Differential equations

• A **differential equation** (**DE**) describes how a quantity changes (as a function of time, position, ...).

- A ball dropped from a building: $$\frac{d}{dt}h(t) = -gt \qquad \textbf{(1)}$$
- Uniformly loaded beam: $$\frac{d^2}{dx^2}w(x) = \frac{S}{EI}w(x) + \frac{qx}{2EI}(x-L) \qquad \textbf{(2)}$$
- Schrödinger equation: $$-\frac{(h/2\pi)^2}{2m}\nabla^2\psi(\mathbf{r}) + V(\mathbf{r})\psi(\mathbf{r}) = E\psi(\mathbf{r}) \qquad \textbf{(3)}$$
- Heat conduction: $$\rho c_P \frac{\partial}{\partial t}T(\mathbf{r}, t) = \nabla \cdot (k\nabla T(\mathbf{r}, t)) \qquad \textbf{(4)}$$

. . .

- DEs can be classified in various ways:

Ordinary DEs (**ODEs**)
Only one independent variable
Eqs (1) and (2)

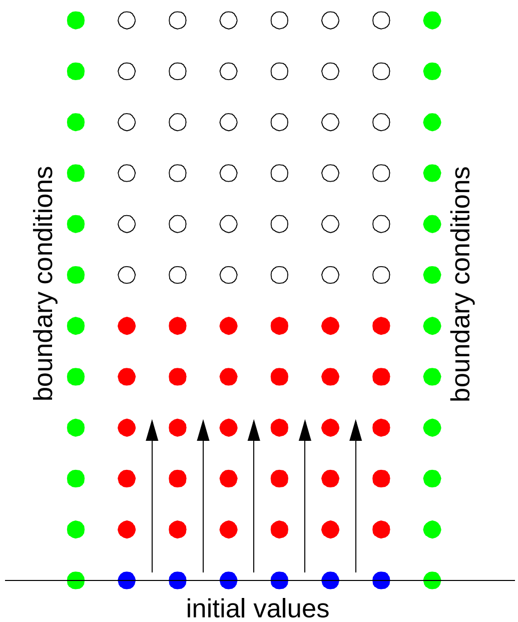
Partial DEs (**PDEs**)
Many independent variables
Eqs (3) and (4)

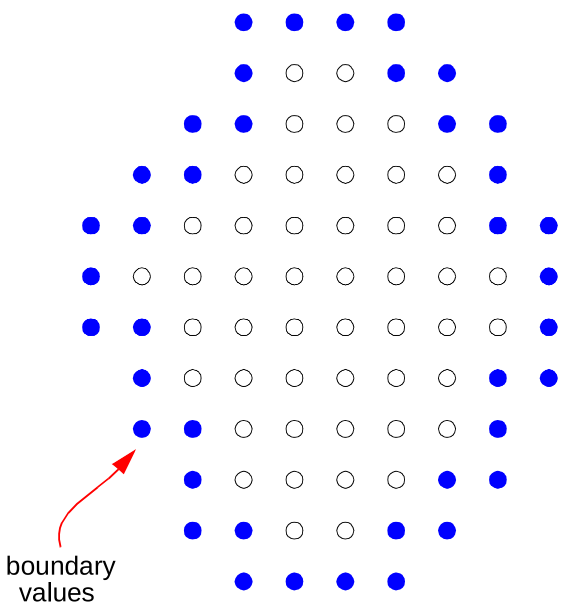
Initial value problems (**IVP**)
Function value(s) known at certain values of the variable(s).
Want function value(s) at arbitrary variable value(s).
Eqs (1) and (4)
In 1D: $f(t_0) = y_0$, compute $f(t)$.

Boundary value problems (**BVP**)
Function value(s) known on the domain borders.
Want function values in other point(s) in the domain.
Eqs (2), (3) and (4)
In 1D: $f(x_1) = y_1$, $f(x_2) = y_2$, compute $f(x)$.
(*two-point BVP*)

Differential equations

• The difference between initial value problems and boundary value problems can be illustrated as below (according to NR, fig. 19.0.1).





- In general boundary value problems are harder to solve than initial value problems:
  BVPs need iteration IVP you ‘only’ need to integrate the equations in ‘time’ to reach the desired point.

- In this chapter we deal with ordinary differential equations both IVPs and BVPs.
- Probably there will be no time partial differential equations.

Differential equations: initial value problems

- The simplest ODE (IVP) is

$$\frac{dy}{dt} = f(t)$$

and its solution (starting from 'time' t_0)

$$y(t) = c + \int_{t_0}^{t} f(\tau)d\tau .$$ (*Integration of the differential equation.*)

- The solution $y(t)$ is unique up to the constant c i.e. we get a family of solutions.

- The initial condition is needed to fix the value of the constant and to get a unique solution:

$$y(t_0) = y_0 .$$

- The general form of an IVP is

$$\frac{dy}{dt} = f(t, y) .$$

Differential equations: initial value problems

- Solution methods for a single ODE are easily generalized to a group if ODEs (Use of vector notation makes it easier.)

$$\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \dots \\ y_n(t) \end{bmatrix}, \quad \mathbf{f}(t, \mathbf{y}) = \begin{bmatrix} f_1(t, \mathbf{y}) \\ f_2(t, \mathbf{y}) \\ \dots \\ f_n(t, \mathbf{y}) \end{bmatrix}, \quad \mathbf{y}_0 = \begin{bmatrix} A_1 \\ A_2 \\ \dots \\ A_n \end{bmatrix}$$

$$y' \equiv \frac{dy}{dt}$$

$$\mathbf{y}' = \mathbf{f}(t, \mathbf{y}), \ \mathbf{y}(t_0) = \mathbf{y}_0 .$$

- ODEs having also higher derivatives can be transformed to groups of ODEs so that methods presented here can be applied to these higher order ODEs:

$$\begin{cases} y^{(n)} = f(t, y, y', \dots, y^{(n-1)}) \\ y(t_0) = A_1 \\ y'(t_0) = A_2 \\ \dots \\ y^{(n-1)}(t_0) = A_n \end{cases} \rightarrow \begin{cases} y_1' = y_2 \\ y_2' = y_3 \\ \dots \\ y_n' = f(t, y_1, y_2, \dots, y_n) \\ y_1(t_0) = A_1 \\ y_2(t_0) = A_2 \\ \dots \\ y_n(t_0) = A_n \end{cases}$$

Differential equations: initial value problems

- Numerical solution calls for discretization of the problem.
  - Instead of computing $y(t)$ at any t we obtain values t_i in discrete points.
    (We talk about time here, but t can - of course - be any variable to be solved.)
  - The interval between points (in IVPs the time step) may vary as a function of time (variable time step).

- There several error sources in the integration of ODEs:
  - Discretization error: a too large time step $\rightarrow$ decrease the time step.
  - Instability: small errors introduced in the beginning grow exponentially $\rightarrow$ change the algorithm.
  - Stiff ODEs: phenomena with different time scales $\rightarrow$ step size selected according to the fastest processes

Differential equations: initial value problems

- Something general can be said about the existence of the solution of the IVP $y' = f(t, y)$
  - A function $f(t, y)$ is said to satisfy a **Lipschitz condition** in the variable y in the set $D \subset \mathbf{R}^2$ if $\exists L > 0$ so that $|f(t, y_1) - f(t, y_2)| \le L|y_1 - y_2|$ whenever $(t, y_1), (t, y_2) \in D$.
    - L is called the Lipschitz constant of f.
  - A set $D \subset \mathbf{R}^2$ is said to be **convex** if whenever $(t_1, y_1), (t_2, y_2) \in D$ the point $((1-\lambda)t_1 + \lambda t_2, (1-\lambda)y_1 + \lambda y_2)$ also belongs to D for each λ, $0 \le \lambda \le 1$.
  - Suppose $f(t, y)$ is defined in a convex set $D \subset \mathbf{R}^2$. If $\exists L > 0$, such that $\left|\frac{\partial}{\partial y} f(t, y)\right| \le L$, for all $(t, y) \in D$ then f satisfies the Lipschitz condition on D in variable y with Lipschitz constant L.
  - Suppose that $D = \{(t, y) | a \le t \le b, -\infty < y < \infty\}$ and that $f(t, y)$ is continuous on D.
    If f satisfies the Lipschitz condition on D in variable y then the IVP
    $$y' = f(t, y), \quad a \le t \le b, \; y(a) = \alpha$$
    has a unique solution $y(t)$ for $a \le t \le b$.

Differential equations: initial value problems

- Example: $y' = 1 + t\sin(ty)$, $0 \le t \le 2$. $y(0) = 0$.
  - Keeping t constant we apply the mean value theorem to function $f(y) = 1 + t\sin(ty)$:

$$\frac{df(\xi)}{dy} = \frac{f(y_2) - f(y_1)}{y_2 - y_1}, \quad y_1 < y_2, y_1 < \xi < y_2.$$

$$\rightarrow \quad t^2\cos(\xi t) = \frac{f(y_2) - f(y_1)}{y_2 - y_1} \quad \rightarrow \quad 4 \ge \frac{|f(y_2) - f(y_1)|}{|y_2 - y_1|} \quad \rightarrow \quad |f(y_2) - f(y_1)| \le 4|y_2 - y_1| .$$

  - Thus $L = 4$ and the IVP has a unique solution.

- What about the stability of the IVP?

- The IVP $y' = f(t, y)$, $a \le t \le b$, $y(a) = \alpha$ is said to be a **well-posed problem** (or a **stable problem**) if
  1. A unique solution exists.
  2. For any $\varepsilon > 0$, $\exists k > 0$ with the property that, whenever $|\varepsilon_0| < \varepsilon$ and $|\delta(t)| < \varepsilon$,

     a unique solution $z(t)$ to the problem

     $z' = f(t, z) + \delta(t)$, $a \le t \le b$, $z(a) = \alpha + \varepsilon_0$ **(PB)** (=perturbed problem)

     exists with

     $|z(t) - y(t)| < k\varepsilon$, $\forall t, a \le t \le b$.

- One can show that the IVP $y' = f(t, y)$ is a well-posed problem if
  1) $f(t, y)$ is a continuous function of t and y for all $(t, y) \in D$ and
  2) it satisfies the Lipschitz condition $|f(t, y_1) - f(t, y_2)| \le L|y_1 - y_2|$, $(t, y_1), (t, y_2) \in D$.

Differential equations: initial value problems

- Example:

$$D = \{(t, y) | 0 \le t \le 1, -\infty < y < \infty\}$$
$$y' = -y + t + 1$$
$$y(0) = 1.$$

  - Since $\left|\frac{\partial(-y + t + 1)}{\partial y}\right| = 1$, the function $f(t, y) = -y + t + 1$ satisfies the Lipschitz condition.
  - Morever, f is continuous $\rightarrow$ the problem is well posed.

  - Let's see what we get when we perturb the problem:

$$\frac{dz}{dt} = -z + t + 1 + \delta, \quad 0 \le t \le 1, \quad z(0) = 1 + \varepsilon_0.$$

  - Solutions to the original and perturbed problems are

$$y(t) = e^{-t} + t, \quad z(t) = (1 + \varepsilon_0 - \delta)e^{-t} + t + \delta.$$

  - One can now easily see that if $|\delta| < \varepsilon$ and $|\varepsilon_0| < \varepsilon$ then

$$|y(t) - z(t)| = |(\delta - \varepsilon_0)e^{-t} - \delta| \le |\varepsilon_0| + |\delta||1 - e^{-t}| \le 2\varepsilon \;\; \forall t.$$

Differential equations: initial value problems

- The above discussion concerns the stability of the problem itself.
  - A problem is stable (well-posed) if the solution depends continuously on the initial value and on the right-hand side function f.

- The IVP could, however, be stable but **ill-conditioned**.
  - Condition number gives the sensitivity of the solution to the initial data.
  - A large condition number means an ill-conditioned problem.

- Assume we have a perturbed IVP: $y'(t;\varepsilon) = f(t, y(t;\varepsilon))$, $y(t_0;\varepsilon) = y_0 + \varepsilon$, $t \in [a, b]$.
  - Let's write $z(t;\varepsilon) = y(t;\varepsilon) - y(t)$, where $y(t)$ is the solution of the unperturbed IVP. This gives us

$$z'(t;\varepsilon) = f(t, y(t;\varepsilon)) - f(t, y(t)) \approx \frac{\partial f(t, y(t))}{\partial y} z(t;\varepsilon) \quad \text{and} \quad z(t_0;\varepsilon) = \varepsilon .$$

  - The above approximation is valid for small ε (we have a well-posed problem).
  - From the equation we can readily solve z:

$$z(t;\varepsilon) \approx \varepsilon \exp\left[\int_{t_0}^{t} \frac{\partial f(t', y(t'))}{\partial y} dt'\right] .$$

  - This means that if

$$\frac{\partial f(t, y(t))}{\partial y} \le 0 , \quad t \in [a, b]$$

  then $z(t;\varepsilon)$ probably remains bounded by ε as t increases. In this case the problem is well-conditioned.

   - An example of an ill-conditioned problem is: $y' = \lambda y + g(t)$, $y(0) = y_0$, $\lambda > 0$.

   - Now we can calculate $z(t;\varepsilon) = \varepsilon e^{\lambda t}$ indicating that the change in $y(t)$ becomes increasing large as t increases.

Differential equations: initial value problems

- As another example of an ill-conditioned problem consider

$$y' = 100y - 101e^{-t}, \quad y(0) = 1 .$$

  - The solution is

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

  - Perturbed problem:

$$y' = 100y - 101e^{-t}, \quad y(0) = 1 + \varepsilon ,$$

  has the solution

$$y(t;\varepsilon) = e^{-t} + \varepsilon e^{100t} .$$

IVP: $y' = ay + be^{-t}$

$y(0) = y_0$

$f(t, y) = ay + be^{-t}$

$\frac{\partial f}{\partial y} = a$

Solution: $y(t) = \frac{1}{a+1}[(y_0 + ay_0 + b)e^{at} - be^{-t}]$

$$z(t;\varepsilon) \approx \varepsilon \exp\left[\int_0^t \frac{\partial f(t', y(t'))}{\partial y} dt'\right] = \varepsilon \exp\left[\int_0^t \frac{\partial (100y - 101e^{-t})}{\partial y} dt'\right] = \varepsilon e^{100t}$$

Differential equations: initial value problems

- Stability of the solution of an IVP $y'(t) = f(t, y)$:

  - Solution is **stable** if $\forall\varepsilon > 0\ \exists\delta > 0$, such that every solution of the problem $\hat{y}$ that fulfills

    $$|y(0) - \hat{y}(0)| \le \delta$$

    also fulfills

    $$|y(t) - \hat{y}(t)| \le \varepsilon$$

    for all $t \ge 0$.

  - Solution is **asymptotically stable** if it is stable and

    $$\lim_{t \to \infty} |y(t) - \hat{y}(t)| = 0 .$$

  - A simple example: $y' = \lambda y$. Solution is $y(t) = y(0)e^{\lambda t}$.
    - If $\lambda > 0$ all solutions grow and diverge from each other exponentially: unstable solution.
    - If $\lambda < 0$ all solution converge to zero: asymptotically stable solution.
    - If λ is complex ($\lambda = a + ib$) then stability depends on $\mathrm{Re}(\lambda)$: $y(t) = y(0)e^{(a + ib)t}$.
    - If $\mathrm{Re}(\lambda) = 0$ then we get an oscillatory solution that is stable but no asymptotically stable.

- In addition to the stability definitions given above, the numerical method used to solve the IVP can also be stable or unstable.

Differential equations: initial value problems

- **Euler's method** is the simplest possible method to solve an IVP.
- It is not used much in practical problems but its derivation illustrates the concepts needed in deriving more advanced methods.

- The IVP we are trying to solve is the same as before

$$\frac{dy}{dt} = f(t, y) , \quad a \le t \le b , \; y(a) = \alpha .$$

- The problem is discretized by computing the y values in mesh points $\{t_0, t_1, \ldots t_N\}$ in interval $[a, b]$:

$$t_i = a + ih , \quad i = 0, 1, \ldots, N .$$

  - The distance between the consecutive points $h = (b - a)/N$ is called the **step size**.

  - Assume the solution $y(t)$ has two continuous derivatives.
  - We will expand it as a Taylor series

$$y(t_{i+1}) = y(t_i) + (t_{i+1} - t_i)y'(t_i) + \frac{1}{2}(t_{i+1} - t_i)^2 y''(\xi_i) , \quad \xi_i \in [t_i, t_{i+1}]$$

or

$$y(t_{i+1}) = y(t_i) + hy'(t_i) + \frac{1}{2}h^2 y''(\xi_i) .$$

  - Since $y(t)$ is the solution to the IVP we can write the expansion as

$$y(t_{i+1}) = y(t_i) + hf(t_i, y(t_i)) + \frac{1}{2}h^2 y''(\xi_i) .$$

Differential equations: initial value problems

- In Euler's method we simply drop the 2nd order term and get the iteration formula

$$y_0 = \alpha,$$
$$y_{i+1} = y_i + hf(t_i, y_i), \quad i = 0, 1, \ldots, N.$$

- So we get an approximation to the solution

$$y_i \approx y(t_i).$$

- The algorithm is extremely simple

1. Set $h = (b-a)/N$, $t = a$, $.y = \alpha$.
2. For $i = 1, 2, \ldots, N$ do
   Set $y \leftarrow y + hf(t, y)$.
   Set $t \leftarrow a + ih$.

Differential equations: initial value problems

- As a simple example take

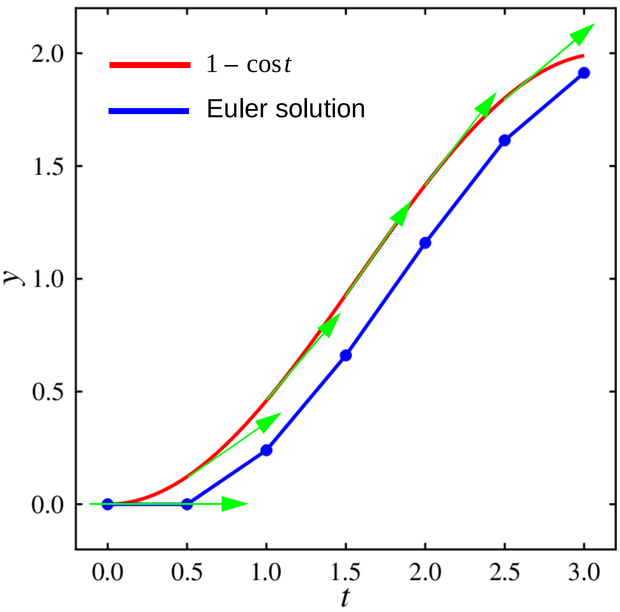
$$\frac{dy(t)}{dt} = \sin t. \; y(0) = 0. \; 0 \le t \le 3.$$

- The exact solution is

$$y(t) = 1 - \cos t.$$

- The Euler algorithm in awk:

```
BEGIN {
  h=0.5; y=0; tm=3.0; n=tm/h;
  print 0,y;
  for (i=0;i<n;i++) {
    t=i*h;
    y=y+h*sin(t);
    t=(i+1)*h
    print t,y;
  }
}
```



- In the figure one can see the graphical interpretation of the alrgorithm.
  - When the problem is well-posed

$$f(t_i, y_i) \approx y'(t_i) = f(t_i, y(t_i)) \; .$$

- Euler's method is easily generalized to groups of equations:

$$\mathbf{y}_{k+1} = \mathbf{y}_k + h\mathbf{f}(t_k, \mathbf{y}_k).$$

Differential equations: initial value problems

- What about the error behavior of the Euler's method?

  - By subtracting the exact solution from the Euler iteration result we get

$$y(t_{k+1}) - y_{k+1} = \underbrace{y(t_k) - y_k + h[f(t_k, y(t_k)) - f(t_k, y_k)]}_{\text{cumulated}} + \underbrace{\frac{h^2 y''(\xi_k)}{2}}_{\text{local}} \quad , \qquad \xi_k \in [t_k, t_{k+1}] . \qquad \textbf{(1)}$$

  - The left hand side of this equation is the total error of the approximation y_{k+1} and it consists of
    1. Errors cumulated from the previous step at t_k
    2. Error formed in computing the current step (*local error*).

  - Error 2 is called local because if we happened to get the exact value in the previous step (k) then the only error that is left is $h^2 y''(\xi_k)/2 \equiv l_{k+1}$.

  - In general the local error is defined as

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

    and $\hat{y}$ is the solution of the local problem:

$$\hat{y}'(t) = f(t, \hat{y}(t))$$
$$\hat{y}(t_{n-1}) = y_{n-1} \quad .$$

  - Due to the local error $h^2 y''(\xi_k)/2 = O(h^2)$ Euler's method is of the order one.

  - In general the order of a method is p if the local error behaves as $O(h^{p+1})$.

Differential equations: initial value problems

  - Using the mean value theorem we can estimate the term in the brackets in **(1)**:

$$h[f(t_k, y(t_k)) - f(t_k, y_k)] = h\frac{\partial f(t_k, \xi_k)}{\partial y}[y(t_k) - y_k] \quad , \qquad \xi_k \in [y(t_k), y_k] .$$

  - Substituting this to eq. **(1)** we get

$$y(t_{k+1}) - y_{k+1} = y(t_k) - y_k + h\frac{\partial f(t_k, \xi_k)}{\partial y}[y(t_k) - y_k] + \frac{h^2 y''(\xi_k)}{2}$$
$$= [y(t_k) - y_k][1 + hf_y(t_k, \xi_k)] + \frac{h^2 y''(\xi_k)}{2}$$

    where $f_y(t_k, \xi_k) = \dfrac{\partial f(t_k, \xi_k)}{\partial y}$

  - So we get the total error at step $k+1$ as

$$e_{k+1} = e_k[1 + hf_y(t_k, \xi_k)] + l_{k+1}$$

e_k: error at previous step; $[1 + hf_y(t_k, \xi_k)]$: amplification factor; l_{k+1}: local error

  - As far as $|1 + hf_y(t_k, \xi_k)| < 1$ the total error in Euler's method does not grow.

  - If $|1 + hf_y(t_k, \xi_k)| > 1$ the error grows during the iteration without limits.

Differential equations: initial value problems

- Let's look at the test problem

$$y' = \lambda y,$$

where λ is a complex constant.

- The solution is $y(t) = y(0)e^{\lambda t}$.

- Applying Euler's method we get

$$y_k = y_{k-1} + h\lambda y_{k-1} = (1 + h\lambda)y_{k-1} = \ldots = y(0)(1 + h\lambda)^k.$$

- Let now $\mathrm{Re}(\lambda) < 0$ so that the solutions are asymptotically stable and approach zero.
- This means that the numerical approximations $|y_k|$ must not grow.

  In Euler's method this means

$$|1 + h\lambda| \le 1.$$

- The **absolute stability region** D of a numerical method consists of those complex numbers $z = h\lambda$ that fulfill the condition

$$|y_k| \le |y_{k-1}|, \quad k = 1, 2, \ldots.$$

- Absolute stability interval is the part of the real axis that belongs to D.

- The absolute stability region of Euler's method is the circle

$$D = \{z \in \mathbf{C} | \ |z + 1| \le 1\}.$$

  and the interval is $[-2, 0]$.

Generally for the Euler's method:

$$|1 + hf_y| < 1.$$

Differential equations: initial value problems

- Example: solving

$$y' = -100y + 100, \ y(0) = y_0.$$

  using Euler's method.

  - Exact solution is $y(t) = (y_0 - 1)e^{-100t} + 1$ and it is clearly stable.

  - Euler's method gives a difference equation

$$y_{k+1} = y_k + h(-100y_k + 100) = (1 - 100h)y_k + 100h$$

    with solution

$$y_k = (y_0 - 1)(1 - 100h)^k + 1.$$

  - Let the initial value be $y_0 = 2$.
  - Exact solution is now $y(t) = e^{-100t} + 1$.
  - Euler's method gives $y_k = (1 - 100h)^k + 1$
  - We see immediately that if $h > 0.02$ the solution $|y_k|$ grows without limit when k increases.

Differential equations: initial value problems

- The error formula of Euler's method contains h both in the error amplification term and in the local error term:

$$e_{k+1} = e_k[1 + hf_y(t_k, \xi_k)] + \frac{h^2 y''(\xi_k)}{2} .$$

- In practical computations h should be
  as large as possible in order to speed up the calculations.
  so small that the error amplification factor < 1
  so small that the local error term is acceptable.

- The derivatives are difficult to compute but the local term can be estimated based on the iterations:

$$y_a'' = \frac{y_k' - y_{k-1}'}{t_k - t_{k-1}}.$$

- By requiring that the local error fulfills the condition

$$\frac{|y_a''|h^2}{2} < \varepsilon$$

  we get the upper limit to the step size

$$h < \sqrt{\frac{2\varepsilon}{|y_a''|}}.$$

Differential equations: initial value problems

- Even if it would be possible - from the point of view of the consumed CPU time - to decrease the step size indefinitely we wouldn't get better and better results: finally the round-off errors show up.

- The figure on the right shows the relative error of the solution of the IVP

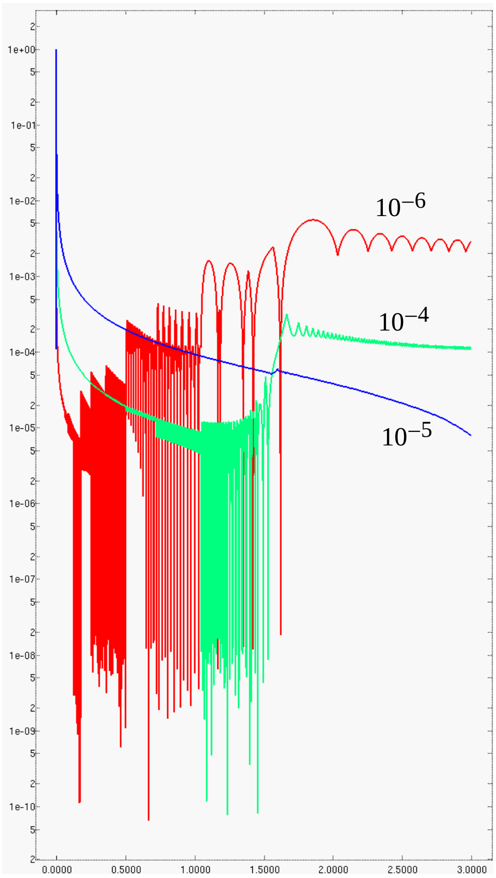
$$y' = \sin t.\ y(0) = 0.\ 0 \le t \le 3$$

  using step sizes

$$h = 10^{-4}, 10^{-5}, 10^{-6}$$

  and using single precision floating point numers.

- You probably remember the same phenomenon in computing the derivative of a function using simple difference quotient.



Differential equations: initial value problems

- We can go further in the Taylor's series

$$y(t_{i+1}) = y(t_i) + (t_{i+1} - t_i)y'(t_i) + \frac{1}{2}(t_{i+1} - t_i)^2 y''(t_i) + \ldots$$

and develop higher order methods.

- These methods have the advantage of higher-order local truncation error.
- However, one has to calculate the higher derivatives $f^{(n)}(t, y)$ which is time consuming in many practical problems.

- An alternative to the Euler's method is an implicit version of it (backward Euler's method):
  - Integrate $y'(t) = f(t, y)$ over one time step

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f(t, y(t))dt .$$

  - By approximating the intergral by $(t_{k+1} - t_k)f(t_k, y_k)$ we get the Euler's method.
  - However, we can also use $(t_{k+1} - t_k)f(t_{k+1}, y_{k+1})$ when we get the backward Euler:

$$y_{k+1} = y_k + hf(t_{k+1}, y_{k+1}) .$$

  - The word implicit means that new point y_{k+1} is on both sides of the iteration formula. It has to be solved by numerical methods.
  - One can show that this methods has a error amplification factor $(1 - hf_y)^{-1}$ which means that when applied to a stable linear problem with constant coefficients when $\mathrm{Re}(\lambda) \le 0$, we get $|1 - h\lambda|^{-1} \le 1$ for all $h > 0$.

Differential equations: initial value problems

- The methods described above were presented to illustrate the important concepts in numerical integration of IVPs.
- They are seldom used in practical applications.
- Probably the most often used method to solve IVPs is the **Runge-Kutta (RK) method** (or one of the many RK methods).
- It offers the higher orders without the need to compute derivatives of f.
- The only drawback is that the number of function evaluations of f per time step may be large.
- We need a couple of results concerning the Taylor's theorem of a function of two variables.
  - Suppose $f(y, t)$ has all its partial derivatives up to $n + 1$ continuous on $D = \{(t, y) | a \le t \le b, c \le y \le d\}$.
  - Let $(t_0, y_0) \in D$.
  - For every $(t, y) \in D$ $\exists(\xi \in [t, t_0], \eta \in [y, y_0])$ with

$$f(t, y) = P_n(t, y) + R_n(t, y)$$

$$P_n(t, y) = f(t_0, y_0) + \left[(t - t_0)\frac{\partial f(t_0, y_0)}{\partial t} + (y - y_0)\frac{\partial f(t_0, y_0)}{\partial y}\right]$$

$$+ \left[\frac{(t - t_0)^2}{2}\frac{\partial^2}{\partial t^2}f(t_0, y_0) + \frac{(y - y_0)^2}{2}\frac{\partial^2}{\partial y^2}f(t_0, y_0) + (t - t_0)(y - y_0)\frac{\partial^2}{\partial t \partial y}f(t_0, y_0)\right]$$

$$+ \ldots + \left[\frac{1}{n!}\sum_{j=0}^{n}\binom{n}{j}(t - t_0)^{n-j}(y - y_0)^j\frac{\partial^n}{\partial t^{n-j}\partial y^j}f(t_0, y_0)\right]$$

$$R_n(t, y) = \frac{1}{(n+1)!}\sum_{j=1}^{n+1}\binom{n+1}{j}(t - t_0)^{n+1}(y - y_0)^j\frac{\partial^{n+1}}{\partial t^{n+1-j}\partial y^j}f(\xi, \eta) .$$

  - In other words P_n is the Taylor polynomial and R_n is the remainder term.

Differential equations: initial value problems

- Let's see how by using the RK method we could increase the order of the Euler's method by one.
- This means that we have to approximate the term

$$T^{(2)}(t,y) = f(t,y) + \frac{h}{2}f'(t,y)$$

with error no greater than $O(h^2)$.
- We take the approximation to have the form $a_1 f(t+\alpha_1, y+\beta_1)$.

- Since $f'(t,y) = \frac{df(t,y)}{dt} = \frac{\partial}{\partial t}f(t,y) + \frac{\partial}{\partial y}f(t,y)y'(t)$ and $y'(t) = f(t,y)$

$\rightarrow T^{(2)}(t,y) = f(t,y) + \frac{h}{2}\frac{\partial}{\partial t}f(t,y) + \frac{h}{2}\frac{\partial}{\partial y}f(t,y)y'(t)$.

Higher order Taylor method:

$$y_0 = y(t_0)$$

$$y_{i+1} = y_i + hT^{(n)}(t_i, y_i)$$

$$T^{(n)} = f(t_i, y_i) + \frac{h}{2}f'(t_i, y_i) + \ldots + \frac{h^{n-1}}{(n+1)!}f^{(n-1)}(t_i, y_i)$$

- Now we expand $f(t+\alpha_1, y+\beta_1)$ into a Taylor polynomial of degree one about (t,y)

$$a_1 f(t+\alpha_1, y+\beta_1) = a_1 f(t,y) + a_1\alpha_1\frac{\partial}{\partial t}f(t,y) + a_1\beta_1\frac{\partial}{\partial y}f(t,y) + a_1 R_1(t+\alpha_1, y+\beta_1)$$

$$R_1(t+\alpha_1, y+\beta_1) = \frac{\alpha_1^2}{2}\frac{\partial^2}{\partial t^2}f(\xi,\eta) + \frac{\beta_1^2}{2}\frac{\partial^2}{\partial y^2}f(\xi,\eta) + \alpha_1\beta_1\frac{\partial^2}{\partial t\partial y}f(\xi,\eta).$$

- The coefficients a_1, α_1, and β_1 can be determined by setting

$$f(t,y) + \frac{h}{2}f'(t,y) = a_1 f(t+\alpha_1, y+\beta_1)$$

$$\rightarrow \quad f(t,y) + \frac{h}{2}\frac{\partial}{\partial t}f(t,y) + \frac{h}{2}\frac{\partial}{\partial y}f(t,y)y'(t) = a_1 f(t,y) + a_1\alpha_1\frac{\partial}{\partial t}f(t,y) + a_1\beta_1\frac{\partial}{\partial y}f(t,y) \quad \text{(remember } y' = f\text{)}$$

and matching the coefficients of f and its derivatives.

Differential equations: initial value problems

- The results is

$$a_1 = 1,\quad \alpha_1 = \frac{h}{2},\quad \beta_1 = \frac{h}{2}f(t,y).$$

and

$$T^{(2)}(t,y) = f\left(t+\frac{h}{2}, y+\frac{h}{2}f(t,y)\right) - R_1\left(t+\frac{h}{2}, y+\frac{h}{2}f(t,y)\right).$$

- If the 2nd partial derivatives of f are bounded then the remainder term R_1 is of order $O(h^2)$.

- The method derived above is a 2nd order (and two-stage) RK method and the algorithm itself is simple:

1. $y_0 = y(0)$.
2. $y_{i+1} = y_i + hf\left(t_i + \frac{h}{2}, y_i + \frac{h}{2}f(t_i, y_i)\right)$.

- This has a name of its own: the **midpoint method**.

Differential equations: initial value problems

- However, the most used RK method is the 4th order version:

$$
\begin{aligned}
y_0 &= y(0) \\
k_1 &= hf(t_i, y_i) \\
k_2 &= hf\left(t_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right) \\
k_3 &= hf\left(t_i + \frac{h}{2}, y_i + \frac{k_2}{2}\right) \\
k_4 &= hf(t_{i+1}, y_i + k_3) \\
y_{i+1} &= y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4).
\end{aligned}
$$

- Note that in this algorithm the function f is evaluated 4 times per time step.

- In general a s-stage RK method can be written as

$$
\begin{aligned}
y_{i+1} &= y_i + h\sum_{j=1}^{s} b_j k_j \\
k_j &= f\left(t_i + c_j h, y_i + h\sum_{l=1}^{2} a_{jl} k_l\right) \\
c_j &= \sum_{l=1}^{s} a_{jl}.
\end{aligned}
$$

Differential equations: initial value problems

- One can describe the RK method as follows:

1. Take an Euler step of length $c_2 h$ from (t_i, y_i). The derivative at the new point is k_2.
2. Return to the original point and take a new Euler step $(c_3 h)$ using the weighted average of k_1 and k_2 as the derivative. We get a new derivative k_3.

- In this way a number of approximation are computed for the derivative and they all are used in the end to compute the step taken from (t_i, y_i).

- The number of stages in a RK method is not directly the order of the method:

Order	1	2	3	4	5	6	7	8	9	10
Stages	1	2	3	4	6	7	9	11	$12 \le s \le 17$	$13 \le s \le 17$

- The table above explains the popularity of the four-stage RK method.

Differential equations: initial value problems

- Most practical routines for integration of an IVP require the user to give an upper limit to the error in the end result.
  - The program then adjusts the time step h accordingly.

  - This means that the algorithm somehow has to be able to estimate the error.

  - One way to do this is to compute the solution at the same time with step sizes h and $2h$.
  - Assuming that the local error introduced by the larger step is twice that of the smaller step one can get an estimate for the error by

$$|l_n| \approx \frac{1}{2}\frac{|\tilde{y}_n - y_n|}{2^p - 1},$$

where $\tilde{y}_n$ and y_n are the solution at time n.

  - However, computing with two time steps is an extra effort.

  - In the RK method the error estimate can be computed by solving the IVP with two RK solvers simultaneously.
  - The two solvers can be chosen in such a way that the derivatives needed by the lower order solver are already computed by the higher order method.
  - This pair of solvers is called the embedded Runge-Kutta pair.
  - They are sometimes also called Runge-Kutta-Fehlberg methods.

  - If the RK method is written in the form $y_{n+1} = y_n + F(t_n, y_n, h; f)$ then the method is stable if F fulfills the Lipschitz condition

$$|F(t, y, h; f) - F(t, z, h; f)| \le L|y - z|,$$

for all $a \le t \le b$, $-\infty < y, z < \infty$.

Differential equations: initial value problems

- Below is a comparison of the maximum error of various one-step methods[1].
  - A one-step method is a method that only uses y_i to compute the next step y_{i+1}.
  - The IVP was $y' = -5ty^2 + \frac{5}{t} - \frac{1}{t^2}$, $y_0 = 1$, $t \in [1, 10]$
  - The exact solution is $y(t) = 1/t$.

h	Euler	Impl. Euler	Trapetsi	RK2	RK4
0.2	3.33e-2	9.23e-3	1.39e-3	1.39e-3	5.34e-3
0.1	9.09e-3	5.21e-3	2.83e-4	2.83e-4	2.04e-4
0.05	3.43e-3	2.80e-3	7.01e-5	7.01e-5	9.65e-6
0.02	1.27e-3	1.17e-3	1.11e-5	1.11e-5	2.09e-7
0.01	6.20e-4	5.97e-4	2.77e-6	2.77e-6	1.23e-8
0.005	3.08e-4	3.01e-4	6.92e-7	6.92e-7	7.47e-10
0.002	1.22e-4	1.21e-4	1.11e-7	1.11e-7	1.88e-11

  - The trapezoidal method is $y_{k+1} = y_k + \frac{h}{2}[f(t_{k+1}, y_{k+1}) + f(t_k, y_k)]$.

1. From J. Haataja et al., *Numeeriset menetelmät käytännössä*, CSC, 1999

Differential equations: initial value problems

- In **multistep (MS) methods** one uses also information from earlier time steps than only the previous one.

  - I.e. we want to use also

  $$\begin{array}{lll} y_{k-1}, & y_{k-2}, & \dots \\ f(t_{k-1}, y_{k-1}), & f(t_{k-2}, y_{k-2}), & \dots \end{array}$$

  to approximate the next point y_{k+1}.

  - In general a r step linear multistep method can be written as $f_j \equiv f(t_j, y_j)$

  $$\sum_{j=-r+1}^{1} \alpha_j y_{k+j} = h \sum_{j=-r+1}^{1} \beta_j f_{k+j}$$

  where the coefficients fulfill the conditions

  $$\alpha_1 = 1, \alpha_{-r+1} \neq 0, \beta_{-r+1} \neq 0.$$

  - Methods can be derived by approximating the integral in

  $$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f(t, y(t))dt$$

  with a polynomial

  $$P(t) = a_0 + a_1 t + a_2 t^2 + \dots + a_s t^s$$

  and by integrating it.

Differential equations: initial value problems

- The unknown coefficients a_i are determined by fixing the polynomial to certain values y_i and f_i, $i = k+1, k, k-1, \dots$.

- The good point in the MS methods is that the function f is only evaluted once per time step.

- On the other hand, in the beginning of the iteration there are no previous data points available.
  - These can be generated by perfoming a couple of steps using a one-step method.

- Also, changing the time step is not so straightforward in MS methods.

- MS methods are divide to

  explicit (open) methods, $\beta_1 = 0$ and

  implicit (closed) methods, $\beta_1 \neq 0$.

Differential equations: initial value problems

• Of the explicit MS methods the **Adams-Bashford** (**AB**) is maybe the most well known.

- It is of the form

$$y_{k+1} = y_k + h \sum_{j=-r+1}^{0} \beta_j f_{k+j} .$$

- The unknown new iteration point does not show up in the RHS of the equation so this is an open method.

- Below are the five first AB methods

$$y_{k+1} = y_k + hf_k$$

$$y_{k+1} = \frac{h}{2}[3f_k - f_{k-1}]$$

$$y_{k+1} = y_k + \frac{h}{12}[23f_k - 16f_{k-1} + 5f_{k-2}]$$

$$y_{k+1} = y_k + \frac{h}{24}[55f_k - 59f_{k-1} + 37f_{k-2} - 9f_{k-3}]$$

$$y_{k+1} = y_k + \frac{h}{720}[1901f_k - 2774f_{k-1} + 2616f_{k-2} - 1274f_{k-3} + 251f_{k-4}] \quad .$$

- One can show that an AB method with r steps is of the order $O(h^r)$.

- The stability interval of the AB methods are rather narrow, so that they are not suitable for stiff problems.

• The implicit **Adams-Moulton** (**AM**) method is more suitable for stiff problems:

$$y_{k+1} = y_k + h \sum_{j=-r+2}^{1} \beta_j f_{k+j} .$$

- The five first AM methods are

$$y_{k+1} = y_k + hf_{k+1}$$

$$y_{k+1} = y_k + \frac{h}{2}[f_{k+1} + f_k]$$

$$y_{k+1} = y_k + \frac{h}{12}[5f_{k+1} + 8f_k - f_{k-1}]$$

$$y_{k+1} = y_k + \frac{h}{24}[9f_{k+1} + 19f_k - 5f_{k-1} + f_{k-2}]$$

$$y_{k+1} = y_k + \frac{h}{720}[251f_{k+1} + 646f_k - 264f_{k-1} + 106f_{k-2} + 19f_{k-3}] \quad .$$

- When $r \geq 2$ the order of the AM method is $r + 1$.

- The absolute stability intervals of the methods are

1	$[-\infty, 0]$
2	$[-6, 0]$
3	$[-8, 0]$
4	$[-1.8, 0]$

Differential equations: initial value problems

- The AM methods give better results than the AB methods.

  - However, the drawback of the AM method is that due to its implicity one has to solve the new iteration point from a group of equations.

- A combination method often used is so called predictor-corrector method.
  - In it the AB method produces a initial prediction to the new y value while the AM method is used to correct it more precise.
  - The correction may be done many times but not until the final convergence.
  - For example the pair of 3rd degree:

$$\begin{cases} y_{k+1}^{\mathrm{p}} = y_k + \frac{h}{12}[23f_k - 16f_{k-1} + 5f_{k-2}] \\ y_{k+1}^{\mathrm{c}} = y_k + \frac{h}{12}[5f(t_{k+1}, y_{k+1}^{\mathrm{p}}) + 8f_k - f_{k-1}] \end{cases} .$$

Differential equations: initial value problems

• So called backward differentiation formulae (BDF) methods are well suited for stiff ODEs.
  - They are derived by requiring a polynomial going through the previous iteration points:

$$P_r(t) = \sum_{j=-1}^{r-1} y(t_{n-j}) l_{jn}(t)$$, where $l_{jn}(t)$ is the Lagrange interpolation polynomial.

  - Because P_r is (an approximate) solution

$$P_r'(t_{n+1}) = y'(t_{n+1}) = f(t_{n+1}, y(t_{n+1}))$$

  - Combining the above equations and solving $y(t_{n+1})$ we get

$$y_{n+1} = \sum_{j=0}^{r-1} \alpha_j y_{n-j} + h\beta f(t_{n+1}, y_{n+1}),$$

  where α_j and β are obtained by expanding the Lagrange interpolation polynomial in the first equation above.

  - The method is implicit and the order is r.
  - Below are the five first BDF methods:

$$y_{k+1} = y_k + hf_{k+1}$$ (This is backward Euler.)

$$y_{k+1} = \frac{4}{3}y_k - \frac{1}{3}y_{k-1} + \frac{2}{3}hf_{k+1}$$

$$y_{k+1} = \frac{18}{11}y_k - \frac{9}{11}y_{k-1} + \frac{2}{11}y_{k-2} + \frac{6}{11}hf_{k+1}$$

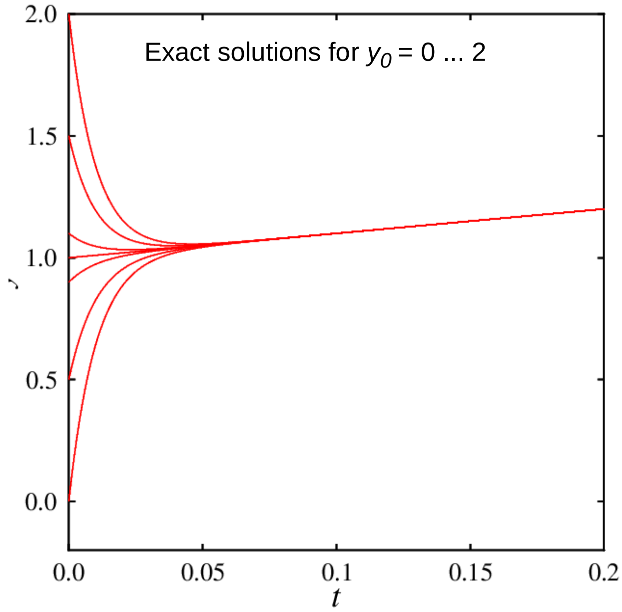
$$y_{k+1} = \frac{48}{25}y_k - \frac{36}{25}y_{k-1} + \frac{16}{25}y_{k-2} - \frac{3}{25}y_{k-3} + \frac{12}{25}hf_{k+1}$$

$$y_{k+1} = \frac{300}{137}y_k - \frac{300}{137}y_{k-1} + \frac{200}{137}y_{k-2} - \frac{75}{137}y_{k-3} + \frac{12}{137}y_{k-4} + \frac{60}{137}hf_{k+1} .$$

Differential equations: initial value problems

• **Stiff ODEs** are those where there are two or more disparate time scales.

- An example: $y' = -100y + 100t + 101$, $y(0) = y_0$.
  - Solution: $y(t) = (y_0 - 1)e^{-100t} + t + 1$
  - By varying the initial value y_0 slightly causes large deviations in the initial behavior of the solution.
  - Euler method is results very sensitive to the initial value: even a small error during the iteration causes the iteration to pick a solution with a different y_0; particularly one with $y_0 \neq 1$.
  - Morever, with large enough time steps we get outside the stability interval.
  - **Demo.**



  - Stiffness is difficult to define exactly:
    1) Process described by the ODE contains disparate timescales.
    2) A group of ODEs is stiff if the eigenvalues of the Jacobian $\mathbf{J}$ ($J_{ij} = \partial f_i / \partial y_j$) has eigenvalues that differ greatly in magnitude.
    3) When using an explicit method a much smaller time step must be used as which would be dictated by the accuracy criteria.

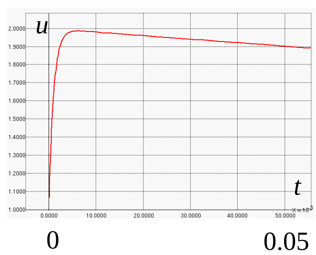
Differential equations: initial value problems

- Example (from NR, section 16.6)

$$\begin{cases} u' = 998u + 1998v \\ v' = -999u - 1999v \end{cases}.$$

- Exact solution

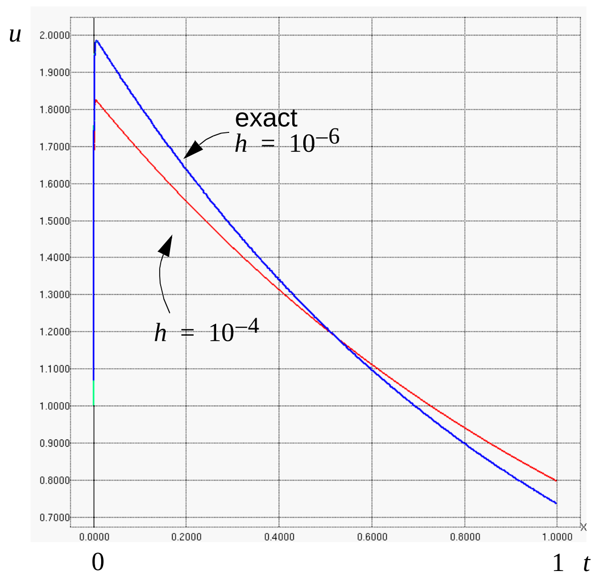
$$\begin{cases} u = 2e^{-t} - e^{-1000t} \\ v = -e^{-t} + e^{-1000t} \end{cases}.$$



- Using Euler's method we have to use really small steps:

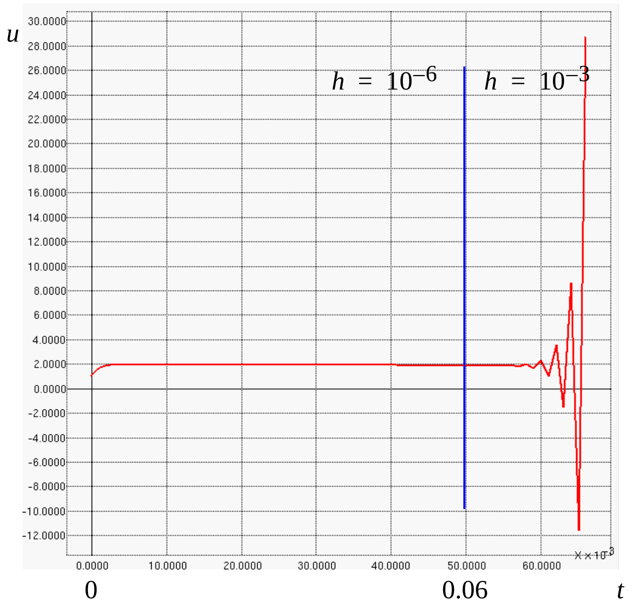
- Although the smoothness of the function would allow a longer time step outside the initial transient behavior (term e^{-1000t})

- What about having a small step in the beginning and increasing it at - say - $t > 10^{-2}$



Differential equations: initial value problems

- Below is the results of a version of Euler which changes the stepsize after a certain time:



- This does not help because the instability of the transient is still there though the solution itself is a smoothly varying function.
- Or to put it in another way: In order the solution to be stable we must use a smaller time step than the one based only on the accuracy criteria.

Differential equations: initial value problems

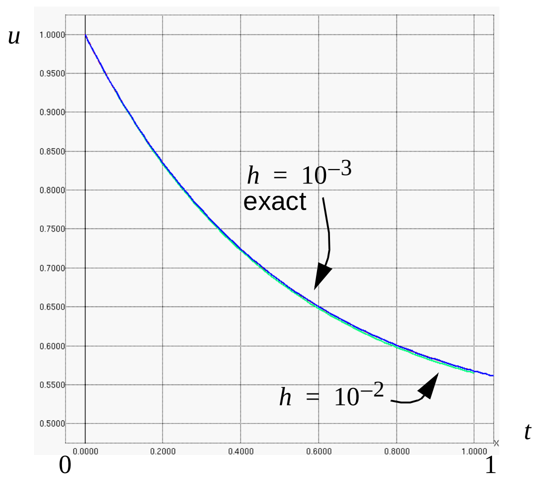
- For comparison consider the following equation that is not stiff but has a rather similar solution:

$$\begin{cases} u' = -u + v \\ v' = u - v \end{cases}$$

- And the solution with initial conditions $u(0) = 1$, $v(0) = 0$ is

$$\begin{cases} u = \frac{1}{2}(1 + e^{-2t}) \\ v = \frac{1}{2}(1 - e^{-2t}) \end{cases}.$$

- Using Euler with $h = 10^{-3}, 10^{-2}$ gives reasonable results:



Differential equations: initial value problems

- For stiff ODEs the implicit methods work better.
- Take a simple example:

$$y' = -cy\,,\; c > 0\,.$$

- Euler's iteration gives us

$$y_{n+1} = y_n + hy'_n = (1 - hc)y_n\,.$$

- The method is unstable if $h > \frac{2}{c}$.

- Now using implicit method (backward Euler) we get

$$y_{n+1} = y_n + hy'_{n+1} \text{ or}$$

$$y_{n+1} = \frac{y_n}{1 + ch}\,.$$

- This method is stable for all values of h.

- So with implicit methods we can use larger time steps in considering the *stability* of the method.
- However, the accuracy may require to use smaller stepsize.

Differential equations: initial value problems

• Practical tools:

GSL: routines for many stepping algorithms and for timestep control (including various Runge-Kutta and implicit methods)
Matlab: also many routines: say `help funfun` to get a list.
Maxima: symbolic solver, example solving the last example equation:

```
Maxima 5.9.1 http://maxima.sourceforge.net
Using Lisp CMU Common Lisp 19a
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
This is a development version of Maxima. The function bug_report()
provides bug reporting information.
(%i1) e1:'diff(u(t),t)=-u(t)+v(t);
```

(%o7) $\frac{d}{d\,t}u(t) = v(t) - u(t)$

```
(%i8) e2:'diff(v(t),t)=u(t)-v(t);
```

(%o8) $\frac{d}{d\,t}v(t) = u(t) - v(t)$

```
(%i9) desolve([e1,e2],[u(t),v(t)]);
```

(%o9) $\left[u(t) = \frac{v(0)+u(0)}{2} - \frac{(v(0)-u(0))\,e^{-2t}}{2}, v(t) = \frac{(v(0)-u(0))\,e^{-2t}}{2} + \frac{v(0)+u(0)}{2}\right]$

```
(%i10)
```

Differential equations: initial value problems

- Practical tools (continued)
  - **SLATEC** library includes (at least) the following algorithms:
    - `DDEABM`: Adams-Bashforth method
    - `DDEBDF`: backward differentiation; for stiff problems
    - `DDERKF`: Runge-Kutta-Fehlberg method
  - User should, in addition, provide the function that calculates the RHS of the equation(s) and for the backward differentiation method Jacobian $\mathbf{J}$ of the problem needed:

$$\begin{cases} y_1' = f_1(t, y_1, \ldots, y_N) \\ y_2' = f_2(t, y_1, \ldots, y_N) \\ \ldots \\ y_N' = f_N(t, y_1, \ldots, y_N) \end{cases} \qquad \mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial y_1} & \frac{\partial f_1}{\partial y_2} & \ldots & \frac{\partial f_1}{\partial y_N} \\ \frac{\partial f_2}{\partial y_1} & \frac{\partial f_2}{\partial y_2} & \ldots & \frac{\partial f_2}{\partial y_N} \\ \ldots & \ldots & \ldots & \ldots \\ \frac{\partial f_N}{\partial y_1} & \frac{\partial f_N}{\partial y_2} & \ldots & \frac{\partial f_N}{\partial y_N} \end{bmatrix}$$

- **What method to use?**
  For non-stiff ODEs explicit RK, or predictor-corrector and AM if computing the derivative is expensive.
  For stiff ODEs BDF and implicit RK.

  - **Now a nice demo.**

Differential equations: boundary value problems[1]

- **Boundary value problems** (**BVP**s) are harder to solve because the conditions have to be fulfilled in more than one point.
  - The general form of 1D BVP is

$$\begin{cases} \mathbf{y}' = \mathrm{f}(x, \mathbf{y}) \\ \mathbf{g}(\mathbf{y}(a), \mathbf{y}(b)) = 0 \end{cases} \quad (\mathbf{y} \text{ and } \mathbf{g} \text{ have } N \text{ components})$$

  - Linear boundary conditions (BC) can be written in the form $B_a\mathbf{y}(a) + B_b\mathbf{y}(b) = \mathbf{b}$.
  - If the BCs are separable they can be written as

$$\mathbf{y}(a) = \mathbf{b}_a$$
$$\mathbf{y}(b) = \mathbf{b}_b.$$

  - Sometimes the DE and BC contain a parameter λ:

$$\begin{cases} \mathbf{y}' = \mathrm{f}(x, \mathbf{y}, \lambda) \\ \mathbf{g}(\mathbf{y}(a), \mathbf{y}(b), \lambda) = 0 \end{cases}$$

  and the objective is to determine those values of λ for which the system has non-trivial solutions (eigenvalues and eigenfunctions).
    - E.g. Schrödinger equation

$$\left[-\frac{1}{2}\nabla^2 + V(\mathbf{r})\right]\phi(\mathbf{r}) = E\phi(\mathbf{r})$$

      - In 1D a particle constrained by infinite potential walls:

$$\begin{cases} -\frac{1}{2}\phi''(x) + V(x)\phi(x) = E\phi(x) \\ \phi(a) = 0 \\ \phi(b) = 0 \end{cases}$$

, which can be written as a system of 1st order DEs $\begin{cases} \psi(x) = 2[V(x) - E]\phi(x) \\ \phi(x) = \psi'(x) \end{cases}$

1. Sources: J. Haataja et al., *Numeeriset menetelmät käytännössä*, CSC, 1999; R. L. Burden, J. D. Faires, *Numerical Analysis*, PWS-KENT, 1989.

Differential equations: boundary value problems

- In the **shooting method** (**SM**) we utilize methods developed for IVPs.

  - We integrate the DE starting from a chosen initial condition:

    $$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \ \mathbf{y}(a) = \mathbf{s}.$$

  - We have to choose $\mathbf{s}$ so that the solution $\mathbf{y}(x;\mathbf{s})$ fulfills the BC

    $$\mathbf{g}(\mathbf{y}(a;\mathbf{s}), \mathbf{y}(b;\mathbf{s})) = 0.$$

  - The solution of the original BVP is thus $\mathbf{y}(x;\mathbf{s}^*)$ where $\mathbf{s}^*$ is the solution of the group of equations

    $$\mathbf{F}(\mathbf{s}) = \mathbf{g}(\mathbf{s}, \mathbf{y}(b;\mathbf{s})) = 0.$$

    - This generally a non-linear group of equations.

Differential equations: boundary value problems

- A common form of BVPs is the 2nd order equation

  $$y'' = f(x, y, y'), \ a \le x \le b,$$
  $$y(a) = \alpha,$$
  $$y(b) = \beta.$$

- This is converted to a pair of 1st order DEs as ($y \to y_1$, $y' \to y_2$)

  $$\begin{cases} y_1' = y_2 \\ y_2' = f(x, y_1, y_2) \end{cases}.$$

- Existence of the solution:

  - Assume that the function f and its derivatives $\partial f / \partial y$, $\partial f / \partial y'$ are continuous in

    $$D = \{(x, y, y') | a \le x \le b, -\infty \le y \le \infty, -\infty \le y' \le \infty\}.$$

  - Now, if for all for all $(x, y, y') \in D$ the following apply

    a) $\frac{\partial}{\partial y} f(x, y, y') > 0$ and

    b) $\exists M$ such that $\left|\frac{\partial}{\partial y'} f(x, y, y')\right| \le M$

    then the abovementioned BVP has a unique solution.

Differential equations: boundary value problems

- An example:

$$y'' + e^{-xy} + \sin y' = 0,\ x \in [1, 2],\ y(1) = 0,\ y(2) = 0$$

or $\quad y'' = f(x, y, y'),\ f(x, y, y') = -e^{-xy} - \sin y'.$

Now $\frac{\partial}{\partial y} f(x, y, y') = xe^{-xy} > 0,\ \left|\frac{\partial}{\partial y'} f(x, y, y')\right| = |-\cos y'| \le 1.$

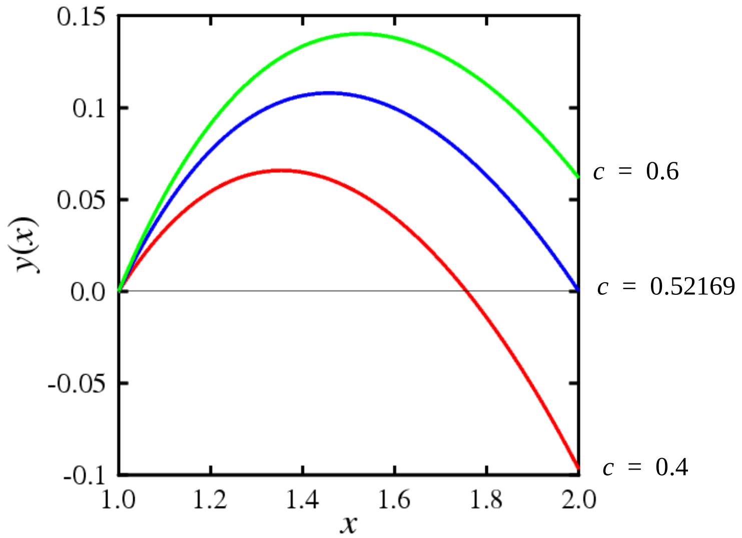
So, the equation has a unique solution.

Solution of the IVP

$$y'' + e^{-xy} + \sin y' = 0$$
$$y(1) = 0$$
$$y'(1) = c$$

with values

$$c = \begin{cases} 0.6 \\ 0.52169 \\ 0.4 \end{cases}$$



Differential equations: boundary value problems

- The **linear** form of the 2[nd] order BVP can be written as

$$y'' = p(x)y' + q(x)y + r(x),\ a \le x \le b,\ y(a) = \alpha,\ y(b) = \beta. \quad \textbf{(1)}$$

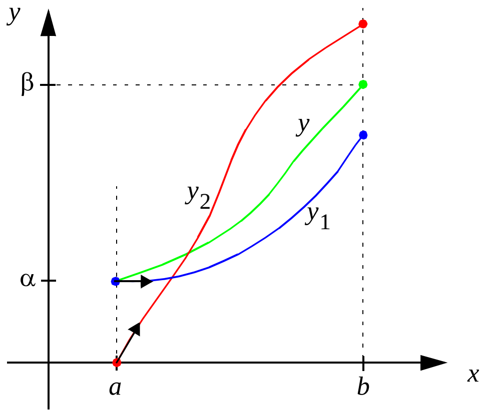
- For this form the existence of a unique solution requires that
  a) $p(x)$, $q(x)$, $r(x)$ are continuous in $[a, b]$ and
  b) $q(x) > 0$ in $[a, b]$.

- Consider two IVPs based on the BVP above:
  $y'' = p(x)y' + q(x)y + r(x),\ x \in [a, b],\ y(a) = \alpha,\ y'(a) = 0,$
  solution $y_1(x)$
  $y'' = p(x)y' + q(x)y,\ x \in [a, b],\ y(a) = 0,\ y'(a) = 1,$
  solution $y_2(x)$

- It is easy to show that the unique solution to the original equation **(1)** can be expressed in terms of $y_1(x)$ and $y_2(x)$:

$$y(x) = y_1(x) + \frac{\beta - y_1(b)}{y_2(b)} y_2(x),$$

assuming $y_2(b) \neq 0$[1].



- A demo using GSL.

1. No need to worry about this. One can show that if y_2 is the solution of $y'' = p(x)y' + q(x)y$ with $y_2(a) = y_2(b) = 0$ then $y_2(x) = 0 \quad \forall x \in [a, b]$.

Differential equations: boundary value problems

- In the **multiple shooting method** (**MSM**) the interval is divided into smaller parts and the SM is applied to every part so that the parts join to each other continuously.
  - Using the method you end up solving a (possibly nonlinear) group of equations.

  - Divide the interval $[a, b]$ to parts

$$a = x_1 < x_2 < \ldots < x_m = b.$$

  - Let $\mathbf{y}'(x; x_k, \mathbf{s}_k)$ be the solution to the IVP $\mathbf{y}' = \mathbf{f}(x, \mathbf{y})$, $\mathbf{y}(x_k) = \mathbf{s}_k$ on the interval $x \in [x_k, x_{k+1}]$.

  - In the **MSM** method we pursue the vectors $\mathbf{s}_k$, $k = 1, 2, \ldots, m-1$ that satisfy the continuity and boundary conditions

$$\mathbf{y}(x_{k+1}; x_k, \mathbf{s}_k) = \mathbf{s}_{k+1}, \ k = 1, 2, \ldots, m-2$$
$$\mathbf{g}(\mathbf{s}_1, \mathbf{y}(b; x_{m-1}, \mathbf{s}_{m-1})) = 0.$$

  - The unknown initial values $\mathbf{s} = (\mathbf{s}_1\ \mathbf{s}_2\ \ldots\ \mathbf{s}_{m-1})^T$ are obtained from the nonlinear system of equations

$$\mathbf{F}(\mathbf{s}) = \begin{bmatrix} \mathbf{y}(x_2; x_1, \mathbf{s}_1) - \mathbf{s}_2 \\ \mathbf{y}(x_3; x_2, \mathbf{s}_2) - \mathbf{s}_3 \\ \ldots \\ \mathbf{g}(\mathbf{s}_1, \mathbf{y}(x_m; x_{m-1}, \mathbf{s}_{m-1})) \end{bmatrix} = 0$$

- The **finite difference** method is not based on solving the corresponding IVP but the BVP is discretized to a finite mesh and derivatives are approximated by finite differences.
  - An example shows how the method works[1]

  - The BVP is

$$y'' = f(x, y, y')$$
$$y(a) = \alpha$$
$$y(b) = \beta.$$

  - Divide interval $[a, b]$ to m parts: $h = \frac{b-a}{m}$, $x_i = a + ih$, $i = 0, \ldots, m$
  - Approximate the derivatives in the BVP by finite differences:

$$y'(x_n) \approx \frac{y_{n+1} - y_{n-1}}{2h}$$
$$y''(x_n) \approx \frac{y_{n+1} - 2y_n + y_{n-1}}{h^2}.$$

  - Now we get the difference equation

$$y_{n+1} - 2y_n + y_{n-1} = h^2 f\left(x_n, y_n, \frac{y_{n+1} - y_{n-1}}{2h}\right), \quad n = 1, 2, \ldots, m-1$$
$$y_0 = \alpha, \ y_m = \beta.$$

1. J.Haataja et al., *Numeeriset menetelmät käytännössä*, example 7.5.2

Differential equations: boundary value problems

- In matrix form

$$\mathbf{A}\mathbf{y} = h^2\mathbf{f}(\mathbf{y})$$

$$\mathbf{A} = \begin{bmatrix} -2 & 1 & 0 & \dots & 0 & 0 \\ 1 & -2 & 1 & \dots & 0 & 0 \\ 0 & 1 & -2 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -2 & 1 \\ 0 & 0 & 0 & \dots & 1 & -2 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \dots \\ y_{m-1} \end{bmatrix}, \quad \mathbf{f}(\mathbf{y}) = \begin{bmatrix} f\left(x_1, y_1, \frac{y_2 - y_0}{2h}\right) - \alpha \\ f\left(x_2, y_2, \frac{y_3 - y_1}{2h}\right) \\ f\left(x_3, y_3, \frac{y_4 - y_2}{2h}\right) \\ \dots \\ f\left(x_{m-1}, y_{m-1}, \frac{y_m - y_{m-2}}{2h}\right) - \beta \end{bmatrix}.$$

- Note that this is a nonlinear group of equations.

ifferential equations: boundary value problems

- There are also methods based on expansion of the solution in terms of basis functions; e.g. **collocation method** (**CM**), **Galerkin's method**.

- Take for example the equation

$$y'' = f(x, y, y'),\ y(a) = \alpha,\ y(b) = \beta.$$

- We expand the solution in terms of basis functions $\{\phi_1, \phi_2, \dots, \phi_N\}$:

$$y(x) = \sum_{i=1}^{N} c_i \phi_i(x).$$

- The basis set can be chosen in an appropriate way considering the boundary conditions.
- In the **CM** method the above expansion is substituted in the equation and we require that it solves the equation in N points x_j within the interval $[a, b]$:

$$\sum_{i=1}^{N} c_i \phi_i''(x_j) = f\left(x_j, \sum_{i=1}^{N} c_i \phi_i(x_j), \sum_{i=1}^{N} c_i \phi_i'(x_j)\right),\ j = 1, 2, \dots, N.$$

- From this equation we can determine the coefficients $\{c_1, c_2, \dots, c_N\}$.

Differential equations: boundary value problems

- Practical tools:

  Matlab: see `help funfun` for more information.

  NAG: many routines for BVPs.

  SLATEC: many routines for BVPs.

  NETLIB: see the package `ode` (*http://www.netlib.org/ode/*)