Ordinary Differential Equations

1 Introduction: first order ODE

We are given
- a function \( f(t, y) \) which describes a “direction field” in the \((t, y)\) plane
- an initial point \((t_0, y_0)\)

We want to find a function \( y(t) \) for \( t \in [t_0, T] \) such that
- \( y(t_0) = y_0 \) “initial condition”
- \( y'(t) = f(t, y(t)) \) for \( t \in [t_0, T] \) “ordinary differential equation” (ODE)

This is called an initial value problem (IVP).

The partial derivative \( f_y(t, y) = \frac{\partial f}{\partial y}(t, y) \) is important for the behavior of the differential equation.

**Theorem 1.1.** Assume that \( f(t, y) \) and \( f_y(t, y) \) are continuous for \( t \in [t_0, T] \), \( y \in \mathbb{R} \).
For a given initial value \( y_0 \) there is a unique solution \( y(t) \) of the initial value problem. Either the solution exists for all \( t \in [0, T] \), or it only exists on a smaller interval \([t_0, t_∗)\) with \( t_0 < t_∗ < T \).

We can solve the IVP in Matlab with \texttt{ode45}:

```matlab
f = @(t,y) ... % define function f(t,y)
[ts,ys] = ode45(f,[t0,T],y0); % find column vectors ts,ys with values of solution
ys(end) % value y(T) of solution
plot(ts,ys) % plot solution
```

**Example:** Find a function \( y(t) \) for \( t \in [-1, 3] \) such that

\[
y'(t) = t - y(t)^2
\]
\[
y(-1) = 0
\]

Here we have \( t_0 = -1 \), \( y_0 = 0 \), \( f(t, y) = t - y^2 \).

**Numerical solution in Matlab:** (using m-file \texttt{dirfield.m} from course web page)

```matlab
f = @(t,y) t-y^2 % define function f(t,y)
dirfield(f, -1:.2:3, -1:.2:1.6); hold on % plot direction field
[ts,ys] = ode45(f,[-1,3],0); % solve IVP for t from -1 to 3, initial value 0
% this gives vectors ts,ys
plot(ts,ys,'b'); hold off % plot solution
```
What happens if we perturb the initial value $y_0$?

**Theorem 1.2.** Let $y(t)$ denote the solution of the IVP with initial condition $y(t_0) = y_0$, let $\tilde{y}(t)$ denote the solution of the IVP with initial condition $\tilde{y}(t_0) = \tilde{y}_0$. Assume $f_y(t, y) \leq M$ for $t \in [t_0, T]$, $y \in \mathbb{R}$. Then

$$|\tilde{y}(t) - y(t)| \leq |\tilde{y}_0 - y_0| e^{M(t-t_0)} \quad \text{for } t \in [t_0, T]$$

For $M < 0$ the difference $|\tilde{y}(t) - y(t)|$ decays exponentially for increasing $t$. For $M > 0$ the difference may increase exponentially.

We call the ODE **unstable** if we have $f_y(t, y) > 0$ for all $t \in [t_0, T]$, $y \in \mathbb{R}$.

We call the ODE **stable** if we have $f_y(t, y) < 0$ for all $t \in [t_0, T]$, $y \in \mathbb{R}$.

## 2 System of ODEs, higher order ODEs

We want to find $n$ functions $y_1(t), \ldots, y_n(t)$ for $t \in [t_0, T]$ satisfying the differential equations

$$y'_1(t) = f_1(t, y_1(t), \ldots, y_n(t))$$
$$\vdots$$
$$y'_n(t) = f_n(t, y_1(t), \ldots, y_n(t))$$

and the initial conditions $y_1(t_0) = y_1(0), \ldots, y_n(t_0) = y_n(0)$.

We use vector notation: E.g., for $n = 2$ we want to find $\vec{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$ such that

$$\begin{bmatrix} y'_1(t) \\ y'_2(t) \end{bmatrix} = \begin{bmatrix} f_1(t, y_1(t), y_2(t)) \\ f_2(t, y_1(t), y_2(t)) \end{bmatrix}, \quad \begin{bmatrix} y_1(t_0) \\ y_2(t_0) \end{bmatrix} = \begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix}$$

$$\vec{y}'(t) = \vec{f}(t, \vec{y}(t)), \quad \vec{y}(t_0) = \vec{y}(0)$$
We will omit the vector arrows from now on.

We denote by \(D_y f(t, y)\) the Jacobian of \(f(t, y)\) with respect to \(y\):

\[
D_y f(t, y) = \begin{bmatrix}
    \frac{\partial f_1}{\partial y_1} & \cdots & \frac{\partial f_1}{\partial y_n} \\
    \vdots & \ddots & \vdots \\
    \frac{\partial f_n}{\partial y_1} & \cdots & \frac{\partial f_n}{\partial y_n}
\end{bmatrix}
\]

It is important for the behavior of the differential equation.

**Theorem 2.1.** Assume that \(f(t, y)\) and \(D_y f(t, y)\) are continuous for \(t \in [t_0, T]\), \(y \in \mathbb{R}^n\).

For a given initial value \(y^{(0)}\) there is a unique solution \(y(t)\) of the initial value problem. Either the solution exists for all \(t \in [0, T]\), or it only exists on a smaller interval \([t_0, t_*]\) with \(t_0 < t_* < T\).

We can solve the IVP in Matlab with **ode45**: For \(n = 2\) we use

\[
\begin{align*}
    f &= @(t,y) \ [ \ldots ; \ldots ] \quad \text{% define function } f(t,y) \text{ using } t, y(1), y(2) \\
    [ts,ys] &= \text{ode45}(f, [t0,T], y0); \quad \text{% find column vector ts, array ys with values of solution} \\
    \text{ys(end,:)} &= \text{values y1, y2 at final time } T \\
    \text{plot}(ts,ys(:,1)) &= \text{plot solution } y1(t)
\end{align*}
\]

**2nd order ODE**

So far the differential equations only contained the first derivative \(y'(t)\). But in many applications (e.g. Newton’s law) we have differential equations containing \(y''(t)\). We then need initial conditions for \(y(t_0)\) and \(y'(t_0)\).

We can **rewrite this as a first order system**: Let \(y_1(t) := y(t)\) and \(y_2(t) := y'(t)\). Then we have \(y_1' = y_2\) and \(y_2' = \cdots\) where we solve the 2nd order ODE for \(y''\).

**Example:** Find a function \(y(t)\) for \(t \in [0, 4]\) such that

\[
\begin{align*}
y''(t) - y'(t) + 3y(t) &= t \\
    y(0) &= 1, \quad y'(0) = -2
\end{align*}
\]

This gives the first order system

\[
\begin{bmatrix}
y_1' \\
y_2'
\end{bmatrix} = \begin{bmatrix}
y_2 \\
t + y_2 - 3y_1
\end{bmatrix}, \quad \begin{bmatrix}
y_1(0) \\
y_2(0)
\end{bmatrix} = \begin{bmatrix}
-1 \\
2
\end{bmatrix}
\]

**Numerical solution in Matlab:** Print out \(y(T)\) and plot the function \(y(t)\)

\[
\begin{align*}
f &= @(t,y) \ [y(2); t+y(2)-3*y(1)]; \quad \text{% define function } f(t,y) \\
[ts,ys] &= \text{ode45}(f, [0,4],[1;2]); \quad \text{% solve IVP for } t \text{ from } 0 \text{ to } 4, \text{ initial value } [-1;2] \\
\text{finalval} &= \text{ys(end,1)} \quad \text{% value of } y1 \text{ at final time } T \\
\text{plot}(ts,ys(:,1)) &= \text{plot solution } y1(t)
\end{align*}
\]

**3 Euler method**

Consider a first order system of ODEs: We want to find \(y(t)\) for \(t \in [t_0, T]\) such that

\[
y'(t) = f(t, y(t)), \quad y(t_0) = y^{(0)}
\]

For the **Euler method** we divide the interval \([t_0, T]\) into \(N\) subintervals of equal length \(h = (T-t_0)/N\) (we can also use subintervals of different length). Let \(t_j = t_0 + jh\). We then want to find approximations \(y^{(1)}, \ldots, y^{(N)}\) for \(y(t_j)\).
• start at the initial value $t_0, y^{(0)}$
• for $k = 0, \ldots, N - 1$ do
  
  $s := f(t_k, y^{(k)})$
  $y^{(k+1)} := y^{(k)} + hs$
  $t_{k+1} := t_k + h$

**Example:** Consider the Initial Value Problem (1), (2). Use 2 steps of the Euler method with $h = \frac{1}{2}$ to find approximations for $y(1)$ and $y'(1)$. We use $y_1(t) := y(t)$ and $y_2(t) = y'(t)$ and obtain the first order ODE (3).

**Step 1:** $s = f(t_0, y^{(0)}) = f \left( 0,\begin{bmatrix} -1 \\ 2 \end{bmatrix} \right) = \begin{bmatrix} 2 \\ 5 \end{bmatrix}$, $y^{(1)} = y^{(0)} + s = \begin{bmatrix} -1 \\ 2 \end{bmatrix} + \frac{1}{2} \cdot \begin{bmatrix} 2 \\ 5 \end{bmatrix} = \begin{bmatrix} 0 \\ 4.5 \end{bmatrix}$

**Step 2:** $s = f(t_1, y^{(1)}) = f \left( \frac{1}{2}, \begin{bmatrix} 0 \\ 4.5 \end{bmatrix} \right) = \begin{bmatrix} 4.5 \\ 5 \end{bmatrix}$, $y^{(2)} = y^{(1)} + s = \begin{bmatrix} 0 \\ 4.5 \end{bmatrix} + \frac{1}{2} \cdot \begin{bmatrix} 4.5 \\ 5 \end{bmatrix} = \begin{bmatrix} 2.25 \\ 7 \end{bmatrix}$

This gives $y(1) \approx 2.25$ and $y'(1) \approx 7$.

**Errors for Euler method**

We consider the case $n = 1$. At time $t_k$ the Euler method gives an approximation $y_k$ for the exact value $y(t_k)$. We denote the **error** by

$$e_k := y_k - y(t_k)$$

By Taylor’s theorem we have with a remainder term $r_k = \frac{1}{2} y''(\tau_k) h^2$

$$y(t_{k+1}) = y(t_k) + h \cdot f(t_k, y(t_k)) + r_k$$

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

The second equation is just the definition of the Euler approximation $y_{k+1}$. Subtracting the first from the second equation gives

$$e_{k+1} = e_k + h \cdot \left[ f(t_k, y_k) - f(t_k, y(t_k)) \right] - r_k$$

Using the mean value theorem for $g(y) := f(t_k, y)$

$$f(t_k, y_k) - f(t_k, y(t_k)) = f_y(t_k, \eta_k) \cdot [y_k - y(t_k)]$$

hence the new error is

$$e_{k+1} = \frac{1 + h f_y(t_k, \eta_k)}{a_k} e_k - r_k$$

with the **amplification factor** $a_k = 1 + h f_y(t_k, \eta_k)$ and the local **truncation error** $r_k = \frac{1}{2} y''(\tau_k) h^2$.

For an unstable ODE we have $f_y(t, y) > 0$ and hence $a_k > 1$.

For a stable ODE we have $f_y(t, y) < 0$ and hence $a_k < 1$. However, in this case we want $|a_k| < 1$, i.e.,

$$-1 < 1 + h f_y(t, y) < 1$$

The right inequality is true for any $h > 0$. The left inequality is true if the following **stability condition** holds:

$$h < \frac{2}{-f_y}$$
(1) General case:

We assume

\[ |f_y(t,y)| \leq C_1 \quad \text{for } t \in [t_0, T], \quad y \in \mathbb{R} \]
\[ |y''(t)| \leq C_2 \quad \text{for } t \in [t_0, T] \]

Then we get bounds \(|a_k| \leq A\) for the amplification factor and \(|r_k| \leq R\) for the local truncation error:

\[ |a_k| = |1 + hf_y(t_k, \eta_k)| \leq 1 + hC_1 =: A \]
\[ |r_k| = \left| \frac{1}{2} y''(\tau_k)h^2 \right| \leq \frac{C_2}{2} h^2 =: R \]

yielding

\[ |e_{k+1}| \leq A|e_k| + R \]

Since \(|e_0| = 0\) we obtain

\[ |e_2| \leq AR + R = (1 + A)R \]
\[ |e_3| \leq A(1 + A)R + R = (1 + A + A^2)R \]
\[ \vdots \]
\[ |e_k| \leq \left(1 + A + \cdots + A^{k-1}\right)R \quad (4) \]

We have for the geometric series

\[ 1 + A + \cdots + A^{k-1} = \frac{A^k - 1}{A - 1} \leq \frac{A^k}{A - 1} = \frac{(1 + hC_1)^k}{hC_1} \]

The function \(e^x\) satisfies \(1 + x \leq e^x\), hence with \(x = hC_1\) we get \(1 + hC_1 \leq e^{hC_1}\). Using this and \(R = \frac{C_2}{2}h^2\) in (4) gives the error bound

\[ |y_k - y(t_k)| \leq \frac{C_2}{2C_1} e^{C_1(t_k - t_0)} \]

This shows:

- if we keep taking Euler steps with a fixed value \(h\) for \(t \to \infty\) the error can increase exponentially. This is not surprising: For an unstable ODE any tiny initial error can cause an exponentially increasing error for \(t \to \infty\).
- if we only want to find the solution for \(t \in [t_0, T]\): We use \(h = \frac{T-t_0}{N}\) and obtain errors bounded by \(ch = c'N^{-1}\): The Euler method is a method of order 1.

Example: The initial value problem

\[ y' = y - \sin t - \cos t, \quad y(0) = 1 \]

has the solution \(y(t) = \cos t\). Here \(f_y(t,y) = 1 > 0\). We use the Euler method with \(h = 0.1\):

\[ f = @(t,y) y - \sin(t) - \cos(t) \]
\[ tv = 0:1:5; \text{plot}(tv,\cos(tv),'b'); \text{hold on} \quad \%	ext{plot exact solution} \]
\[ \text{dirfield}(f,0:3:5,-1:2:3); \]
\[ [ts,ys] = \text{Euler}(f,[0,5],1,50); \quad \%	ext{use 50 steps of size 5/50} \]
\[ \text{plot}(ts,ys,'r.-'); \text{hold off} \]
We see that the Euler values go exponentially to $+\infty$ as $t$ gets larger than 4, whereas the exact solution $y(t) = \cos t$ stays bounded.

(2) Special case: Stable ODE where $h$ satisfies stability condition:

We assume $f_y(t, y) < 0$: We have $C_1 \geq C_0 > 0$ such that

\[-C_1 \leq f_y(t, y) \leq -C_0 \quad \text{for } t \in [t_0, T], \; y \in \mathbb{R} \]
\[|y''(t)| \leq C_2 \quad \text{for } t \in [t_0, T] \]

We now want to have an amplification factor with $|a_k| \leq 1 - C_0 h$, i.e.,

\[-(1 - C_0 h) \leq 1 + h f_y(t_k, \eta_k) \leq 1 - C_0 h \]

The right inequality holds for any $h > 0$. We have $1 - h C_1 \leq 1 + h f_y(t_k, \eta_k)$, therefore the left inequality holds if $-(1 - C_0 h) \leq 1 - h C_1$ or

\[h \leq \frac{2}{C_0 + C_1} \tag{5} \]

If $h$ satisfies this stability condition we have $|a_k| \leq 1 - C_0 h =: A < 1$, hence

\[1 + A + \cdots + A^{k-1} = \frac{1 - A^k}{1 - A} \leq \frac{1}{1 - A} \]

and (4) now gives

\[|y_k - y(t_k)| \leq \frac{C_2}{2C_0} h \]

This shows:

- if we keep taking Euler steps with a fixed value $h$ for $t \to \infty$ the error is bounded by $Ch$ with a fixed constant $C$.

**Example:** The initial value problem

\[y' = -y - \sin t + \cos t, \quad y(0) = 1 \]

has the solution $y(t) = \cos t$. Here $f_y(t, y) = -1 < 0$. We use the Euler method with $h = 0.2$:
Here we have an error of size $Ch$, but the error stays bounded as $t \to \infty$.

4 Improved Euler method (aka RK2 method)

For the Euler method we obtained a local truncation error $r_k$ satisfying $|r_k| \leq ch^2$. Since we use $N = (T - t_0)/h$ steps to get from $t_0$ to the final time $T$, we obtained the error at time $T$ was bounded by $Ch^3$.

The Euler method is a method of order 1. This means that in order to improve the error by a factor of 10, we need to use 10 times as many steps. If we want to achieve an error of size $10^{-8}$ we may need of the order of $10^8$ steps (assuming e.g. a stable problem with $C_2$ and $C_0$ of about 1).

We would like to have a method of order 2, i.e., the error at time $T$ is bounded by $Ch^2$. This means that the local truncation error should satisfy $|r_k| \leq ch^3$. How can we do this? We need more than one evaluation of the direction field per step.

We start at the initial point $(t_0, y_0)$. We take a step of size $h$ to $t_1 = h$ and want to find an approximation $y_1$ for $y(t_1)$. We know that $y'(t) = f(t, y(t))$, so by the fundamental theorem of calculus we have

$$y(t_1) = y_0 + \int_{t_0}^{t_1} f(t, y(t)) \, dt$$

Let $g(t) := f(t, y(t))$. Then we have to approximate the integral $I = \int_{t_0}^{t_1} g(t) \, dt$ with an error $\leq ch^3$. One way to do this is to use the trapezoid rule: Recall that on an interval of size $h$ we have $|Q^{\text{Trap}} - I| \leq \frac{h^3}{12} \max |g''|$. Therefore we want to use

$$I \approx Q^{\text{Trap}} = \frac{h}{2} [g(t_0) + g(t_1)] = \frac{h}{2} \left[ f(t_0, y_0) + f(t_1, y(t_1)) \right]$$

But we cannot evaluate the second term $f(t_1, y(t_1))$ since we don’t know $y(t_1)$. So we use the best approximation we have: the Euler approximation $y^{\text{Euler}} = y_0 + h \cdot f(t_0, y_0)$. Since $|y^{\text{Euler}} - y(t_1)| \leq Ch^2$ we get from the mean value theorem

$$|f(t_1, y^{\text{Euler}}) - f(t_1, y(t_1))| = \left| \frac{\partial f}{\partial y}(t_1, \eta) \cdot (y^{\text{Euler}} - y(t_1)) \right| \leq C_1 \cdot Ch^2$$
We now approximate \( y(t_1) \) by
\[
y_1 := y_0 + \frac{h}{2} \left[ f(t_0, y_0) + f(t_1, y^{\text{Euler}}) \right]
\] (6)
and obtain for the local truncation error
\[
|y_1 - y(t_1)| \leq Ch^3
\]
since the error of the trapezoid rule is bounded by \( ch^3 \), and replacing \( y(t_1) \) by \( y^{\text{Euler}} \) causes an additional error \( \frac{1}{2}C_1 Ch^2 \).

The iteration (6) gives the **Improved Euler method**: we divide the interval \([t_0, T]\) into \( N \) subintervals of equal length \( h = (T - t_0)/N \) (we can also use subintervals of different length). Let \( t_j = t_0 + jh \). We then want to find approximations \( y^{(1)}, \ldots, y^{(N)} \) for \( y(t_j) \).

- start at the initial value \( t_0, y^{(0)} \)
- for \( k = 0, \ldots, N - 1 \) do
  \[
s^{(1)} := f(t_k, y^{(k)})
  
y^{E} := y^{(k)} + hs^{(1)}
  
s^{(2)} := f(t_k + h, y^{E})
  
y^{(k+1)} := y^{(k)} + \frac{1}{2} \left[ s^{(1)} + s^{(2)} \right]
  
t_{k+1} := t_k + h
\]

The local truncation error of the improved Euler method is of order \( O(h^3) \). Hence the error at a time \( t = T \) is of order \( O(h^2) = O(N^{-2}) \). **The improved Euler method is a method of order 2.** Note that we use two evaluations of the function \( f \) per step: \( s^{(1)} = f(t_k, y^{(k)}) \) and \( s^{(2)} = f(t_k + h, y^{E}) \).

**Example:** Consider the Initial Value Problem (1), (2). Use 1 step of the Improved Euler method with \( h = \frac{1}{2} \) to find approximations for \( y\left(\frac{1}{2}\right) \) and \( y\left(\frac{1}{2}\right) \).

We use \( y_1(t) := y(t) \) and \( y_2(t) = y(t) \) and obtain the first order ODE (3).

\[
s^{(1)} = f(t_0, y^{(0)}) = f\left(0, \left[ -\frac{1}{2} \right] \right) = \left[ \frac{2}{3} \right],
  
y^{E} = y^{(0)} + hs^{(1)} = \left[ -\frac{1}{2} \right] + \frac{1}{2} \cdot \left[ \frac{2}{3} \right] = \left[ 0, \frac{1}{3} \right]
  
s^{(2)} = f(t_1, y^{E}) = f\left(\frac{1}{2}, \left[ \frac{1}{3} \right] \right) \approx \left[ 4, \frac{5}{3} \right],
  
y^{(1)} = y^{(0)} + \frac{h}{2} \left( s^{(1)} + s^{(2)} \right) = \left[ -\frac{1}{2} \right] + \frac{1}{2} \left( \left[ \frac{2}{3} \right] + \left[ 4, \frac{5}{3} \right] \right) = \left[ 0, \frac{1}{3} \right]
\]
This gives \( y\left(\frac{1}{2}\right) \approx 0.625 \) and \( y\left(\frac{1}{2}\right) \approx 4.5 \).

### 5 Stiff ODE and ode15s

Consider a “very stable” ODE where \( f_y(t, y) \) is very negative. Then the Euler method only works if the step size \( h \) satisfies the stability condition \( h < \frac{2}{-f_y} \). This can force use to use very tiny steps even if the solution \( y(t) \) is almost constant. This is called a **stiff ODE**. In this case **ode45** uses many tiny steps and takes a long time.

**Example:** A flame propagation model gives the following IVP:

\[
y' = y^2 - y^3 \quad \text{for } t \in [0, \frac{2}{\delta}]
  
y(0) = \delta
\]

Here \( \delta \) is very small, e.g., \( \delta = 10^{-4} \). In this case we want to solve the problem for \( t \in [0, \frac{2}{\delta}] = [0, 2 \cdot 10^4] \). The solution approaches \( y = 1 \). But there the problem becomes stiff: We have near \( y = 1 \) that \( f_y(t, y) = 2y - 3y^2 \approx -1 \), so the stability condition for the Euler method requires \( h < \frac{2}{-f_y} = 2 \). This means that we need \( N = 10^4 \) steps of size \( h = 2 \) to get from \( t_0 = 0 \) to \( T = 2/\delta \), despite the fact that the solution is almost constant for most of \([0, T]\).

The adaptive method **ode45** (with default settings) also requires about \( 10^4 \) steps:
\begin{verbatim}
delta = 1e-4;
f = @(t,y) y^2-y^3;
y0 = delta;
t0 = 0; T = 2/delta;
[ts,ys] = ode45(f,[0,T],y0);
length(ts)  \% print number of steps
\end{verbatim}

This prints out 12113 for the number of steps.

Matlab has a special ode solver \texttt{ode15s} for stiff ODEs: We try this for our problem
\begin{verbatim}
[ts,ys] = ode15s(f,[0,T],y0);
length(ts)  \% print number of steps
\end{verbatim}

and get 108 for the number of steps.

\section{Backward Euler method (aka implicit Euler method)}

For stable problems the Euler method gives magnification factors \(|a_k| = |1 + hf| < 1\) for small \(h\), but \(|a_k| = |1 + hf| > 1\) for large \(h\).

If I look at a problem with \(f_y < 0\) from right to left with decreasing \(t\), then an Euler method in decreasing \(t\) direction always has a magnification factor \(|a| > 1\), for any step size \(h > 0\).

This suggests to use a \textit{"backward Euler step"}:

At time \(t_k\) we have the value \(y(k)\).

For time \(t_{k+1} = t_k + h\) we want to find a value \(y^{(k+1)}\) such that an Euler step to the left takes us to \(t_k\) and \(y(k)\):

\textbf{Find} \(y^{(k+1)}\) \textbf{such that} \(y^{(k+1)} - h \cdot f(t_{k+1}, y^{(k+1)}) = y^{(k)}\) \hfill (7)

Note that the unknown vector \(y^{(k+1)}\) occurs also inside the function \(f\). If the function \(f(t,y)\) is linear in \(y\) this gives linear equations for \(y\). If the function \(f(t,y)\) is nonlinear in \(y\) this gives nonlinear equations for \(y\). We can e.g. use 1 or 2 steps of the Newton method (note that we have a local truncation error of size \(O(h^2)\)).

\textbf{Example:} Consider the Initial Value Problem (1), (2). Use 1 step of the backward Euler method with \(h = \frac{1}{2}\) to find approximations for \(y(\frac{1}{2})\) and \(y'(\frac{1}{2})\).

We use \(y_1(t) := y(t)\) and \(y_2(t) = y'(t)\) and obtain the first order ODE (3). Note that the function \(f(t,y) = \begin{bmatrix} y_2 \\
t + y_2 - 3y_1 \end{bmatrix}\) depends linearly on \(y_1, y_2\).

We want to find a vector \(y^{(1)} = \begin{bmatrix} y_1 \\
y_2 \end{bmatrix}\) such that

\[
\begin{bmatrix} y_1 \\
y_2 \end{bmatrix} - \frac{1}{2} \begin{bmatrix} y_1 \\
y_2 \end{bmatrix} = \begin{bmatrix} -1 \\
2 \end{bmatrix}
\]

This gives the linear system \(\begin{bmatrix} y_1 - \frac{1}{2} y_2 \\
\frac{3}{2} y_1 + \frac{1}{2} y_2 \end{bmatrix} = \begin{bmatrix} -1 \\
2.25 \end{bmatrix}\) \text{which has the solution} \(y^{(1)} = \begin{bmatrix} y_1 \\
y_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\
3 \end{bmatrix}\) \text{. This gives} \(y^{(\frac{1}{2})} \approx 0.5\) \text{and} \(y'(\frac{1}{2}) \approx 3\).

\textbf{Claim:} Let \(n = 1\). For a stable problem with \(f_y(t,y) < 0\) the nonlinear equation has a unique solution.

\textbf{Proof:} The left hand side \(F(y_{k+1}) := y_{k+1} - h \cdot f(t_{k+1}, y_{k+1})\) is strictly increasing for increasing \(y_{k+1}\), with \(F(y) \to -\infty\) for \(y \to -\infty\) and \(F(y) \to \infty\) for \(y \to \infty\).