow 0$ (supposing a finite numerical precision).
- Most explicit and implicit methods share somehow their behavior w.r.t stability with their first-order counterparts.
- Both FE and BE are totally unsuited for marginally stable systems, since the stability domain shares only a single point with the imaginary axis. Fortunately, there are higher-order methods that share larger fractions of their stability domain with the imaginary axis.

Stiff Systems

What are stiff systems?

- We call a linear system stiff, if it is stable and its eigenvalues vary a lot in terms of their real parts.
- Non-linear systems are called stiff, if they are stable and exhibit both fast and slow modes in their behavior. The linearization of such systems leads to stiff linear systems.
- These systems cannot be simulated efficiently by means of any explicit method (FE), because we would need very small time steps to move those eigenvalues (of either the system itself or of its linearization) into the numerical stability domain.
- **For the efficient simulation of stiff systems, implicit integration algorithms are required!**

Stiff Systems: Example

Example:

$$\mathbf{A} = \begin{pmatrix} -5.05 & -4.95 \\ -4.95 & -5.05 \end{pmatrix}$$

- We can perform an eigenvalue decomposition:

$$\mathbf{A} = \begin{pmatrix} \sqrt{2}/2 & -\sqrt{2}/2 \\ \sqrt{2}/2 & \sqrt{2}/2 \end{pmatrix} \begin{pmatrix} -10 & 0 \\ 0 & -0.1 \end{pmatrix} \begin{pmatrix} \sqrt{2}/2 & \sqrt{2}/2 \\ -\sqrt{2}/2 & \sqrt{2}/2 \end{pmatrix}$$

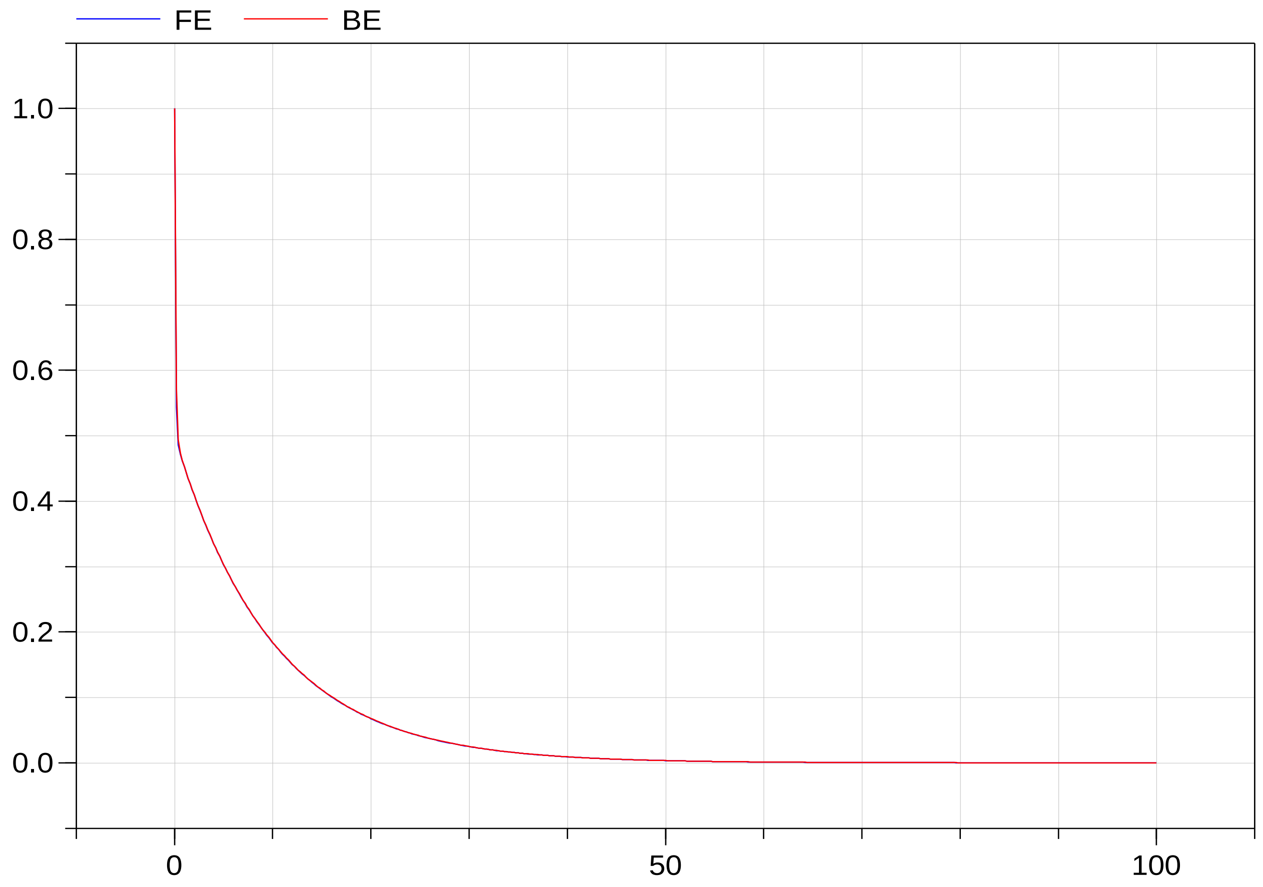
- The system is stable and the eigenvalues are separation by a factor 100. The system is stiff.

- We are going to simulate the system using FE and BE for different step sizes.

Stiff Systems: FE vs. BE

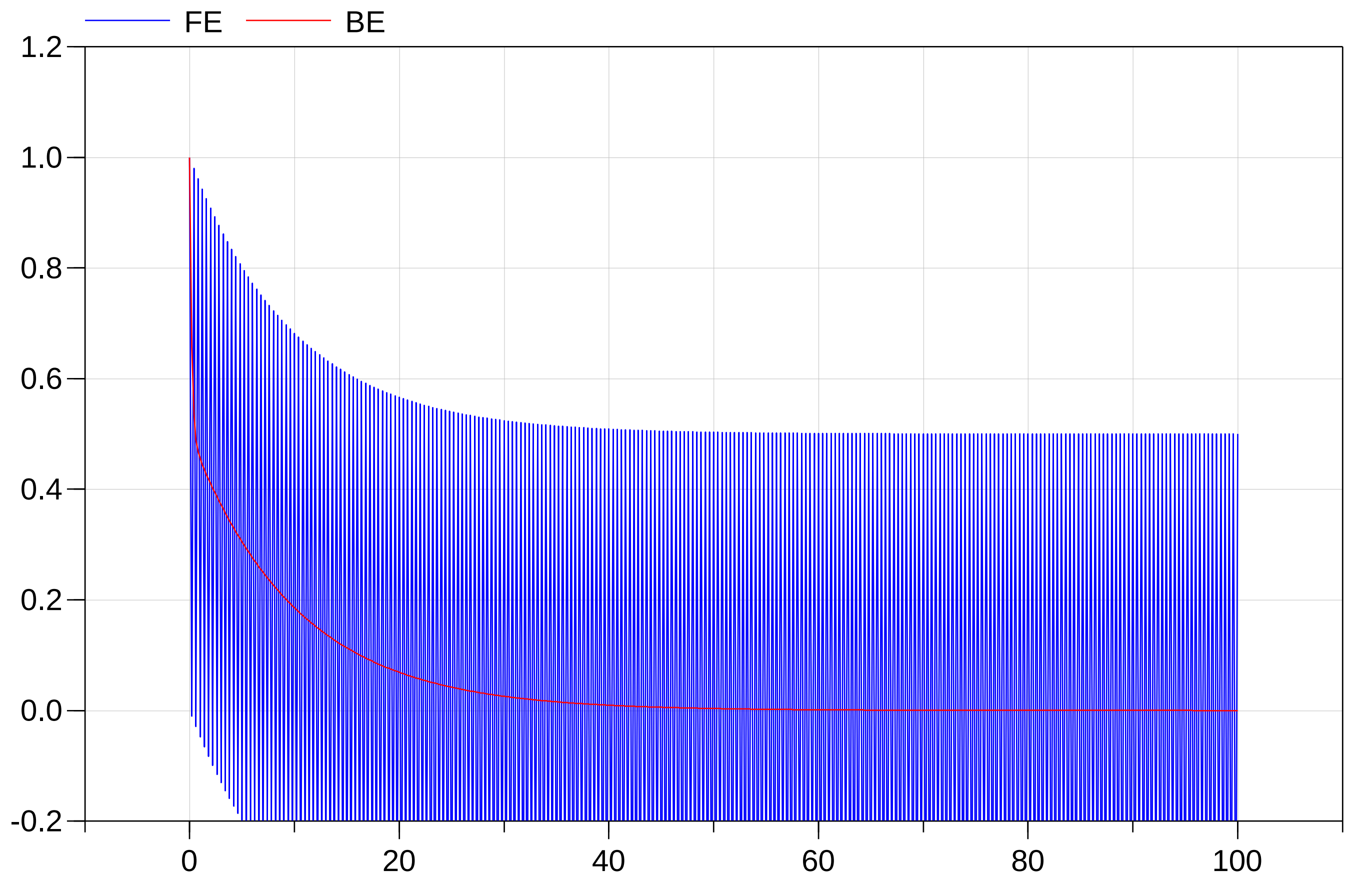
Robotics and Mechatronics Centre

FE and BE are applied on the stiff system (h=0.02).



Stiff Systems: FE vs. BE

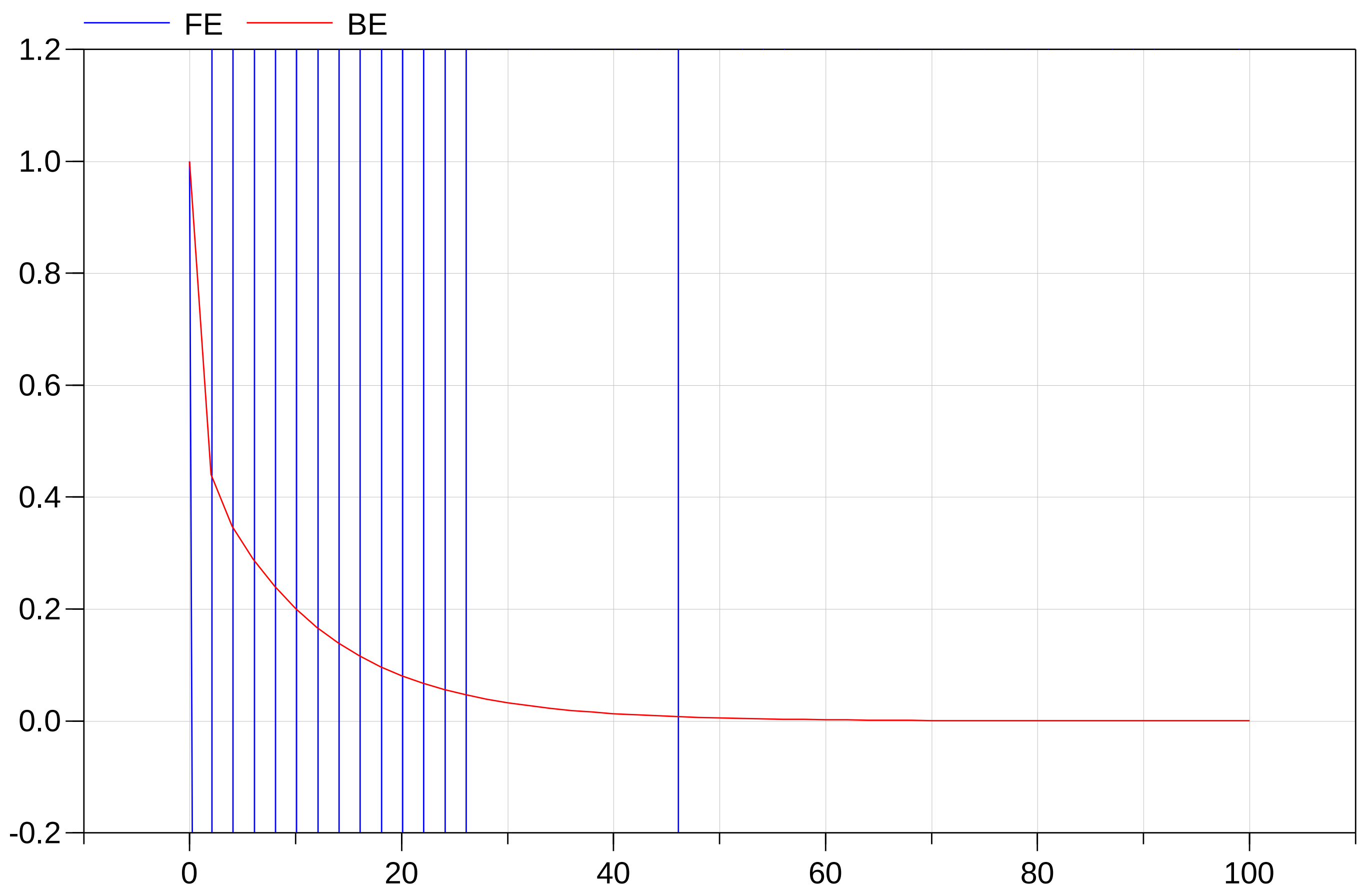
FE and BE are applied on the stiff system (h=0.2).



Stiff Systems: FE vs. BE

Robotics and Mechatronics Centre

FE and BE are applied on the stiff system (h=2).



Questions ?