

Introduction to Simulation

Solving ODEs

Motivation and Goals

Motivation:

- ODEs form the basis of almost all continuous simulations.
- They must almost always be solved numerically – Simulation!
- Many software tools can do this (largely) automatically.

We need to understand the basics of how to solve ODEs,

- To understand the behaviour of the software,
- To avoid common pitfalls.

Contents today:

- The basic ideas in the numerical solution of ODEs

Initial Value Problems

The simulation task:

Given $\frac{dy}{dt} = f(y, t)$
 $y(0) = y_0$

find $y(t), \quad t > 0$

Taylor Series Expansion

Consider a scalar, real-valued function $y = y(t)$.

Represent the value of $y(t + h)$ using derivatives of $y(t)$:

$$y(t + h) = y(t) + hy'(t) + \frac{h^2}{2!} y''(t) + \frac{h^3}{3!} y'''(t) + \dots$$

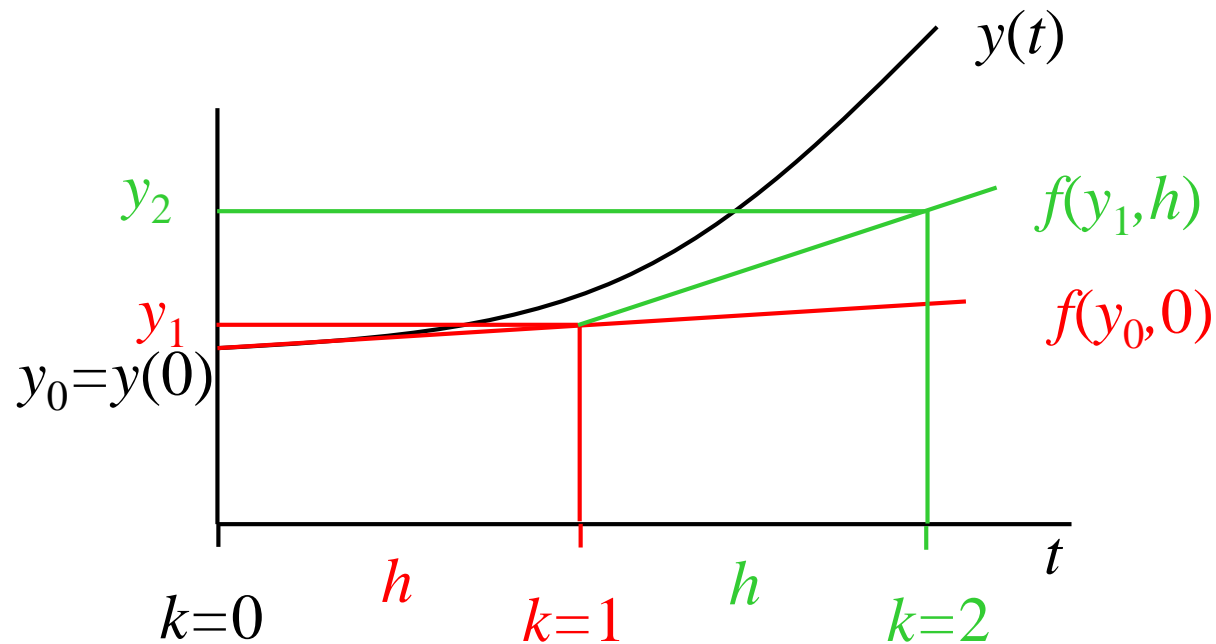
When h is small, we can write:

$$y(t + h) = y(t) + hy'(t) + \frac{h^2}{2!} y''(t) + \frac{h^3}{3!} y'''(t) + O(h^4)$$

Euler's Method

The simplest numerical integration method:

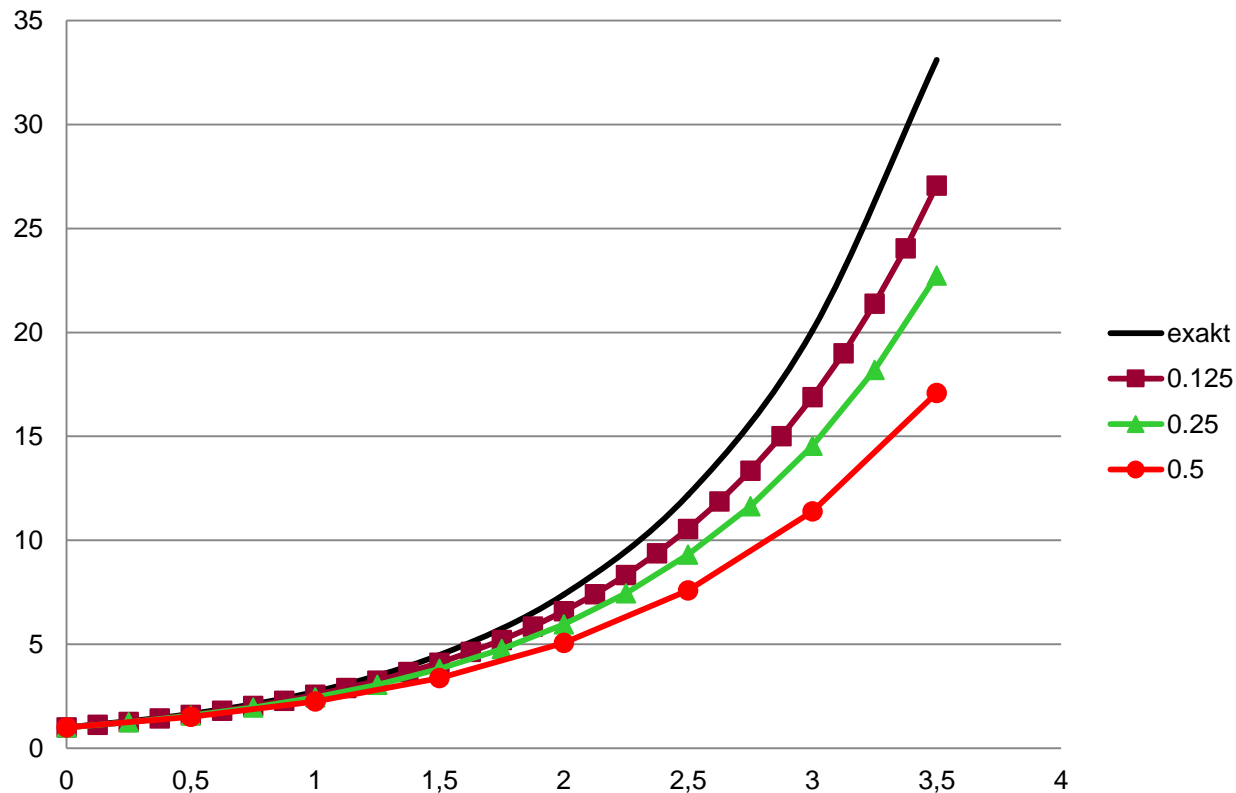
$$y_{k+1} = y_k + h \cdot f(y_k, t_k)$$



Example

IVP: $dy/dt = y$, $y(0) = 1$

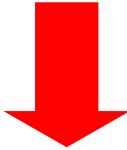
Solution $y = \exp(t)$ and Euler simulation with different h :




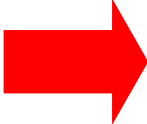
Euler's Method

Euler's method follows directly from the Taylor series:

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2!} y''(t) + \frac{h^3}{3!} y'''(t) + O(h^4)$$


$$y(t+h) = y(t) + hy'(t) + O(h^2)$$

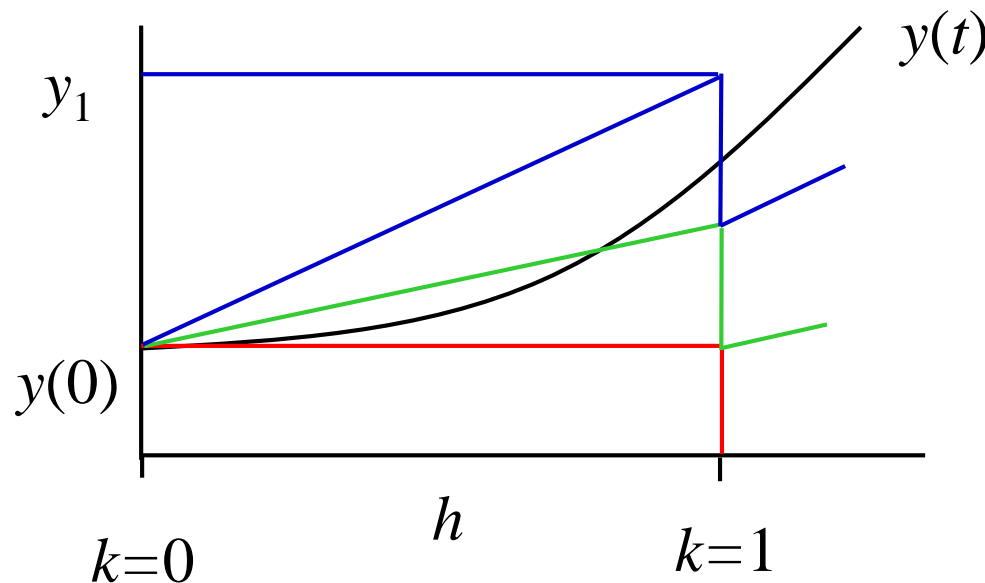

$$y(t+h) \approx y(t) + hy'(t)$$


$$y_{k+1} = y_k + h \cdot f(y_k, t_k)$$

Euler's Implicit Method

The implicit (backward) Euler method:

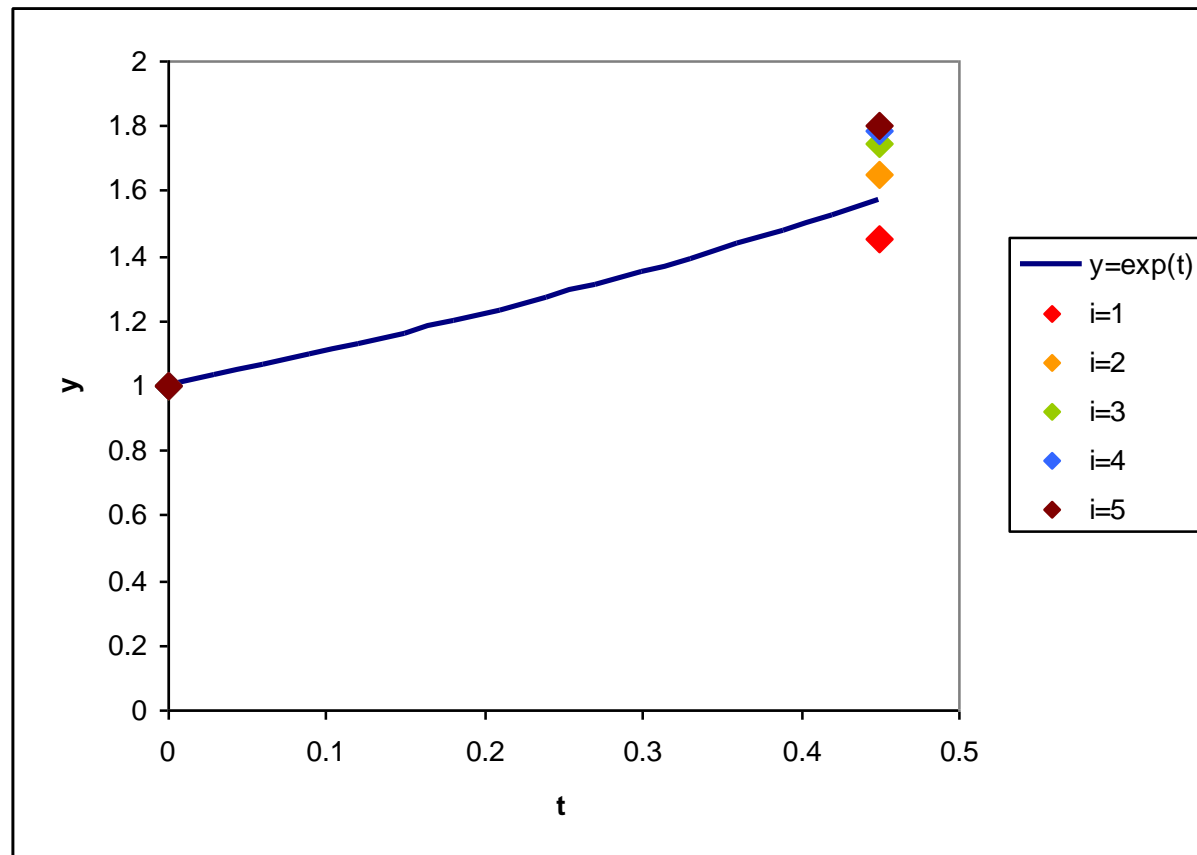
$$y_{k+1} = y_k + h \cdot f(y_{k+1}, t_{k+1})$$



An iteration is needed for each y_k – it is much more expensive!

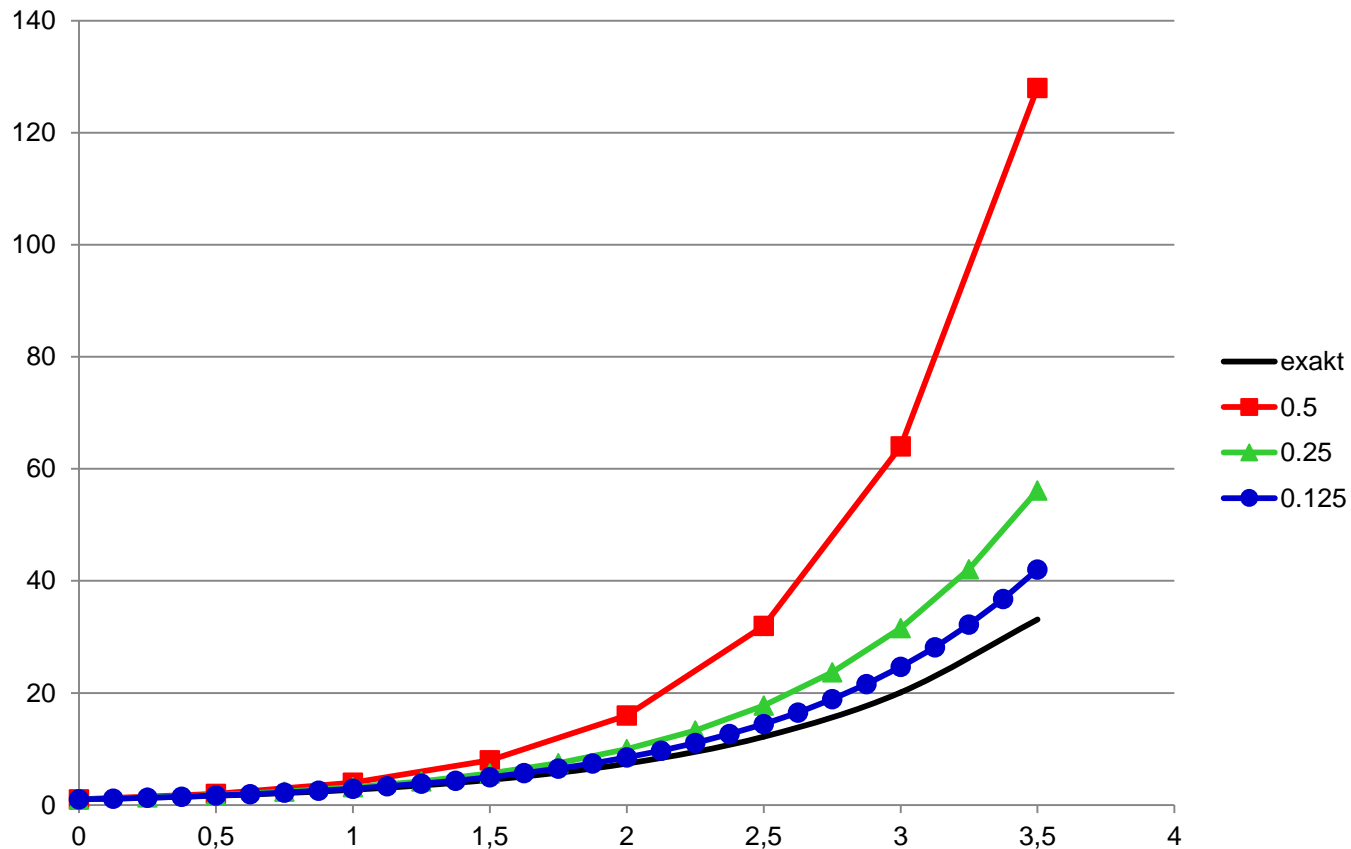
Euler's Implicit Method: Example

$dy/dt = y$ and first five Backward Euler iterates for $h=0.5$:



Euler's Implicit Method: Example

Same situation, backward Euler simulation:

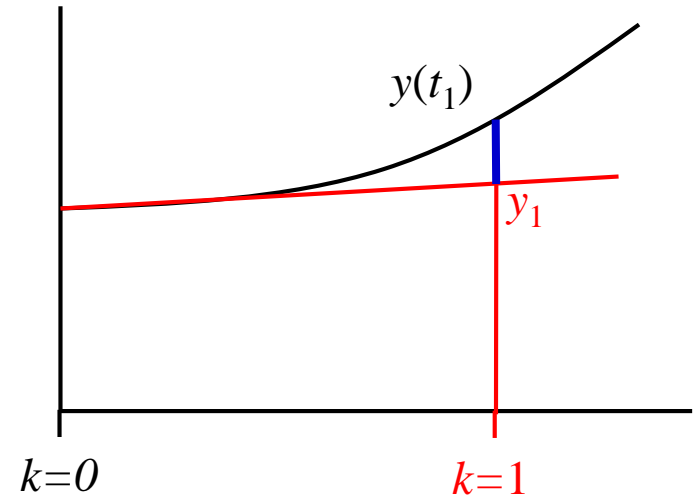


Errors

At time t_k we have ...

- the exact solution $y(t_k)$
- the approximation y_k

The global error is $y(t_k) - y_k$



Assuming there is no error in y_{k-1} (!) then

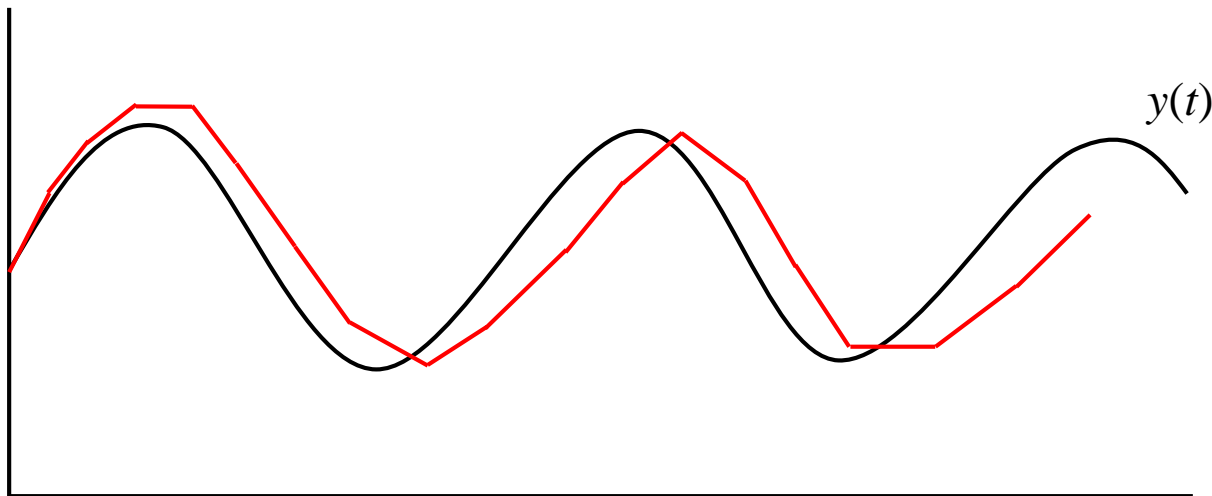
- ... the local truncation error (LTE) is $y(t_k) - y_k$

The LTE is the error made by the method at each step.

Error Growth

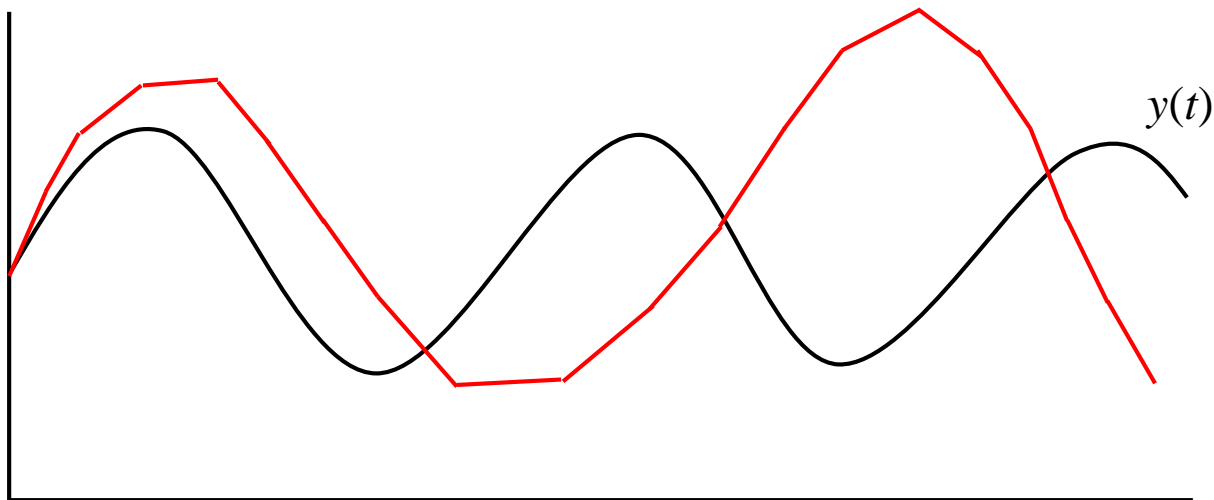
We are interested in the growth of the (global) error.

If h is chosen appropriately, the error will grow slowly:



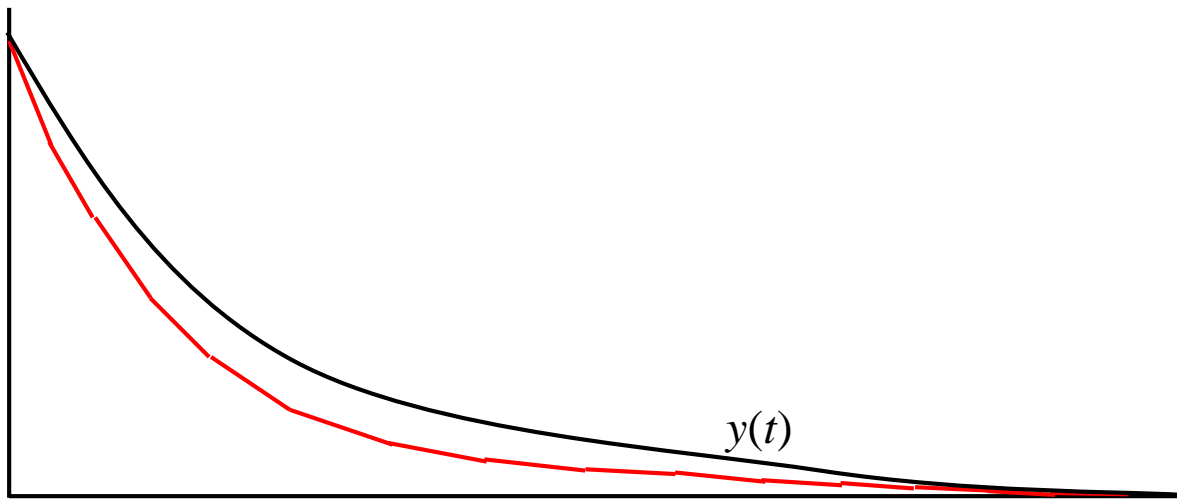
Error Growth

If h is too large, the error grows faster than is desirable:



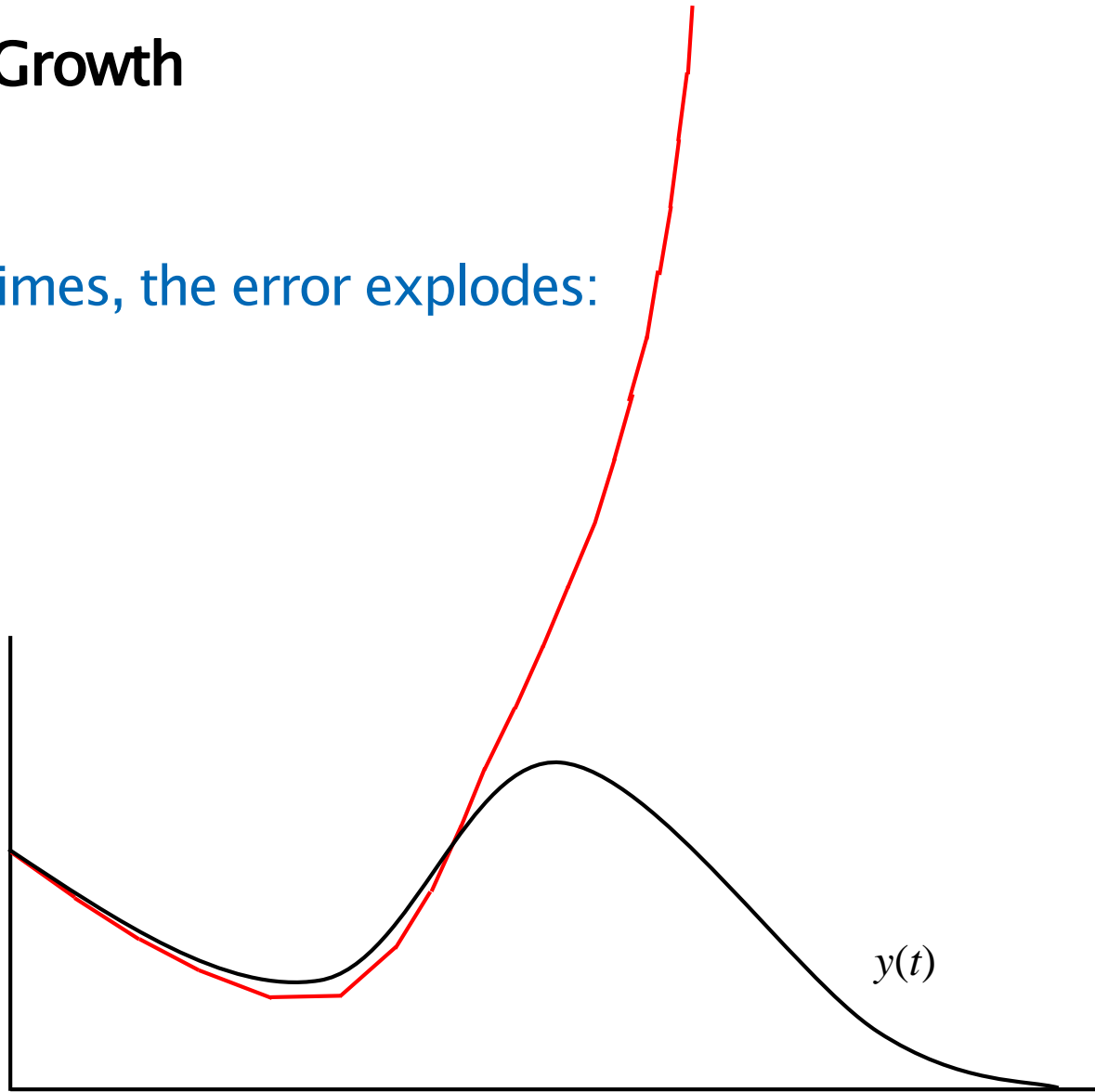
Error Growth

Sometimes, the error tends to zero:



Error Growth

Sometimes, the error explodes:



Order

Euler's method has a LTE of size $O(h^2)$:

$$y(t + h) = y(t) + hy'(t) + O(h^2)$$

It is a first-order method

- It can integrate (only) first-order polynomials exactly

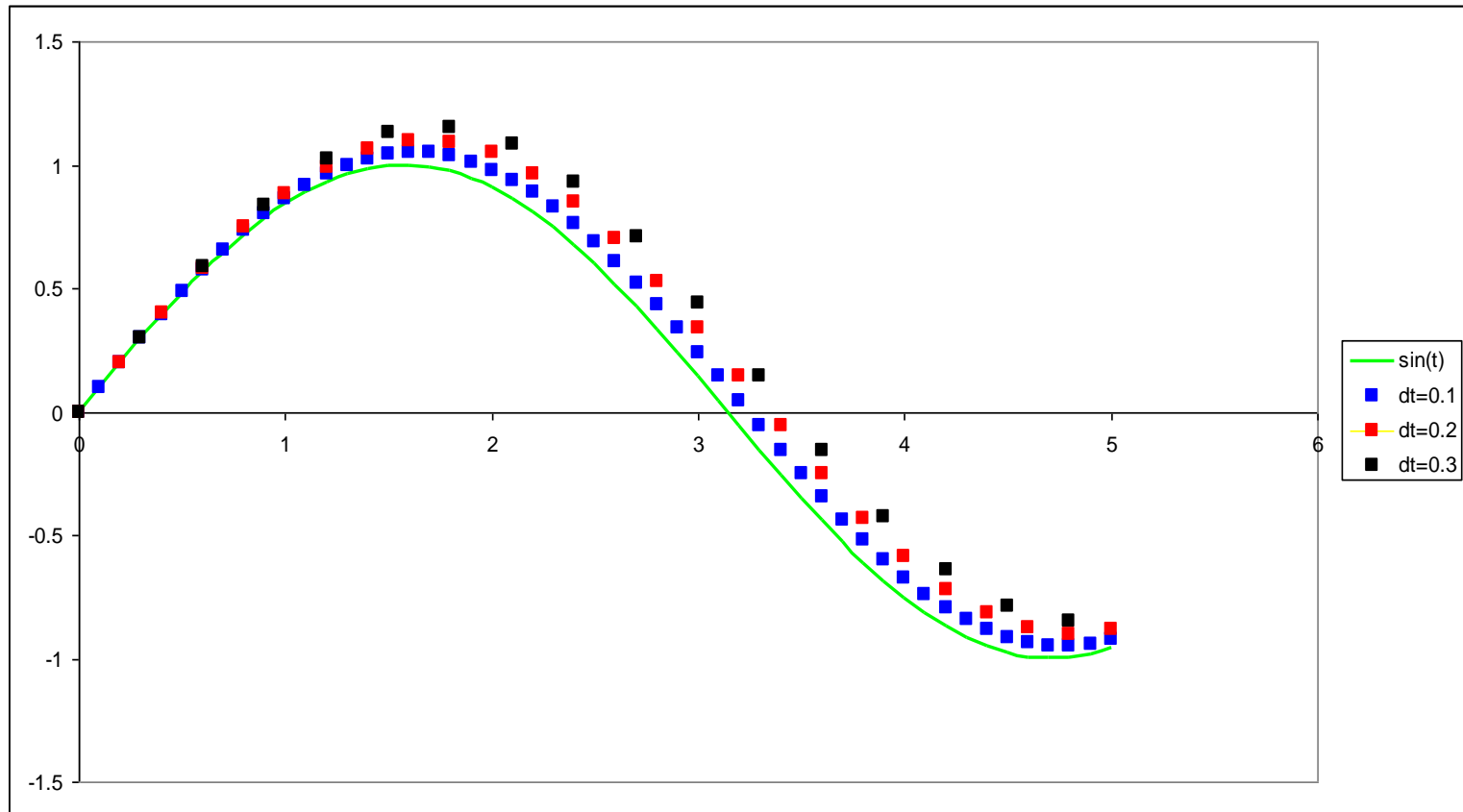
The global error of Euler's method is $O(h)$.

- Reason: GE is the sum of $O(1/h)$ LTEs each of size $O(h^2)$
- For small h : $h \rightarrow h/2 \Rightarrow \text{GE} \rightarrow \text{GE}/2$

Example

Graph of $y = \sin(t)$ and simulation results for $y' = \cos(t)$

- Euler's method with $h=0.1$, $h=0.2$ and $h=0.3$



Stability

Stability describes the development of the global error.

It is a property of the integration method.

- How does the method propagate errors as t increases?

If the error explodes for a given method and problem, then

- ... the method is unstable for that problem.

Implicit methods are always stable.

For explicit methods, stability requires that $h < h_{stable}$.

- This can be a severe limitation!

Stability

Example ODE:

$$u' = 98u + 198v$$

$$v' = -99u - 199v$$

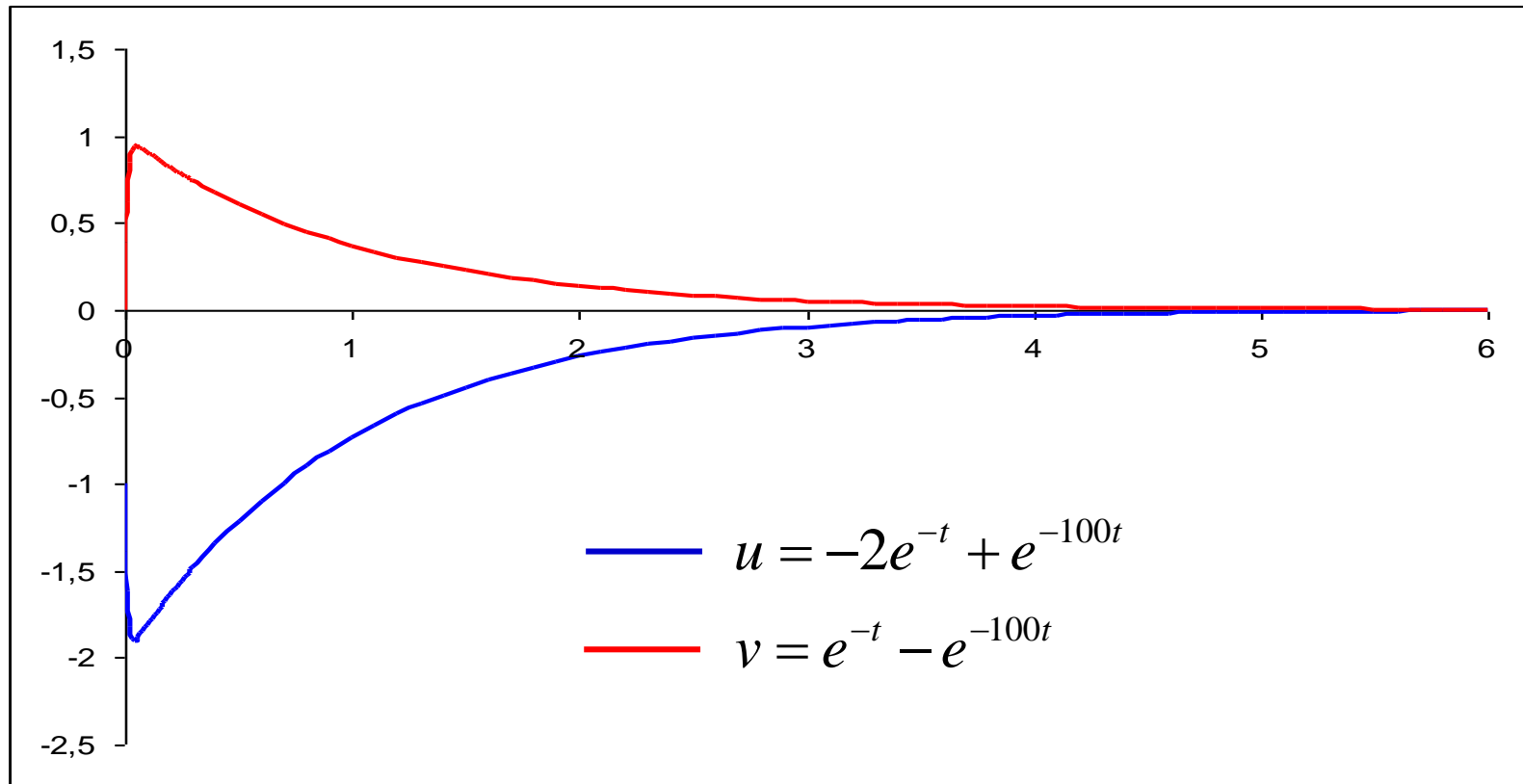
Analytic solution:

$$u = -2e^{-t} + e^{-100t}$$

$$v = e^{-t} - e^{-100t}$$

Stability

Graph of analytic solution:



Stability

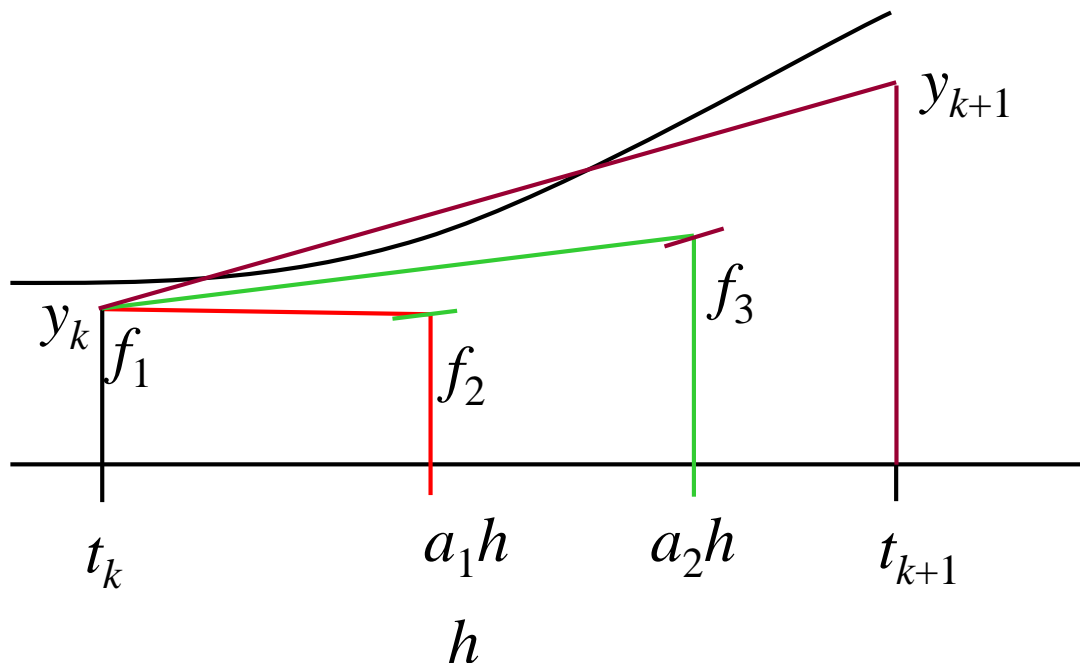
Examine simulation results (Euler's method) at $t = 6$:

| h (exact) | u | v |
|----------------|----------|---------|
| | -0.00496 | 0.00248 |
| 0.005 | -0.00488 | 0.00244 |
| 0.010 | -0.00481 | 0.00241 |
| 0.015 | -0.00474 | 0.00237 |
| 0.017 | -0.00471 | 0.00235 |
| 0.019 | -0.00468 | 0.00234 |
| 0.021 | 3.13E11 | 3.13E11 |

Runge–Kutta Methods

The most widely-used type of integration method.

- Idea: Obtain higher order using intermediate points.



Runge–Kutta Methods

The general case:

$$y_{k+1} = y_k + h \sum c_r f_r$$

$$f_1 = f(y_k, t_k)$$

$$f_r = f(y_k + h \sum b_{rs} f_s, t_k + a_r h)$$

$$a_r = \sum b_{rs}$$

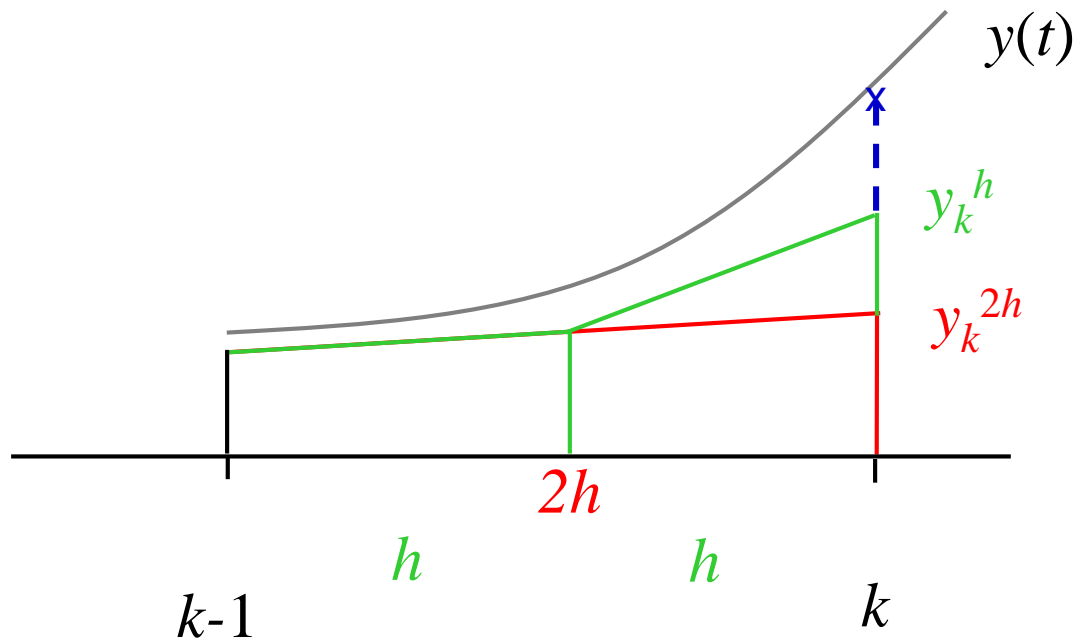
The most popular RK method is four–step, fourth order.

- For small h : $h \rightarrow h/2 \Rightarrow \text{GE} \rightarrow \text{GE}/16$

Extrapolation

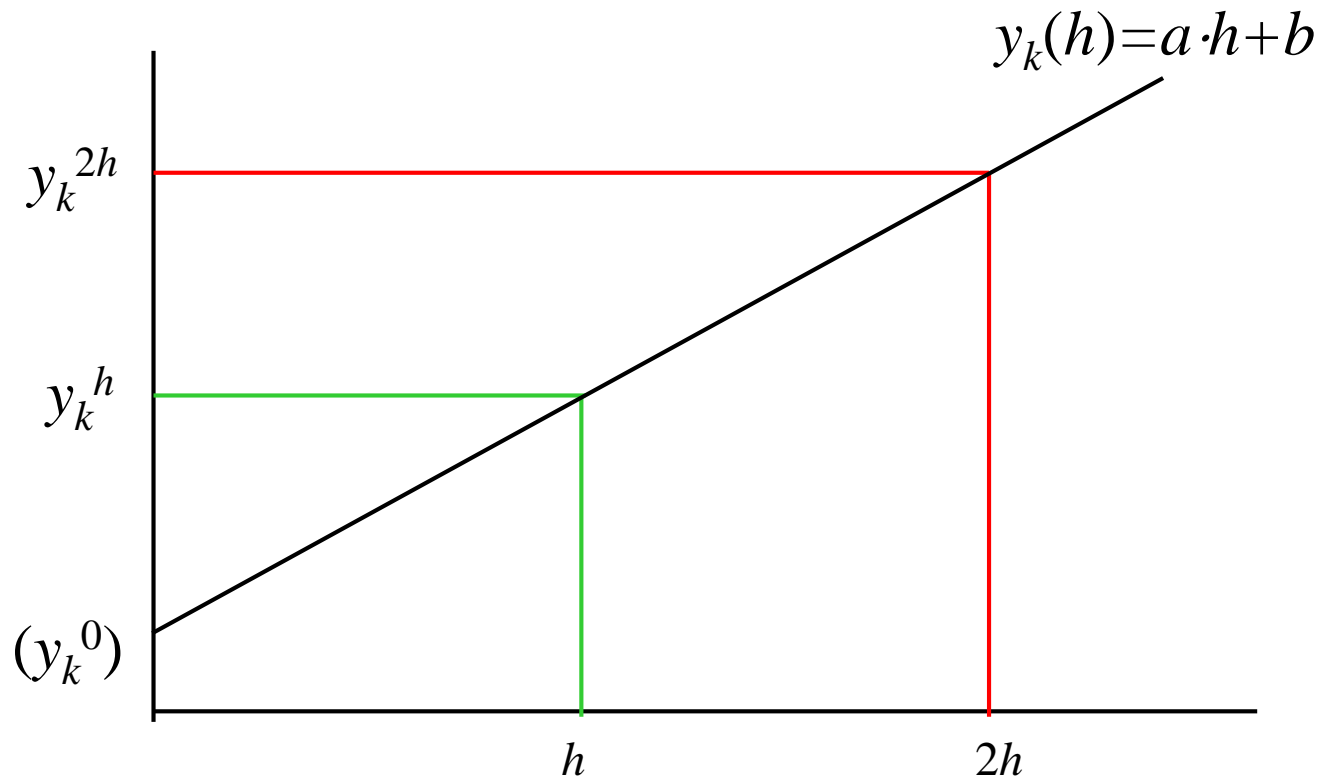
Extrapolation can increase accuracy cheaply.

- Idea: Combine two integration results of different accuracy.
- Approach: use two (or more) different step sizes.
- Goals: To estimate LTE or to improve accuracy.



Extrapolation

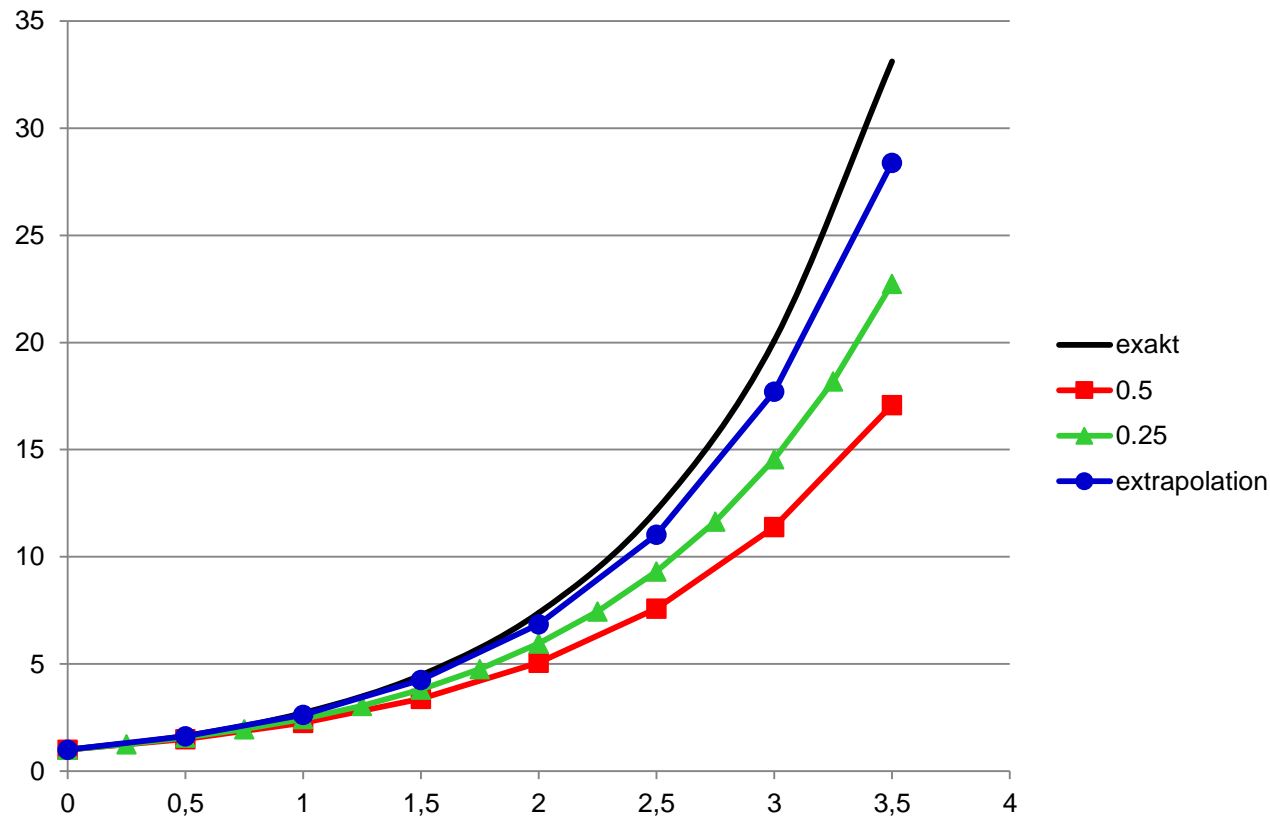
Why is it called extrapolation?



Extrapolation

IVP: $dy/dt = y$, $y(0) = 1$

Solution $y = \exp(t)$ and Euler simulation:



Step Size Control

Step size control is used to optimise computer time.

Ideas:

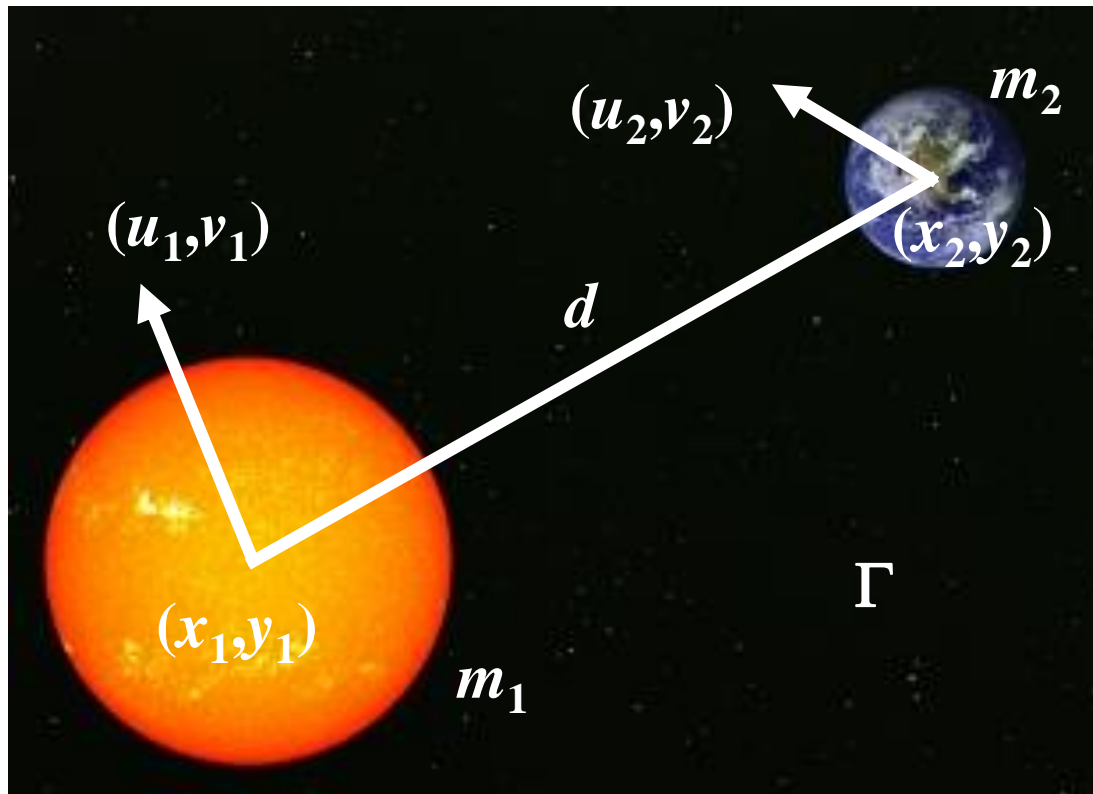
- User prescribes desired range for LTE: $E_{min} < LTE < E_{max}$
- Automatically decrease step size when $LTE > E_{max}$
- Automatically increase step size when $LTE < E_{min}$

Requires estimator for LTE (extrapolation, for example).

All advanced integration methods use step size control.

Simulation of Planet Orbit

The basic variables (2-D):



Mathematical Model

Mathematical model:

Sun:

$$\frac{du_1}{dt} = \frac{x_2 - x_1}{d} \frac{f}{m_1}$$

$$\frac{dv_1}{dt} = \frac{y_2 - y_1}{d} \frac{f}{m_1}$$

$$\frac{dx_1}{dt} = u_1$$

$$\frac{dy_1}{dt} = v_1$$

Earth:

$$\frac{du_2}{dt} = \frac{x_1 - x_2}{d} \frac{f}{m_2}$$

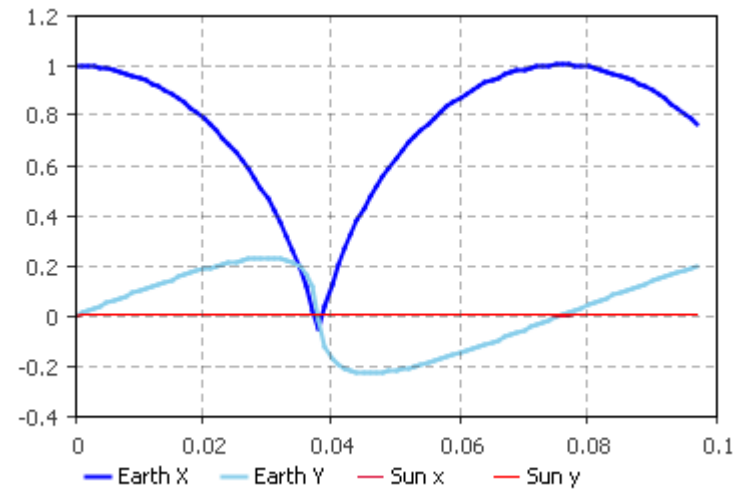
$$\frac{dv_2}{dt} = \frac{y_1 - y_2}{d} \frac{f}{m_2}$$

$$\frac{dx_2}{dt} = u_2$$

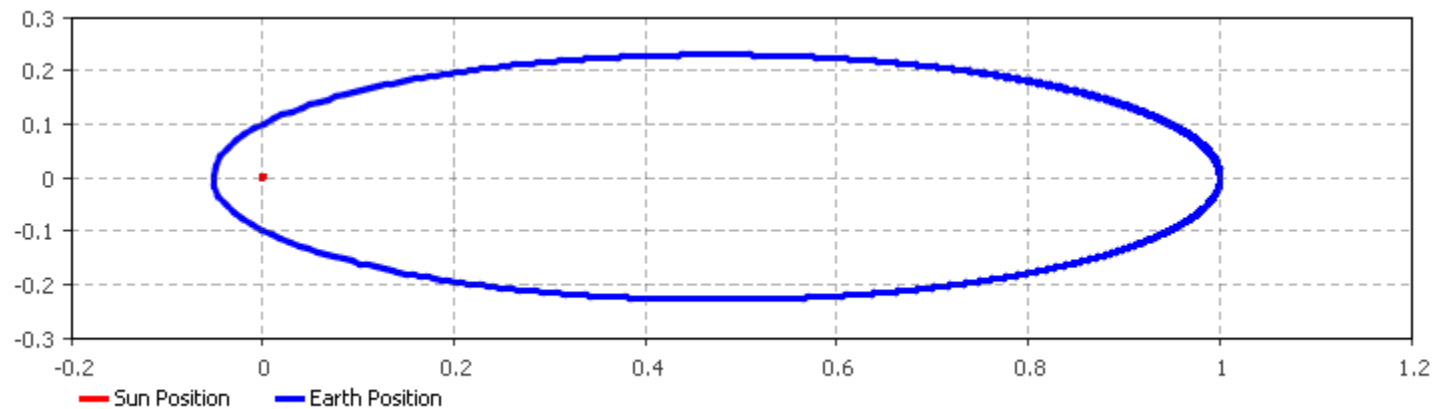
$$\frac{dy_2}{dt} = v_2$$

Simulation Result

In the time domain:



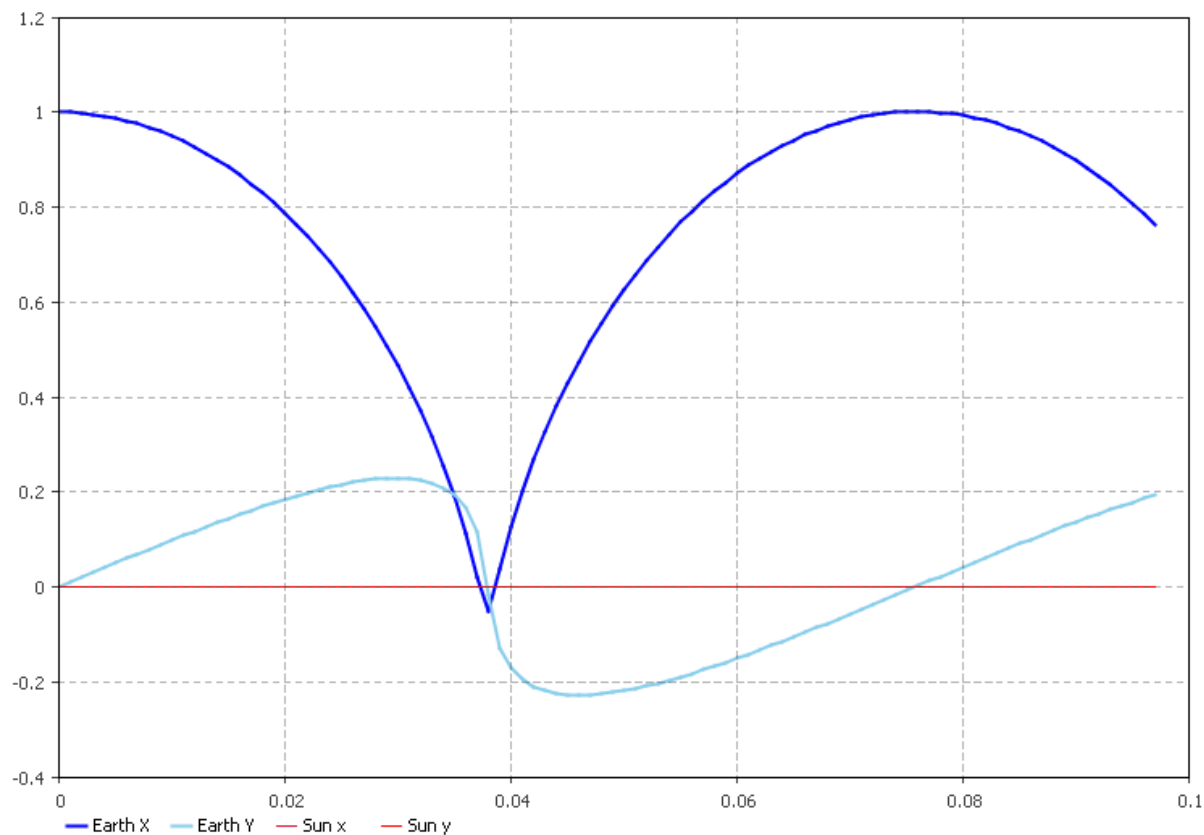
In state / phase space:



Simulation Result

A close-up look

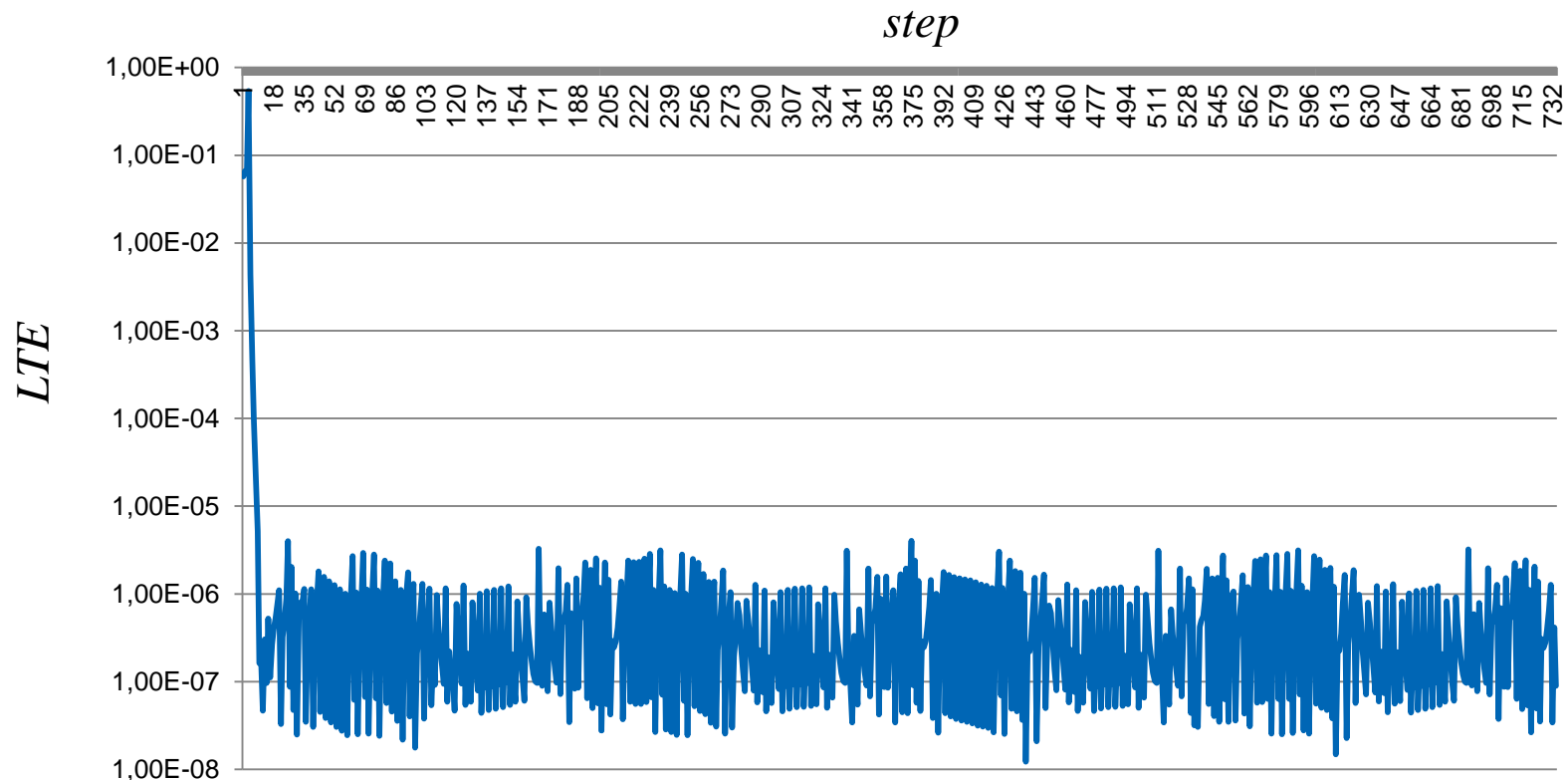
- Extreme acceleration in x-direction for the Earth



Local Truncation Error

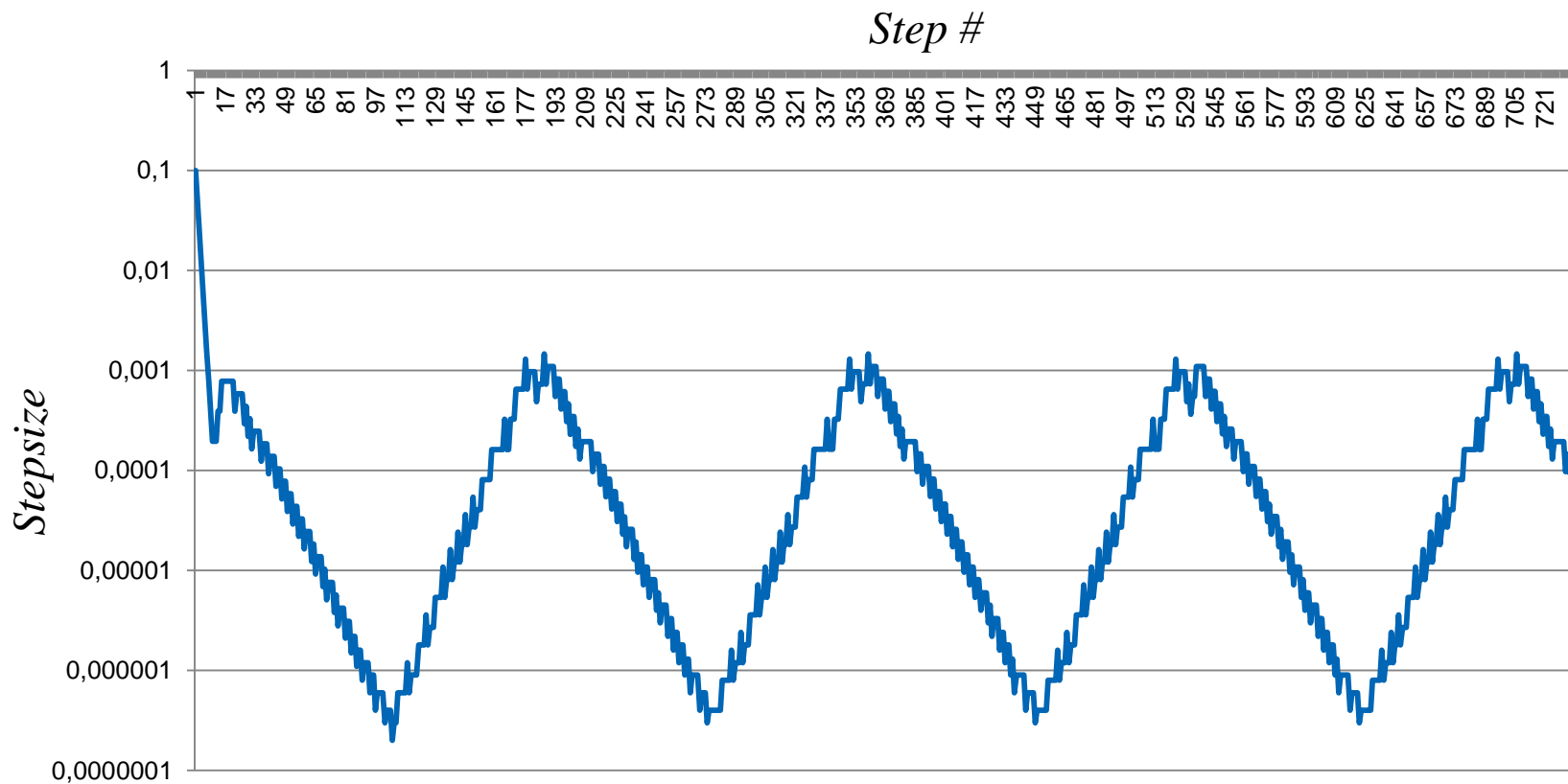
We tell our simulator to use stepsize control:

- Minimum LTE = $1\text{E-}7$; Maximum LTE = $1\text{E-}6$

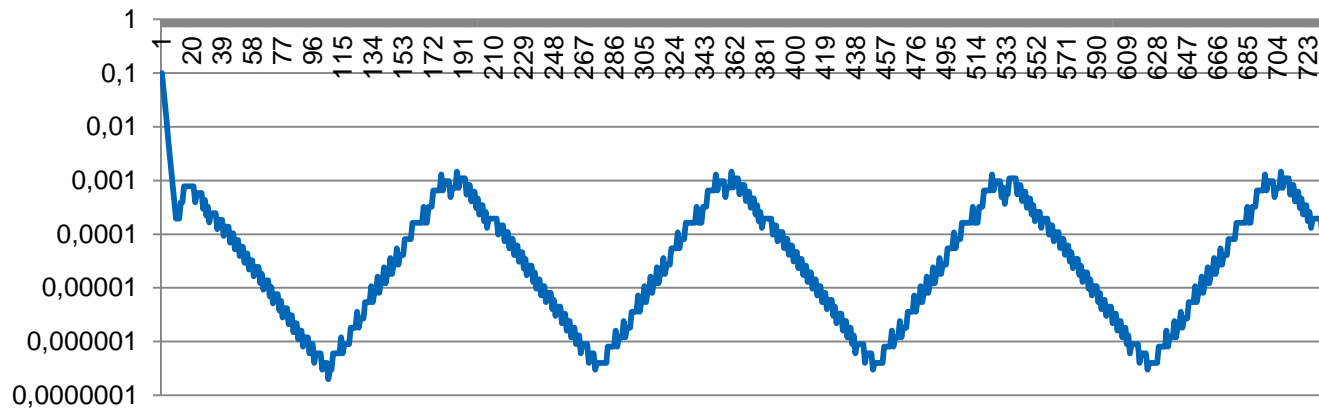
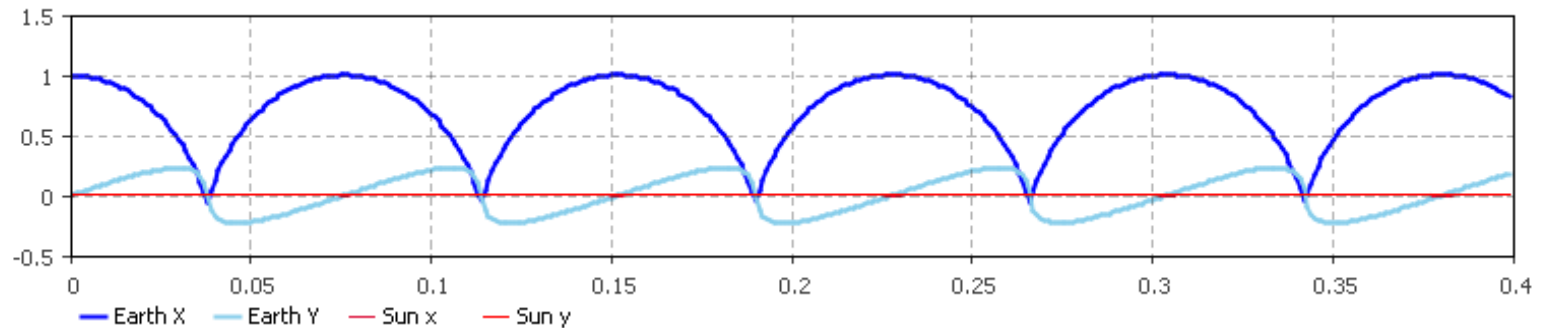


Stepsize

The stepsize varies by almost four orders of magnitude:



Solution v. Stepsize



Stiffness

Stiffness is a property of systems of ODEs.

Stiffness can force the integration step size to be very small.

The integration can therefore become very expensive.

Many problems in Science and Engineering are very stiff.

Stiffness

Consider a (linear) system of ODEs:

$$y' = Ay$$

Sort the eigenvalues λ_i of A :

$$|\operatorname{Re} \lambda_1| \geq |\operatorname{Re} \lambda_2| \geq \dots \geq |\operatorname{Re} \lambda_n|$$

Compute the stiffness ratio S :

$$S = \frac{|\operatorname{Re} \lambda_1|}{|\operatorname{Re} \lambda_n|}$$

Stiffness

If the stiffness ratio S satisfies:

$$S = \frac{|\operatorname{Re} \lambda_1|}{|\operatorname{Re} \lambda_n|} \gg 1$$

- ... then the system $y' = Ay$ is said to be *stiff*

Typical engineering problems can have $S > 10^6$ or more.

Stiffness

Example:

$$u' = 98u + 198v$$

$$v' = -99u - 199v$$

Analytic solution:

$$u = -2e^{-t} + e^{-100t}$$

$$v = e^{-t} - e^{-100t}$$

The eigenvalues of $\begin{pmatrix} 98 & 198 \\ -99 & -199 \end{pmatrix}$ are $\lambda_1 = -100$ and $\lambda_2 = -1$.

The Stiffness ratio is therefore $S = 100$.

Stiffness

Recall that Euler's method is unstable for $h = 0.021$:

| h (exact) | u | v |
|----------------|----------|---------|
| | -0.00496 | 0.00248 |
| 0.005 | -0.00488 | 0.00244 |
| 0.010 | -0.00481 | 0.00241 |
| 0.015 | -0.00474 | 0.00237 |
| 0.017 | -0.00471 | 0.00235 |
| 0.019 | -0.00468 | 0.00234 |
| 0.021 | 3.13E11 | 3.13E11 |

Stiffness

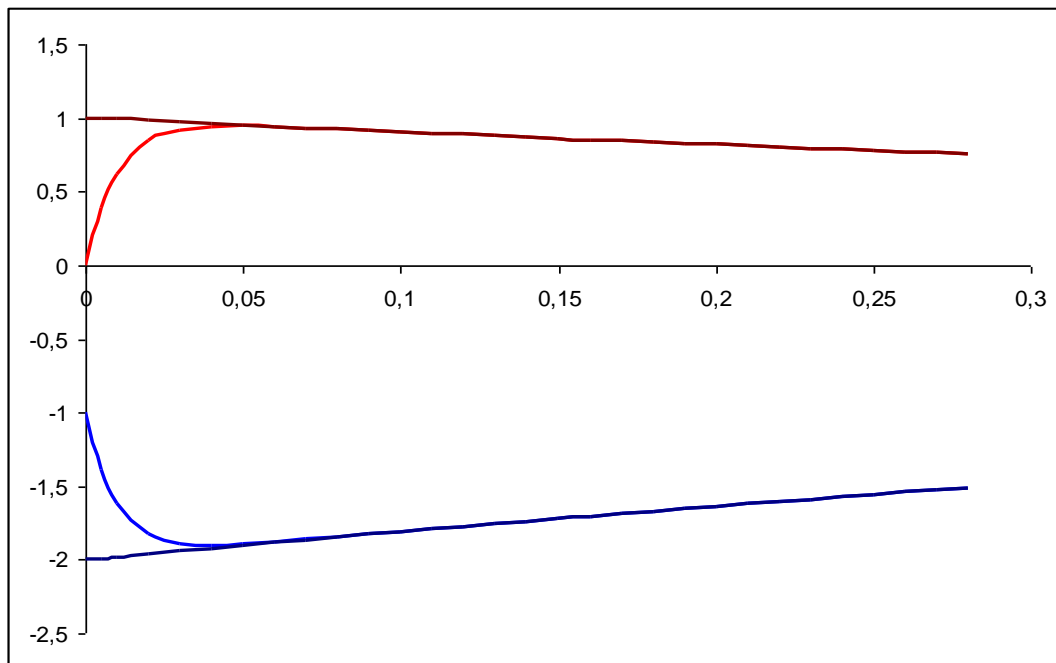
The contribution of the e^{-100t} terms in $u = -2e^{-t} + e^{-100t}$
becomes negligible very quickly.

$$v = e^{-t} - e^{-100t}$$

For $t > 0.1$, the equations are practically identical to:

$$u = -2e^{-t}$$

$$v = e^{-t}$$



Stiffness

The new equations have $S = 2$ (!)

They can be integrated quite accurately with a larger h .

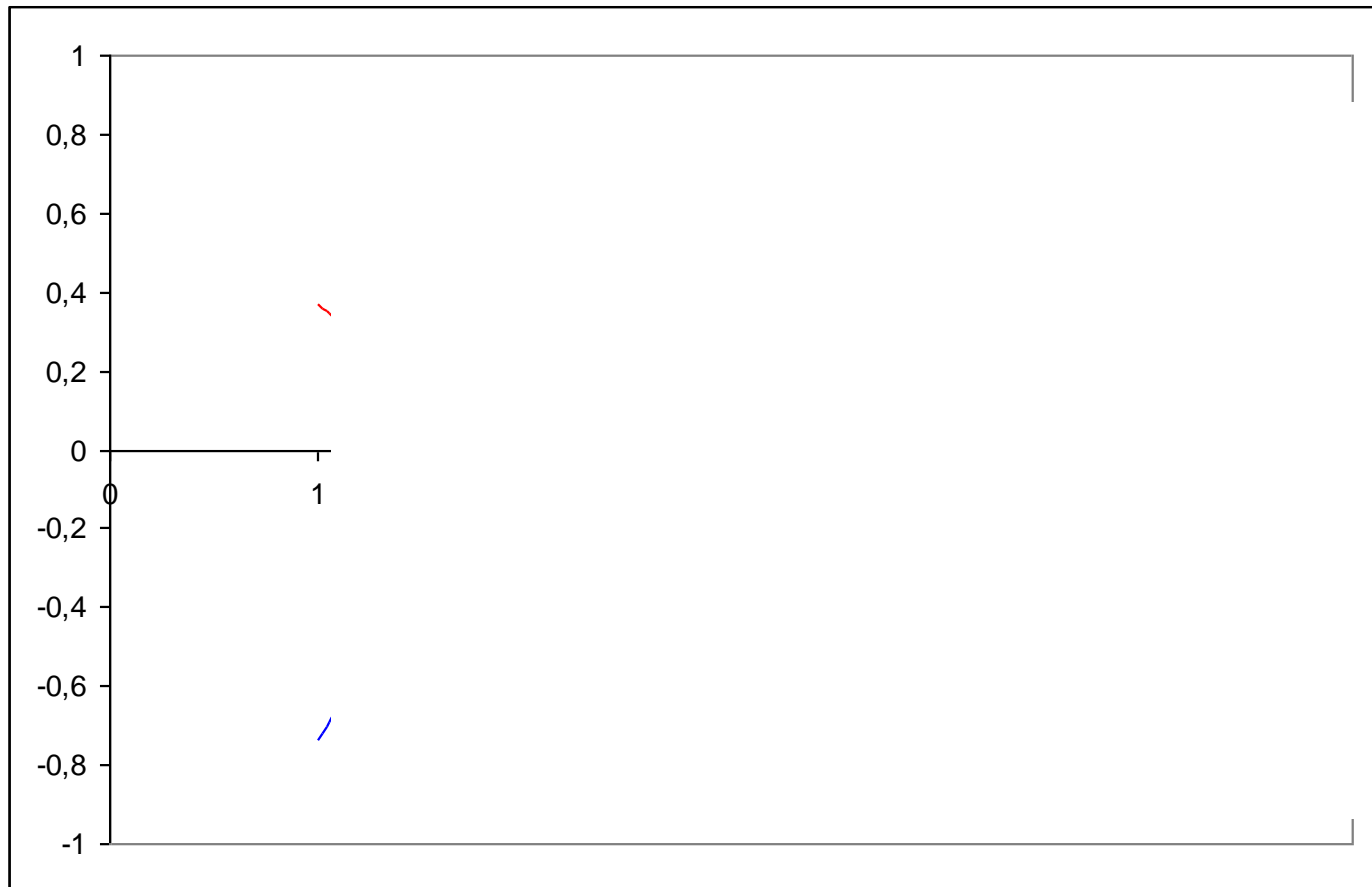
It seems that only the part $t < 0.1$ is critical to stability.

It seems therefore that we could safely do the following:

- Integrate up to $t = 0.1$ using a small step size.
- Integrate from $t > 0.1$ using a larger step size.

Stiffness

Experiment: Integrate from $t = 1$ using $h = 0.021$:



Stiffness

Even when the terms causing stiffness are negligible in the solution, they still force us to use a small step size!

A practical definition of stiffness:

- A system is stiff, when stability demands a small step size, even when accuracy doesn't.

Practical consequences of stiffness:

- We must either use implicit methods or very small step sizes!
- The computation will be expensive!

Learning Questions

Learning Goals:

- How do the implicit and explicit Euler methods work?
- What is stiffness? How does it affect integration methods?
- What is meant by the order of an integration method?
- What is Local Truncation Error and what is it used for?
- How does step size control work?
- How does the extrapolation method increase accuracy?
- What is stability?