Module 2

Initial Value Problems

Akash Anand
MATH, IIT KANPUR

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Euler's method is an explicit method in that it uses only information at time t_k to advance to solution to time t_{k+1}.

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Euler's method is an explicit method in that it uses only information at time t_k to advance to solution to time t_{k+1}.

This is convenient but comes at the cost of having a rather limited region of stability.

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Euler's method is an explicit method in that it uses only information at time t_k to advance to solution to time t_{k+1}.

This is convenient but comes at the cost of having a rather limited region of stability.

A larger stability region can be obtained by using information at t_{k+1}!

Module 2

Initial Value Problems

2.2 Stability

2.3 Euler's method

2.4 Implicit method

- Backward Euler method

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Euler's method is an explicit method *in that it uses only information at time* t_k to advance to solution to time t_{k+1}.

This is convenient but comes at the cost of having a rather limited region of stability.

A larger stability region can be obtained by using information at t_{k+1}!

For the IVP $y' = f(t, y), y(t_0) = y_0$, *we have*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f\big(s, y(s)\big) ds$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Euler's method is an explicit method *in that it uses only information at time* t_k to advance to solution to time t_{k+1}.

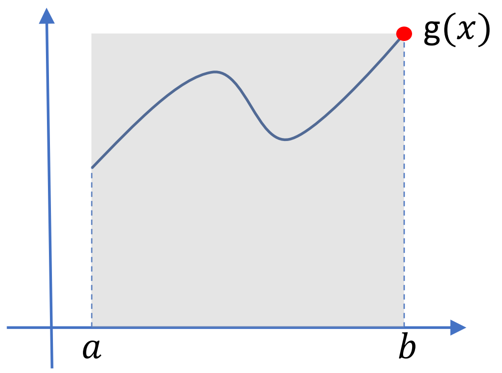
This is convenient but comes at the cost of having a rather limited region of stability.

A larger stability region can be obtained by using information at t_{k+1}!

For the IVP $y' = f(t, y), y(t_0) = y_0$, *we have*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f\big(s, y(s)\big)ds$$

Using the right end-point rule, that is,



$$\int_a^b g(s)ds \approx (b-a)g(b)$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Euler's method is an explicit method *in that it uses only information at time* t_k to advance to solution to time t_{k+1}.

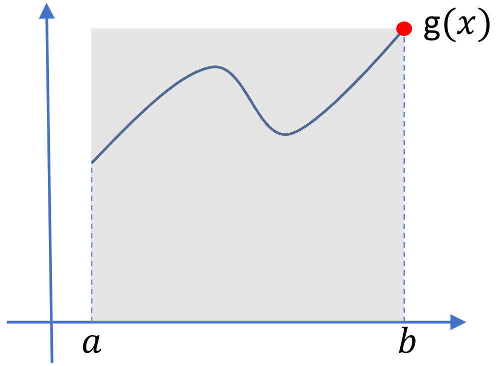
This is convenient but comes at the cost of having a rather limited region of stability.

A larger stability region can be obtained by using information at t_{k+1}!

For the IVP $y' = f(t, y), y(t_0) = y_0$, *we have*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f\big(s, y(s)\big) ds$$

Using the right end-point rule, that is,



$$\int_a^b g(s) ds \approx (b-a) g(b)$$

we get the method

$$y_{k+1} = y_k + h_k f(t_{k+1}, y_{k+1})$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

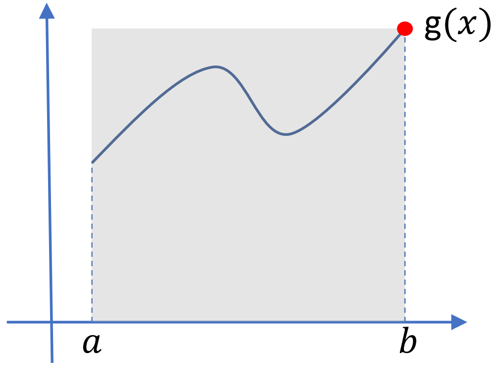
Euler's method is an explicit method *in that it uses only information at time* t_k to advance to solution to time t_{k+1}.

This is convenient but comes at the cost of having a rather limited region of stability.

A larger stability region can be obtained by using information at t_{k+1}!

For the IVP $y' = f(t, y), y(t_0) = y_0$, *we have*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f\big(s, y(s)\big)ds$$



Using the right end-point rule, that is,

$$\int_a^b g(s)ds \approx (b-a)g(b)$$

we get the method

$$\boxed{y_{k+1} = y_k + h_k f(t_{k+1}, y_{k+1})}$$

Background Euler method

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Euler's method is an explicit method *in that it uses only information at time* t_k to advance to solution to time t_{k+1}.

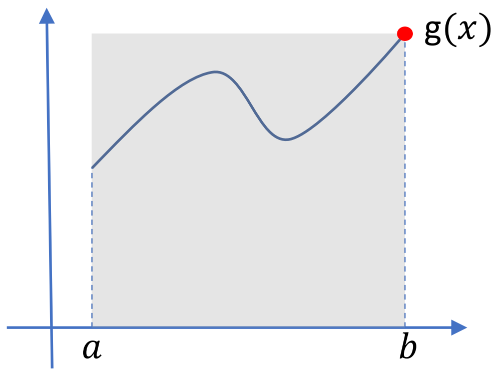
This is convenient but comes at the cost of having a rather limited region of stability.

A larger stability region can be obtained by using information at t_{k+1}!

For the IVP $y' = f(t, y), y(t_0) = y_0$, *we have*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f\big(s, y(s)\big)ds$$

Using the right end-point rule, that is,



$$\int_a^b g(s)ds \approx (b-a)g(b)$$

we get the method

$$\boxed{y_{k+1} = y_k + h_k f(t_{k+1}, y_{k+1})}$$

Background Euler method

When y_{k+1} *appears on the right-hand side of the method*

$$y_{k+1} = \Phi(f, y_{k+1}, y_k, h_k)$$

then the method is called an implicit method.

Example

Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5$.

Example

Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5.$

$$y_1 = y_0 + hf(t_1, y_1)$$

Example

Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5.$

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

Example

Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5$.

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

This is a nonlinear equation that can be solved using some of the techniques you learnt in the first course.

Initial Value Problems: Implicit Methods

Example

Let us solve $y' = -y^3, \; y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5.$

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

This is a nonlinear equation that can be solved using some of the techniques you learnt in the first course. For example, we could use Newton's method to iteratively solve for y_1*:*

Example

Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5.$

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

This is a nonlinear equation that can be solved using some of the techniques you learnt in the first course. For example, we could use Newton's method to iteratively solve for y_1*:*

$$y_1^{(n+1)} = y_1^{(n)} - \frac{0.5\left(y_1^{(n)}\right)^3 + y_1^{(n)} - 1}{1.5\left(y_1^{(n)}\right)^2 + 1},$$

Example

Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5.$

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

This is a nonlinear equation that can be solved using some of the techniques you learnt in the first course. For example, we could use Newton's method to iteratively solve for y_1*:*

$$y_1^{(n+1)} = y_1^{(n)} - \frac{0.5\left(y_1^{(n)}\right)^3 + y_1^{(n)} - 1}{1.5\left(y_1^{(n)}\right)^2 + 1}, \qquad y_1^{(0)} = ?$$

Example

Let us solve $y' = -y^3,\ y(0) = 1$ using the backward Euler method taking the uniform step size $h = h_k = t_{k+1} - t_k = 0.5$.

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

This is a nonlinear equation that can be solved using some of the techniques you learnt in the first course. For example, we could use Newton's method to iteratively solve for y_1:

$$y_1^{(n+1)} = y_1^{(n)} - \frac{0.5\left(y_1^{(n)}\right)^3 + y_1^{(n)} - 1}{1.5\left(y_1^{(n)}\right)^2 + 1}, \qquad y_1^{(0)} = y_0$$

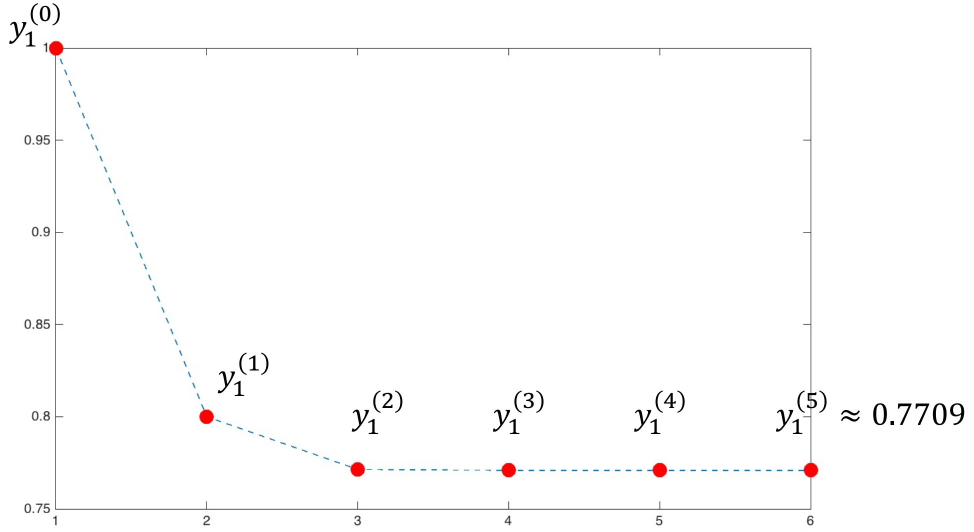
Example

Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5$.

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

This is a nonlinear equation that can be solved using some of the techniques you learnt in the first course. For example, we could use Newton's method to iteratively solve for y_1*:*

$$y_1^{(n+1)} = y_1^{(n)} - \frac{0.5\left(y_1^{(n)}\right)^3 + y_1^{(n)} - 1}{1.5\left(y_1^{(n)}\right)^2 + 1}, \qquad y_1^{(0)} = y_0$$



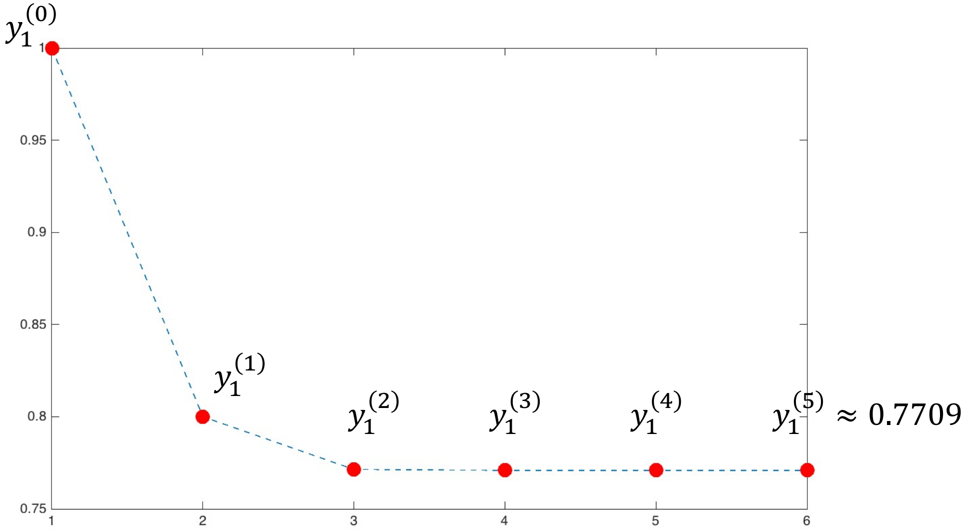
Example

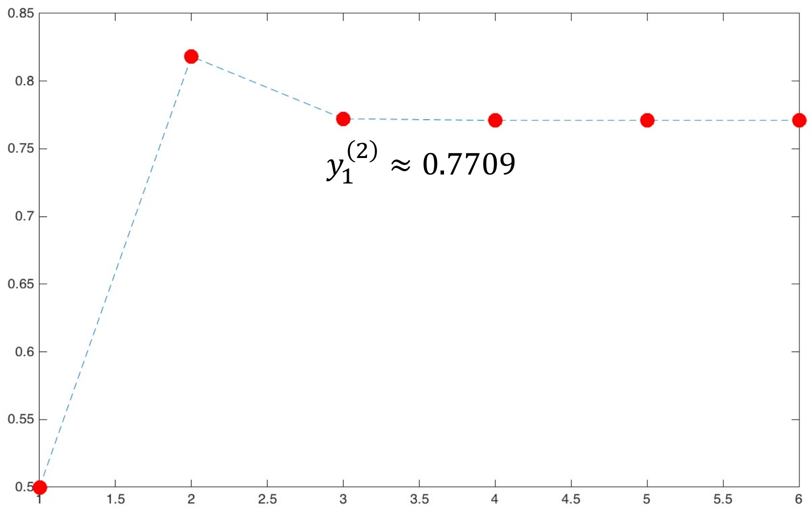
Let us solve $y' = -y^3,\ y(0) = 1$ *using the backward Euler method taking the uniform step size* $h = h_k = t_{k+1} - t_k = 0.5.$

$$y_1 = y_0 + hf(t_1, y_1) = 1 - 0.5y_1^3$$

This is a nonlinear equation that can be solved using some of the techniques you learnt in the first course. For example, we could use Newton's method to iteratively solve for y_1*:*

$$y_1^{(n+1)} = y_1^{(n)} - \frac{0.5\left(y_1^{(n)}\right)^3 + y_1^{(n)} - 1}{1.5\left(y_1^{(n)}\right)^2 + 1}, \qquad y_1^{(0)} = y_0 - 0.5y_0^3 = 0.5$$





Why can we always solve the resulting non-linear system $y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$*?*

Initial Value Problems: Implicit Methods

Why can we always solve the resulting non-linear system $y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$*?*

Define

$$F(z) = y_k + hf(t_{k+1}, z)$$

Why can we always solve the resulting non-linear system $y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$*?*

Define

$$F(z) = y_k + hf(t_{k+1}, z)$$

then

$$|F(z_2) - F(z_1)| = \left|\big(y_k + hf(t_{k+1}, z_2)\big) - \big(y_k + hf(t_{k+1}, z_1)\big)\right|$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Why can we always solve the resulting non-linear system $y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$*?*

Define

$$F(z) = y_k + hf(t_{k+1}, z\ )$$

then

$$|F(z_2) - F(z_1)| = \left|\left(y_k + hf(t_{k+1}, z_2\ )\right) - \left(y_k + hf(t_{k+1}, z_1\ )\right)\ \right|$$

$$|F(z_2) - F(z_1)| \leq h|f(t_{k+1}, z_2\ ) - f(t_{k+1}, z_1\ )| \leq hL|z_2 - z_1|$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Why can we always solve the resulting non-linear system $y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$*?*

Define

$$F(z) = y_k + hf(t_{k+1}, z)$$

then

$$|F(z_2) - F(z_1)| = \left|\left(y_k + hf(t_{k+1}, z_2)\right) - \left(y_k + hf(t_{k+1}, z_1)\right)\right|$$

$$|F(z_2) - F(z_1)| \le h|f(t_{k+1}, z_2) - f(t_{k+1}, z_1)| \le hL|z_2 - z_1|$$

Therefore, if $h < 1/L$ *, then* F *is a contraction. Thus, it has a fixed point, say* y_{k+1}.

Why can we always solve the resulting non-linear system $y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$*?*

Define

$$F(z) = y_k + hf(t_{k+1}, z)$$

then

$$|F(z_2) - F(z_1)| = \left|\left(y_k + hf(t_{k+1}, z_2)\right) - \left(y_k + hf(t_{k+1}, z_1)\right)\right|$$

$$|F(z_2) - F(z_1)| \leq h|f(t_{k+1}, z_2) - f(t_{k+1}, z_1)| \leq hL|z_2 - z_1|$$

Therefore, if $h < 1/L$ *, then* F *is a contraction. Thus, it has a fixed point, say* y_{k+1}*. That is,*

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

Initial Value Problems: Implicit Methods

Why can we always solve the resulting non-linear system $y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$?

Define

$$F(z) = y_k + hf(t_{k+1}, z)$$

then

$$|F(z_2) - F(z_1)| = \left|\left(y_k + hf(t_{k+1}, z_2)\right) - \left(y_k + hf(t_{k+1}, z_1)\right)\right|$$

$$|F(z_2) - F(z_1)| \leq h|f(t_{k+1}, z_2) - f(t_{k+1}, z_1)| \leq hL|z_2 - z_1|$$

Therefore, if $h < 1/L$ *, then* F *is a contraction. Thus, it has a fixed point, say* y_{k+1}*. That is,*

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

Remark:

In the previous example, we could have used the fixed point iteration $y_{k+1}^{(n+1)} = y_k + hf\left(t_{k+1}, y_{k+1}^{(n)}\right)$ *in place of Newton's iterations*

$$y_{k+1}^{(n+1)} = y_{k+1}^{(n)} - \frac{0.5\left(y_{k+1}^{(n)}\right)^3 + y_{k+1}^{(n)} - 1}{1.5\left(y_{k+1}^{(n)}\right)^2 + 1}.$$

To understand the stability, let us again consider the ODE $y' = \lambda y$.

To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

Initial Value Problems: Implicit Methods

To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

To understand the stability, let us again consider the ODE $y' = \lambda y$. *Applying backward Euler method*

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$

For stability of the backward Euler method, we need

$$\left|\frac{1}{1-h\lambda}\right| \leq 1 \quad \text{or} \quad |1 - \lambda h| \geq 1.$$

To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$

For stability of the backward Euler method, we need

$$\left|\frac{1}{1-h\lambda}\right| \leq 1 \quad \text{or} \quad |1 - \lambda h| \geq 1.$$

Note that this condition holds for all $h > 0$ where $Re(\lambda) < 0$.

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

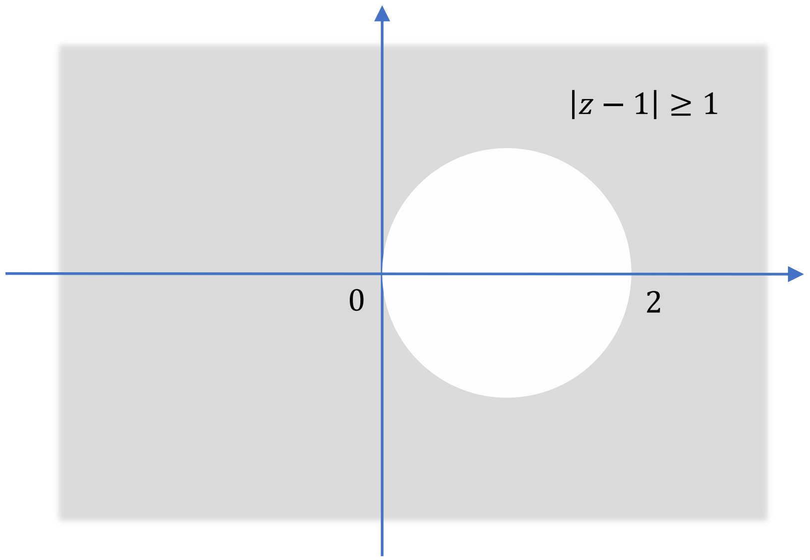
To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$



For stability of the backward Euler method, we need

$$\left|\frac{1}{1-h\lambda}\right| \leq 1 \quad \text{or} \quad |1 - \lambda h| \geq 1.$$

Note that this condition holds for all $h > 0$ where $Re(\lambda) < 0$.

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

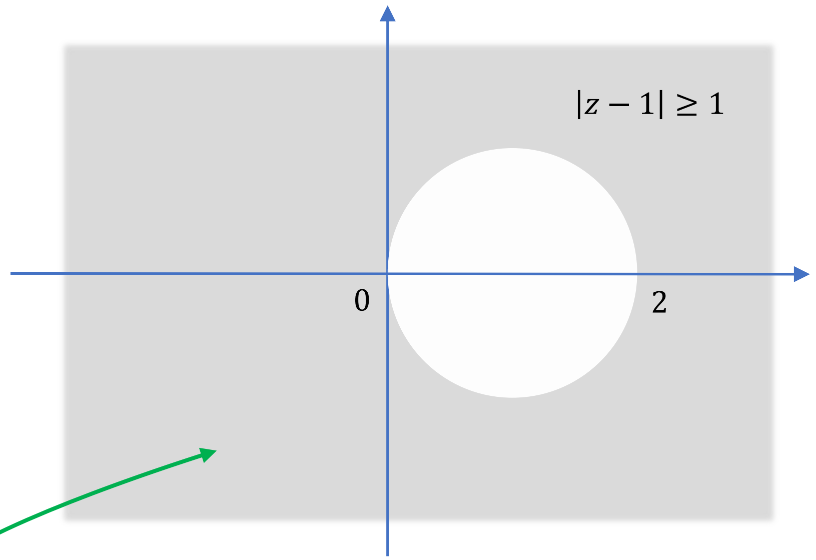
$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$

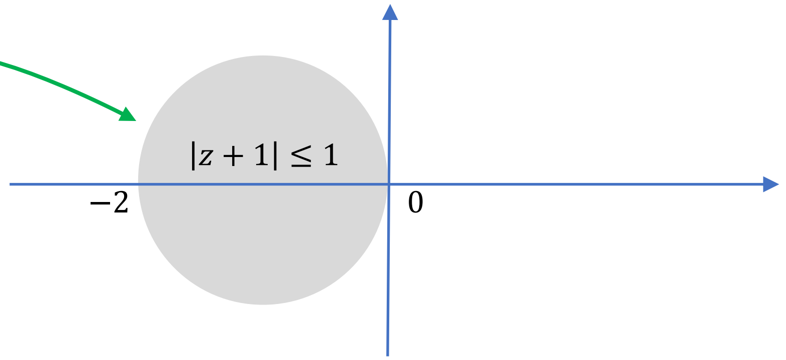
For stability of the backward Euler method, we need

$$\left|\frac{1}{1-h\lambda}\right| \leq 1 \quad \text{or} \quad |1 - \lambda h| \geq 1.$$

Note that this condition holds for all $h > 0$ where $Re(\lambda) < 0$.



Compare with Euler's method



Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

To understand the stability, let us again consider the ODE $y' = \lambda y$. *Applying backward Euler method*

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

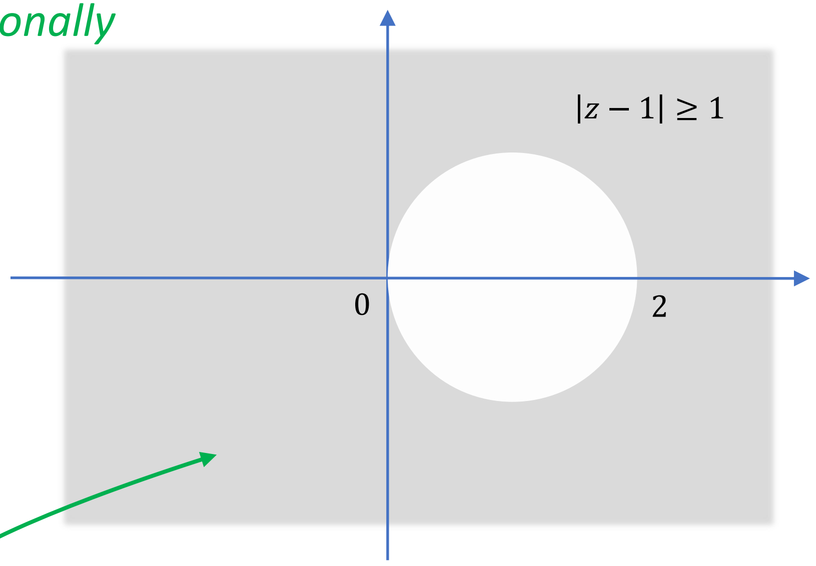
$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$

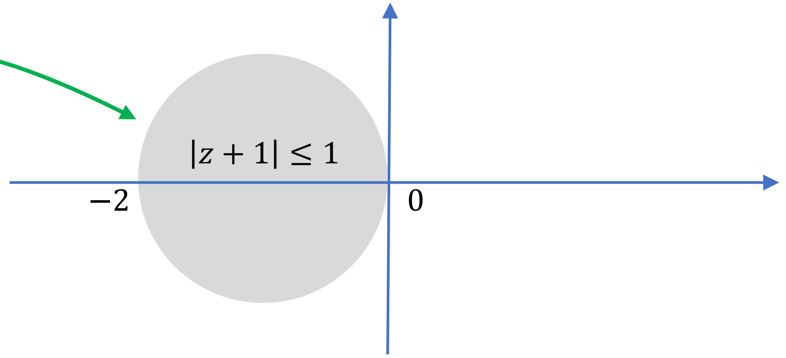
For stability of the backward Euler method, we need

$$\left|\frac{1}{1-h\lambda}\right| \le 1 \quad or \quad |1 - \lambda h| \ge 1.$$

Note that this condition holds for all $h > 0$ *where* $Re(\lambda) < 0$.

Unconditionally stable!





Compare with Euler's method

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$

For stability of the backward Euler method, we need

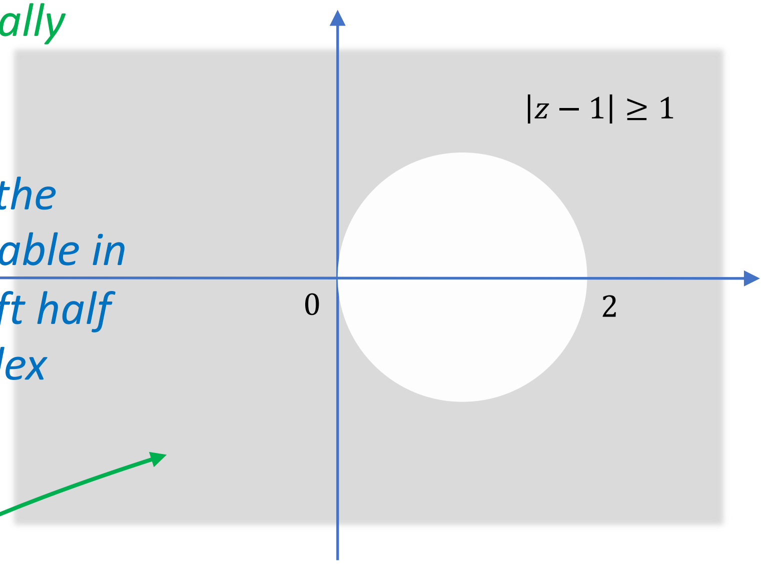
$$\left|\frac{1}{1-h\lambda}\right| \leq 1 \quad \textit{or} \quad |1 - \lambda h| \geq 1.$$

Note that this condition holds for all $h > 0$ where $Re(\lambda) < 0$.

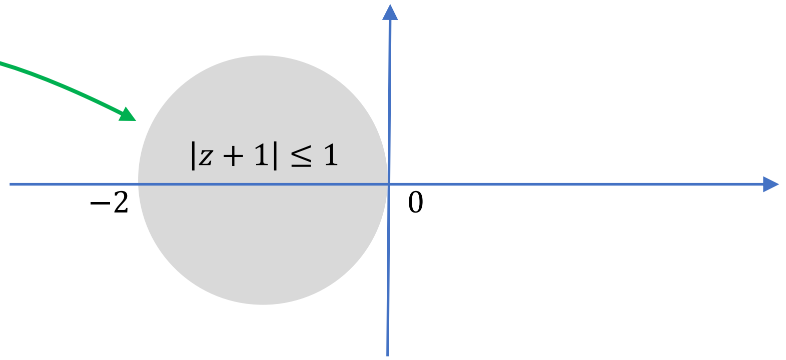
Unconditionally stable!

…

means that the method is stable in the entire left half of the complex plane.



Compare with Euler's method



To understand the stability, let us again consider the ODE $y' = \lambda y$. Applying backward Euler method

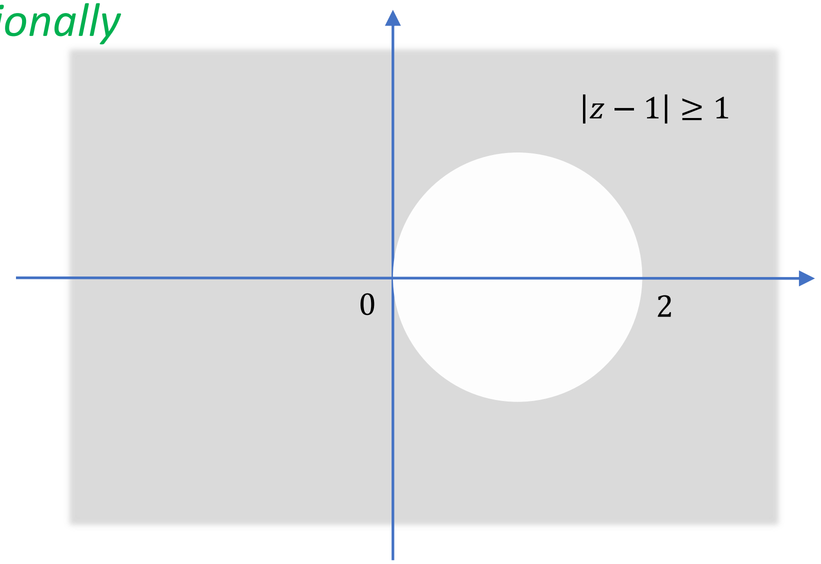
$$y_{k+1} = y_k + h(\lambda y_{k+1})$$

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

$$y_{k+1} = \left(\frac{1}{1-h\lambda}\right) y_k$$

$$y_k = \left(\frac{1}{1-h\lambda}\right)^k y_0$$

Unconditionally stable!



For stability of the backward Euler method, we need

$$\left|\frac{1}{1-h\lambda}\right| \leq 1 \quad \text{or} \quad |1 - \lambda h| \geq 1.$$

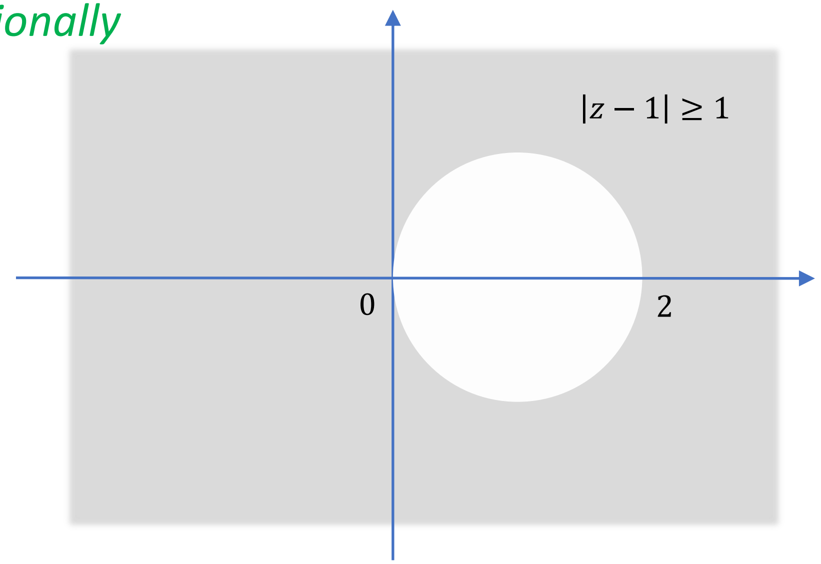
Note that this condition holds for all $h > 0$ where $Re(\lambda) < 0$.

For the general ODE $y' = f(t, y)$, we have

For the general ODE $y' = f(t, y)$, we have

$$e_{k+1} = y_{k+1} - y(t_{k+1})$$

Unconditionally stable!



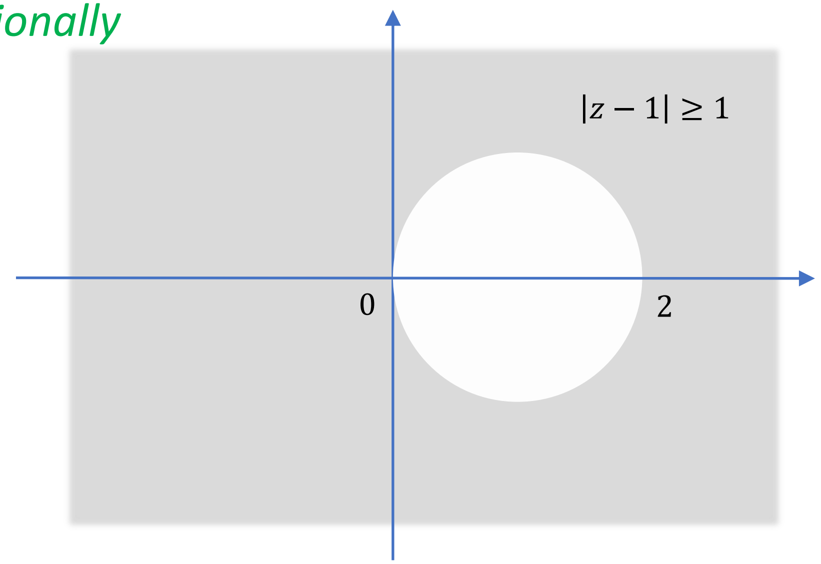
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$e_{k+1} = y_{k+1} - y(t_{k+1})$$
$$= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1})$$

Unconditionally stable!



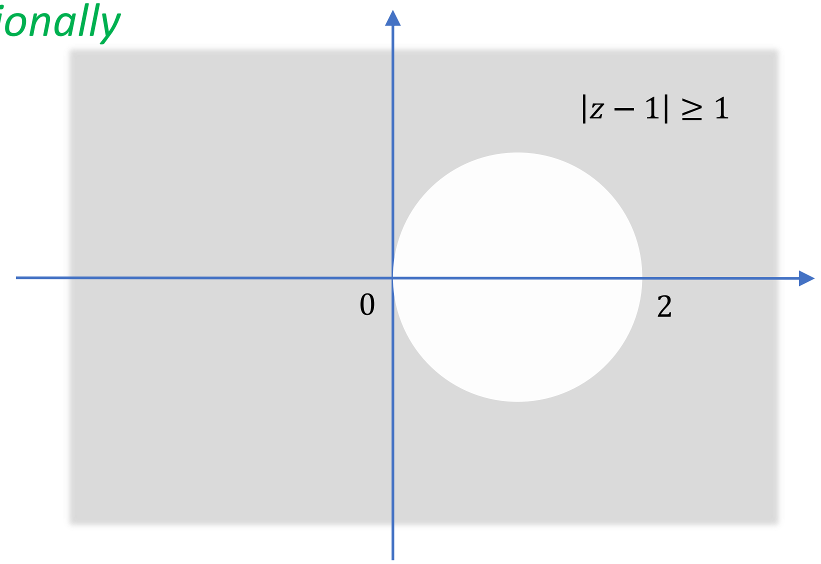
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$\begin{aligned} e_{k+1} &= y_{k+1} - y(t_{k+1}) \\ &= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\ &= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\ &+ [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \end{aligned}$$

Unconditionally stable!



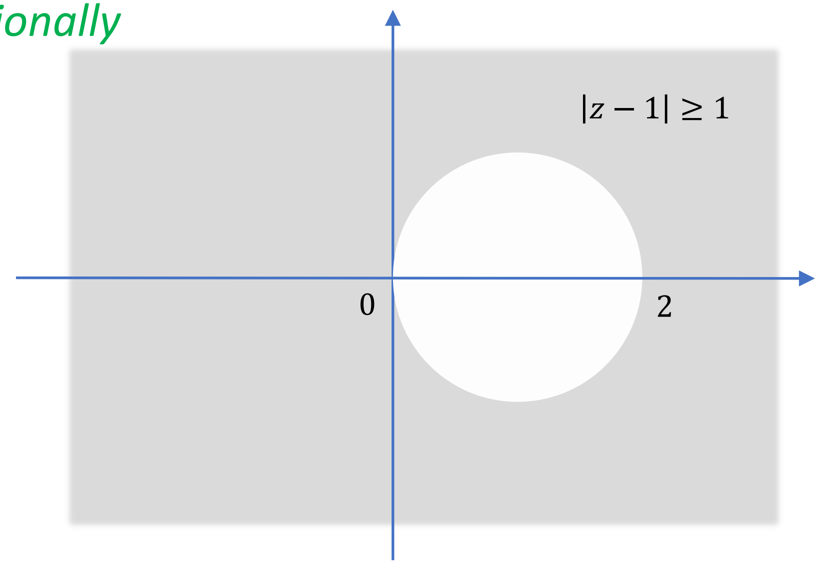
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$
\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&+ [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1}
\end{aligned}
$$

Unconditionally stable!



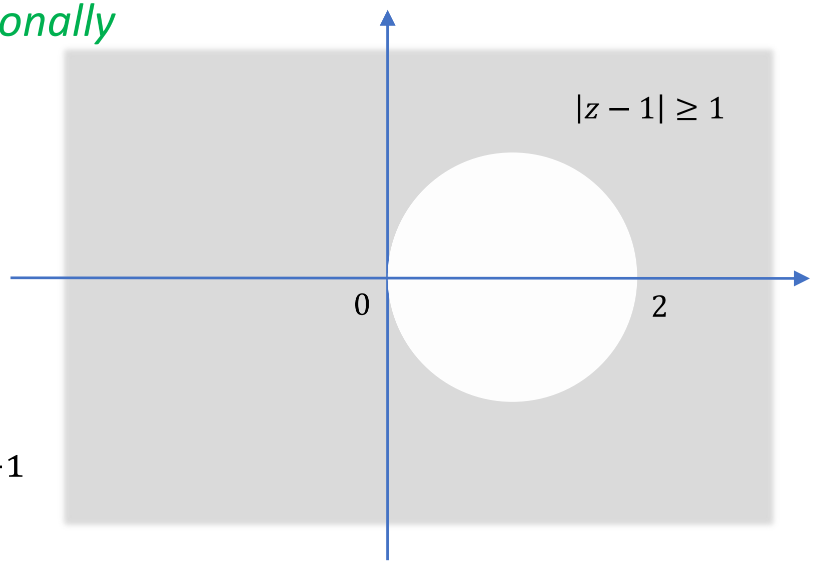
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t,y)$*, we have*

$$
\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&+ [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1}
\end{aligned}
$$

Unconditionally stable!



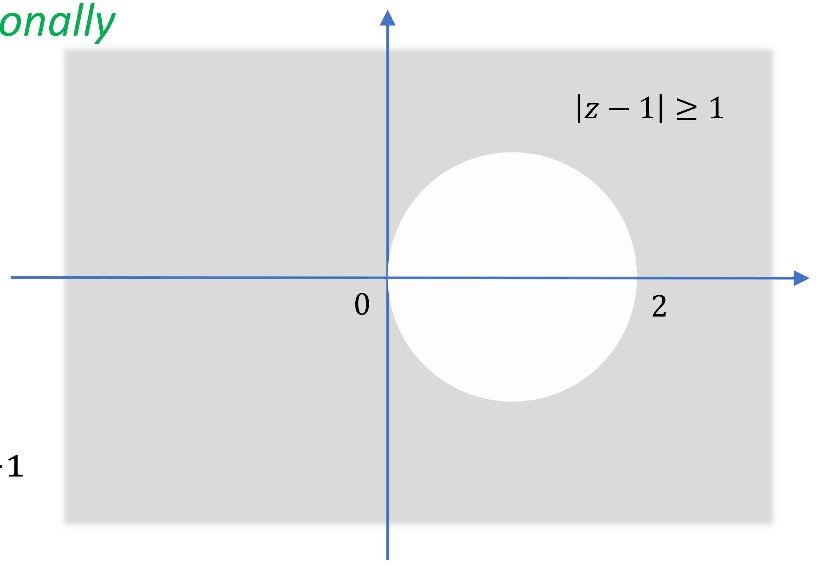
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$
\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))]e_{k+1} + \ell_{k+1}
\end{aligned}
$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f')e_{k+1} = e_k + \ell_{k+1}$$

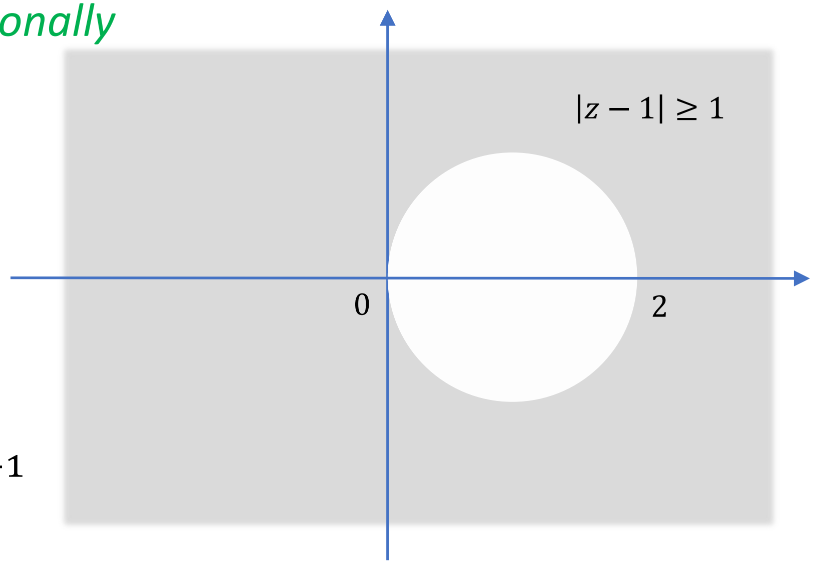
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1}
\end{aligned}$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f') e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

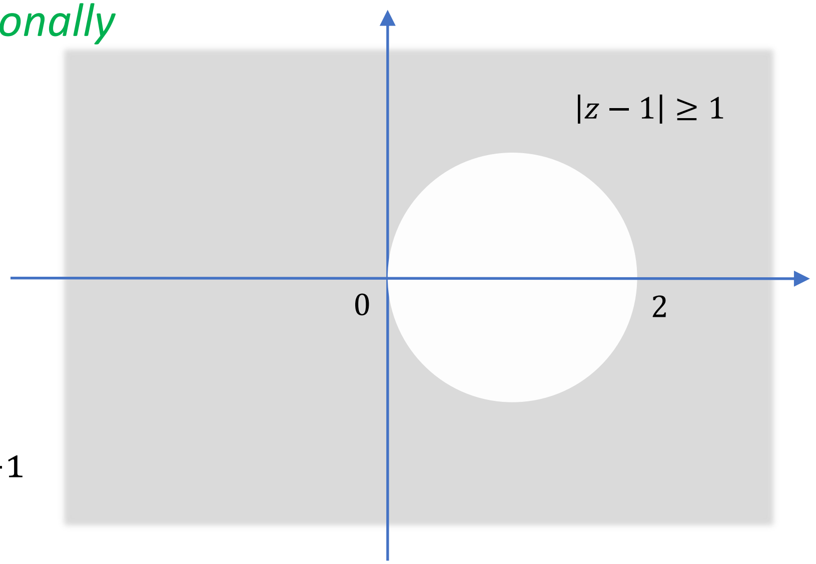
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$\begin{aligned} e_{k+1} &= y_{k+1} - y(t_{k+1}) \\ &= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\ &= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\ &\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\ &= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\ &= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1} \end{aligned}$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f') e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

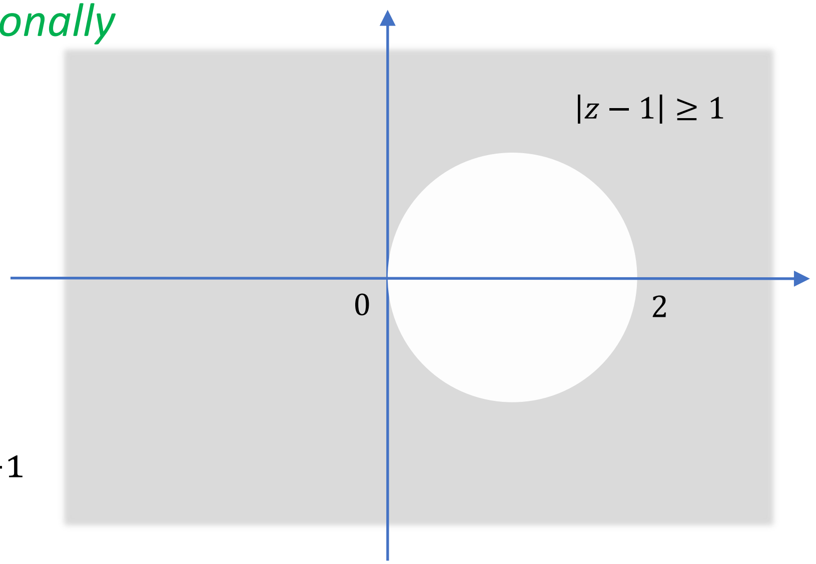
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$e_{k+1} = y_{k+1} - y(t_{k+1})$$
$$= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1})$$
$$= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big)$$
$$+ [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})]$$
$$= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1}$$
$$= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))]e_{k+1} + \ell_{k+1}$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f')e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

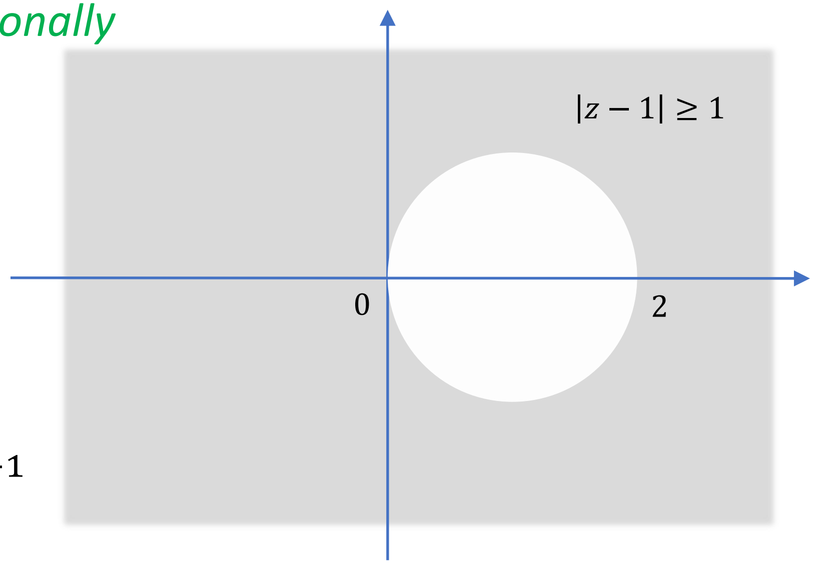
Is backward Euler method more accurate?

Initial Value Problems: Implicit Methods

For the general ODE $y' = f(t, y)$, we have

$$
\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1}
\end{aligned}
$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f') e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

Is backward Euler method more accurate?

$$\ell_{k+1} = y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})$$

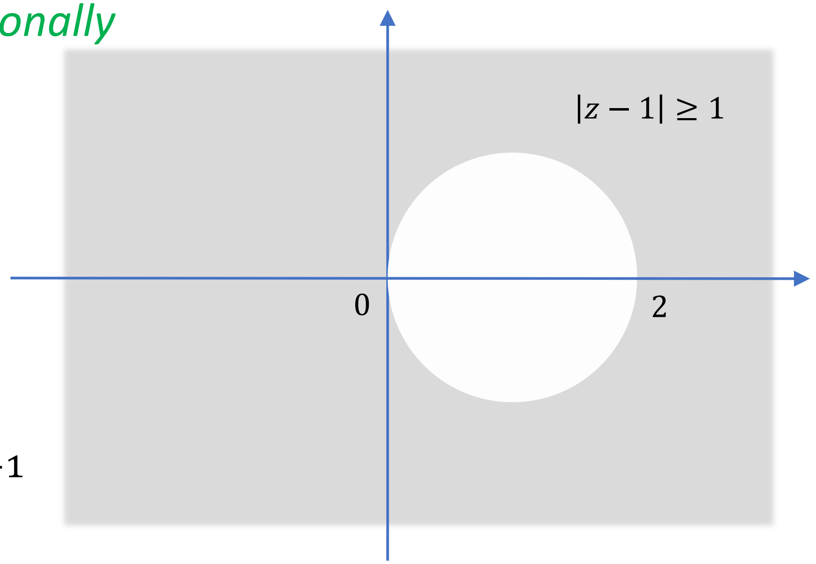
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$\begin{aligned} e_{k+1} &= y_{k+1} - y(t_{k+1}) \\ &= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\ &= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\ &\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\ &= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\ &= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1} \end{aligned}$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f') e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

Is backward Euler method more accurate?

$$\ell_{k+1} = y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1}) = h_k f(t_{k+1}, y(t_{k+1})) + [y(t_k) - y(t_{k+1})]$$

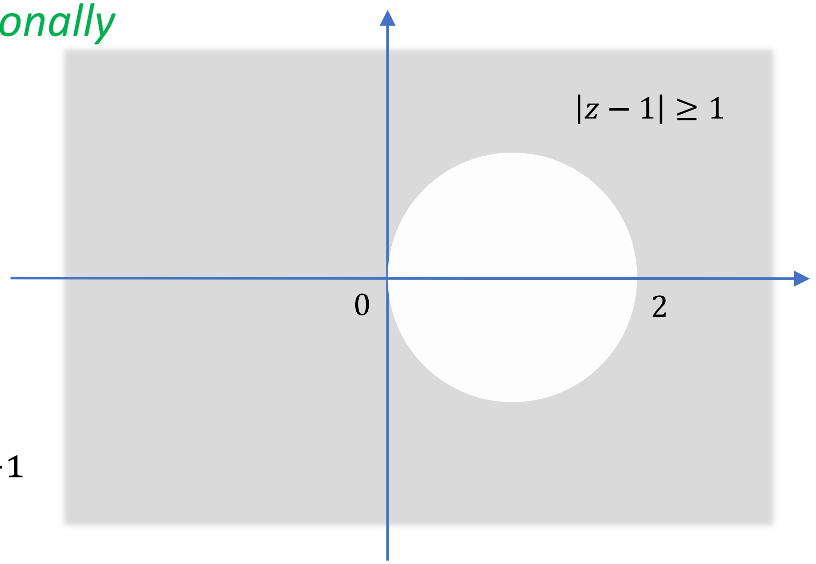
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$
\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1}
\end{aligned}
$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f') e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

Is backward Euler method more accurate?

$$
\begin{aligned}
\ell_{k+1} &= y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1}) = h_k f(t_{k+1}, y(t_{k+1})) + [y(t_k) - y(t_{k+1})] \\
&= h_k f(t_{k+1}, y(t_{k+1})) + [(t_k - t_{k+1}) y'(t_{k+1}) + O\big((t_k - t_{k+1})^2\big)]
\end{aligned}
$$

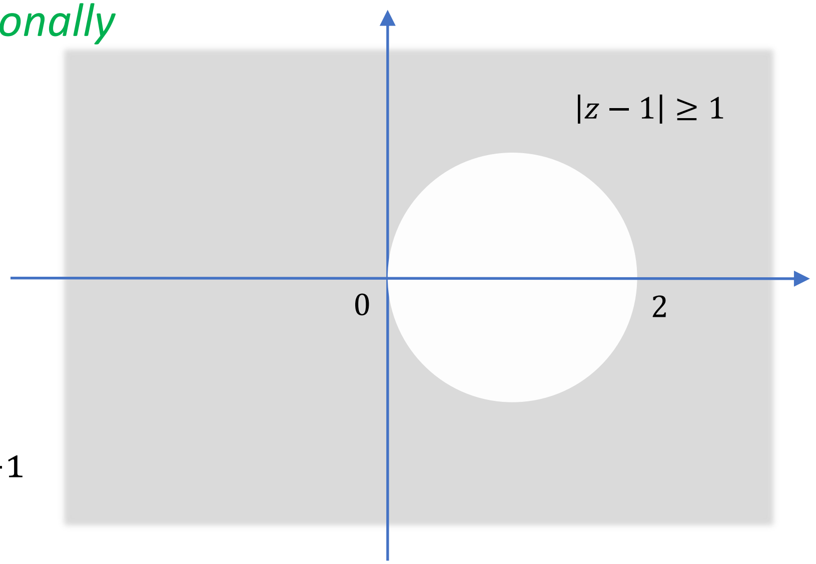
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$
\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&+ [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))]e_{k+1} + \ell_{k+1}
\end{aligned}
$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f')e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

Is backward Euler method more accurate?

$$
\begin{aligned}
\ell_{k+1} &= y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1}) = h_k f(t_{k+1}, y(t_{k+1})) + [y(t_k) - y(t_{k+1})] \\
&= h_k f(t_{k+1}, y(t_{k+1})) + [(t_k - t_{k+1}) y'(t_{k+1}) + O\big((t_k - t_{k+1})^2\big)] \\
&= h_k f(t_{k+1}, y(t_{k+1})) + \big[-h_k f(t_{k+1}, y(t_{k+1})) + O\big(h_k^2\big)\big]
\end{aligned}
$$

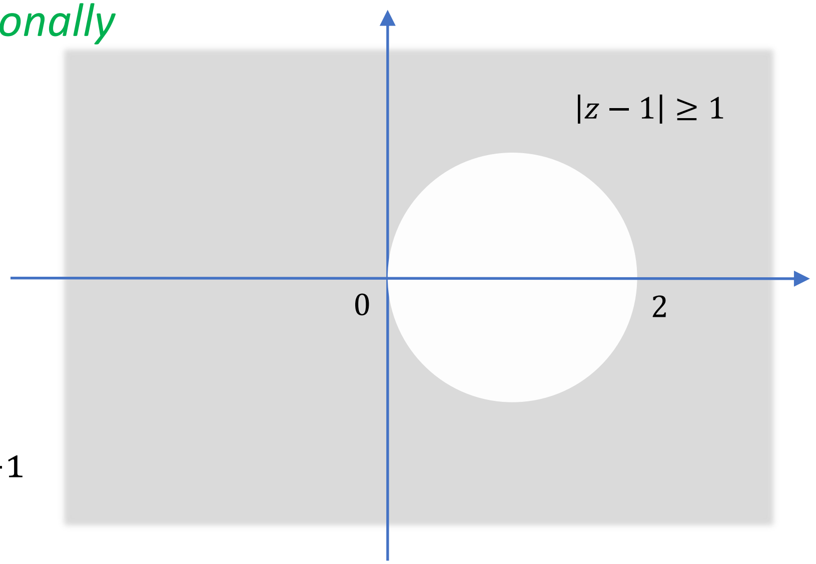
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1}
\end{aligned}$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f') e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

Is backward Euler method more accurate?

$$\begin{aligned}
\ell_{k+1} &= y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1}) = h_k f(t_{k+1}, y(t_{k+1})) + [y(t_k) - y(t_{k+1})] \\
&= h_k f(t_{k+1}, y(t_{k+1})) + \big[(t_k - t_{k+1}) y'(t_{k+1}) + O\big((t_k - t_{k+1})^2\big)\big] \\
&= h_k f(t_{k+1}, y(t_{k+1})) + \big[-h_k f(t_{k+1}, y(t_{k+1})) + O\big(h_k^2\big)\big] = O\big(h_k^2\big)
\end{aligned}$$

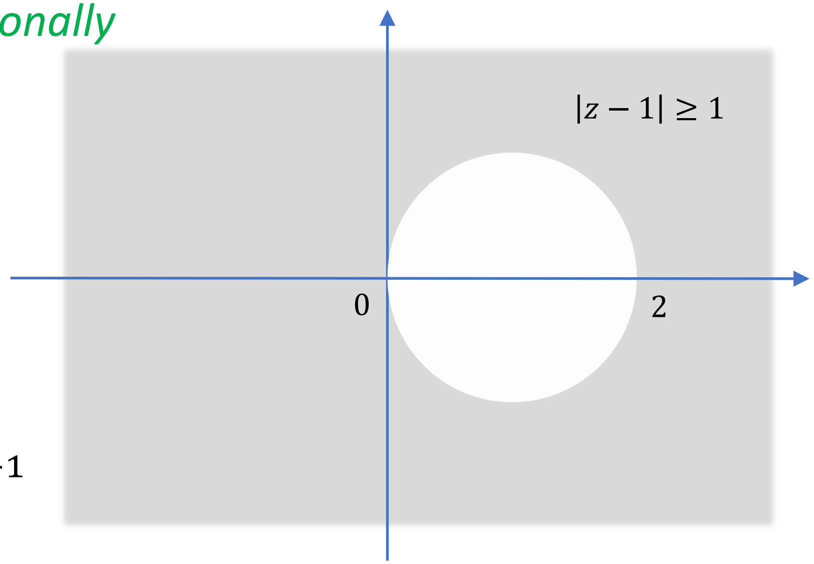
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

For the general ODE $y' = f(t, y)$, we have

$$
\begin{aligned}
e_{k+1} &= y_{k+1} - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_{k+1}) \\
&= y_k + h_k f(t_{k+1}, y_{k+1}) - y(t_k) - h_k f\big(t_{k+1}, y(t_{k+1})\big) \\
&\quad + [y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1})] \\
&= e_k + h_k\big[f(t_{k+1}, y_{k+1}) - f\big(t_{k+1}, y(t_{k+1})\big)\big] + \ell_{k+1} \\
&= e_k + h_k[f'(t_{k+1}, \alpha y_{k+1} + (1-\alpha) y(t_{k+1}))] e_{k+1} + \ell_{k+1}
\end{aligned}
$$

Unconditionally stable!



We, therefore, have

$$(I - h_k f') e_{k+1} = e_k + \ell_{k+1} \quad \text{or} \quad e_{k+1} = (I - h_k f')^{-1} e_k + (I - h_k f')^{-1} \ell_{k+1}$$

For stability, we need $\rho(I - h_k f')^{-1} \leq 1$, i.e., eigenvalues of $h_k f'$ must lie outside the unit circle centered at 1.

Is backward Euler method more accurate?

$$
\begin{aligned}
\ell_{k+1} &= y(t_k) + h_k f(t_{k+1}, y(t_{k+1})) - y(t_{k+1}) = h_k f(t_{k+1}, y(t_{k+1})) + [y(t_k) - y(t_{k+1})] \\
&= h_k f(t_{k+1}, y(t_{k+1})) + \big[(t_k - t_{k+1}) y'(t_{k+1}) + O\big((t_k - t_{k+1})^2\big)\big] \\
&= h_k f(t_{k+1}, y(t_{k+1})) + \big[-h_k f(t_{k+1}, y(t_{k+1})) + O\big(h_k^2\big)\big] = O\big(h_k^2\big)
\end{aligned}
$$

first-order accurate!

Lesson 2

Initial Value Problems

2.2 Stability

2.3 Euler's method

2.4 Implicit method

- Trapezoidal method

Akash Anand
MATH, IIT KANPUR

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

How do we obtain a higher-order method?

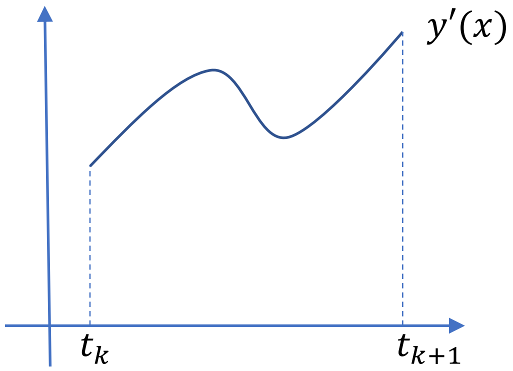
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

How do we obtain a higher-order method?

So far, to solve $y' = f(t, y), y(t_0) = y_0$, *we have used*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$



Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

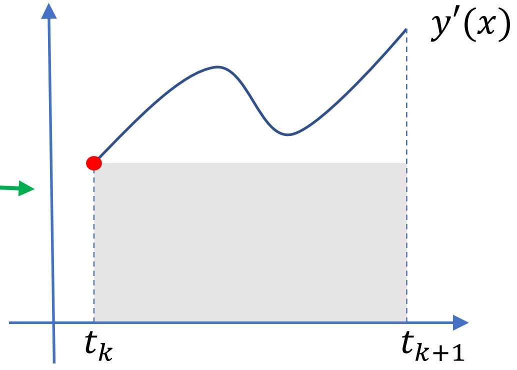
How do we obtain a higher-order method?

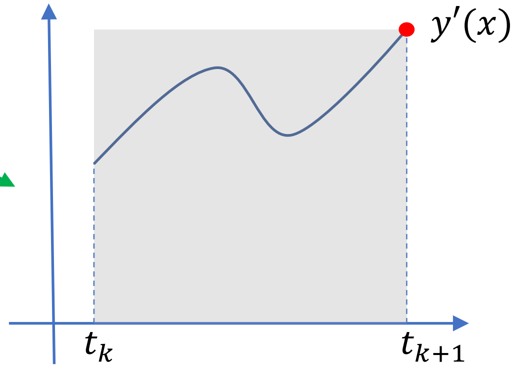
So far, to solve $y' = f(t, y), y(t_0) = y_0$, we have used

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$





Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

How do we obtain a higher-order method?

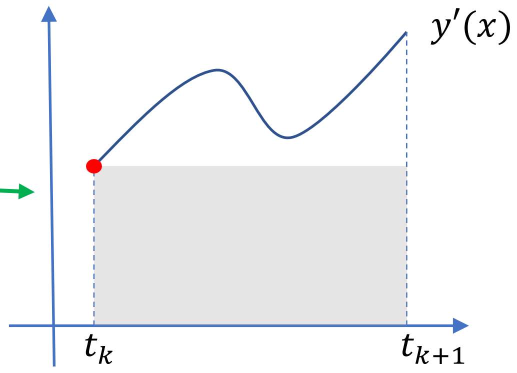
So far, to solve $y' = f(t,y), y(t_0) = y_0$, *we have used*

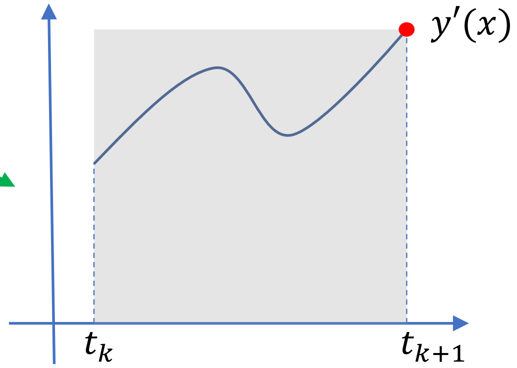
$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$

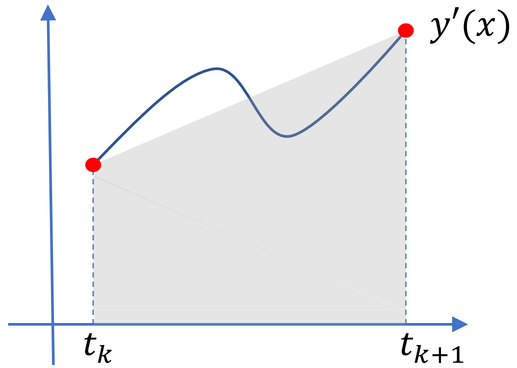
$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$

Approximating the integral by trapezoidal rule, we get







Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

How do we obtain a higher-order method?

So far, to solve $y' = f(t,y), y(t_0) = y_0$, *we have used*

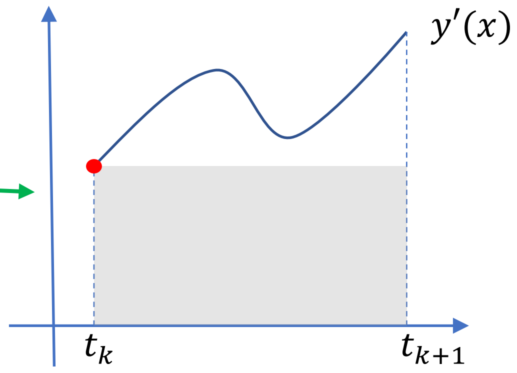
$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$

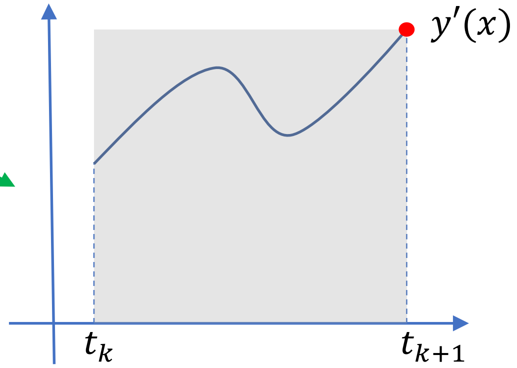
$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

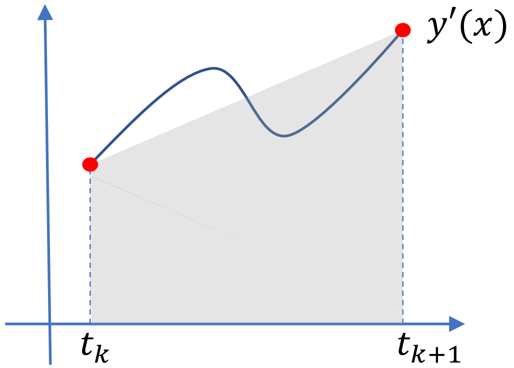
$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$







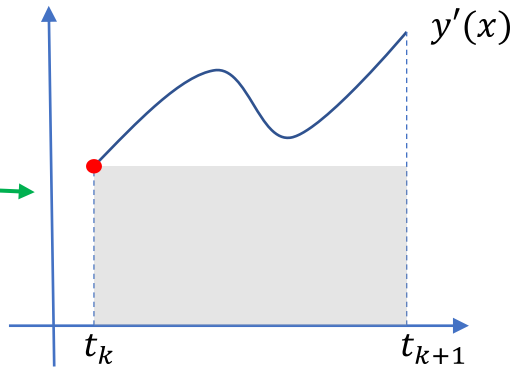
Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

How do we obtain a higher-order method?

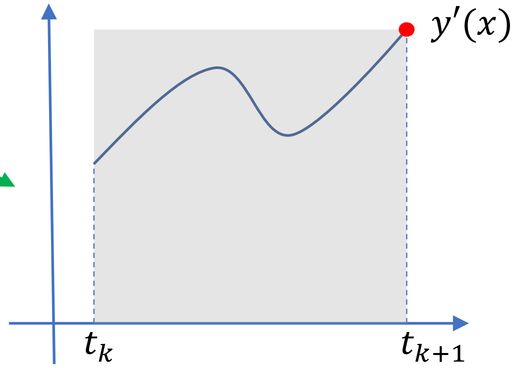
So far, to solve $y' = f(t,y), y(t_0) = y_0$, *we have used*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$



$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

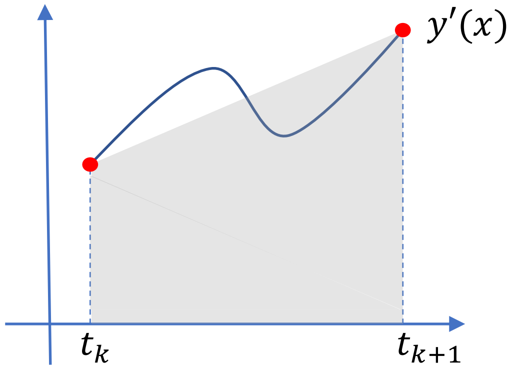
$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$



Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$

Trapezoidal method



Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

How do we obtain a higher-order method?

So far, to solve $y' = f(t, y), y(t_0) = y_0$, *we have used*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$

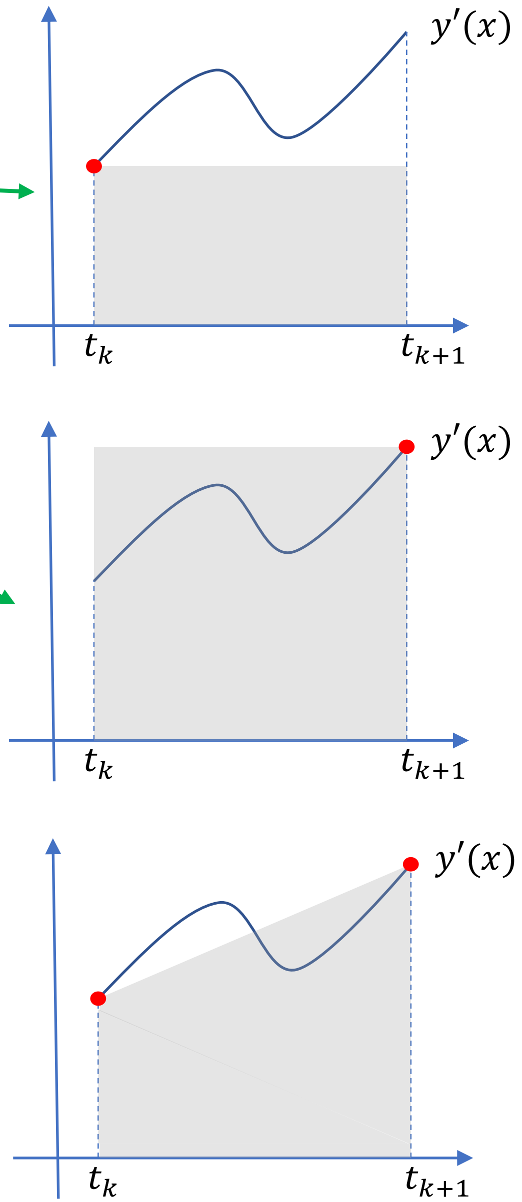
Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$

Trapezoidal method

To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k (y_k + y_{k+1})/2$$



Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

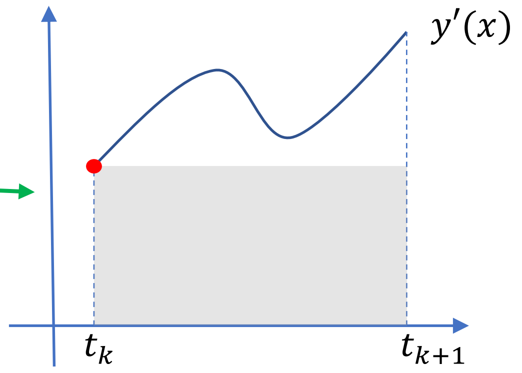
How do we obtain a higher-order method?

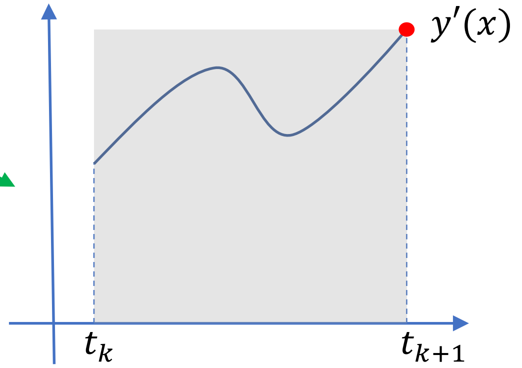
So far, to solve $y' = f(t, y), y(t_0) = y_0$, we have used

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$





Approximating the integral by trapezoidal rule, we get

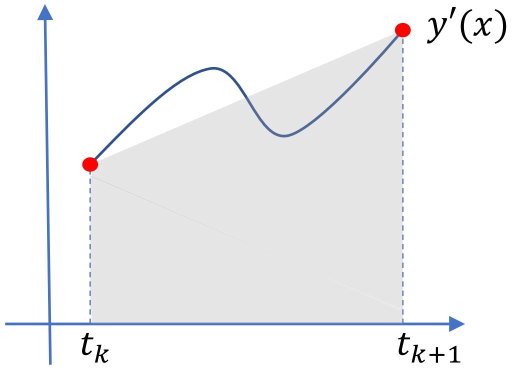
$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$

Trapezoidal method

To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$



Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

How do we obtain a higher-order method?

So far, to solve $y' = f(t,y), y(t_0) = y_0$, we have used

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$

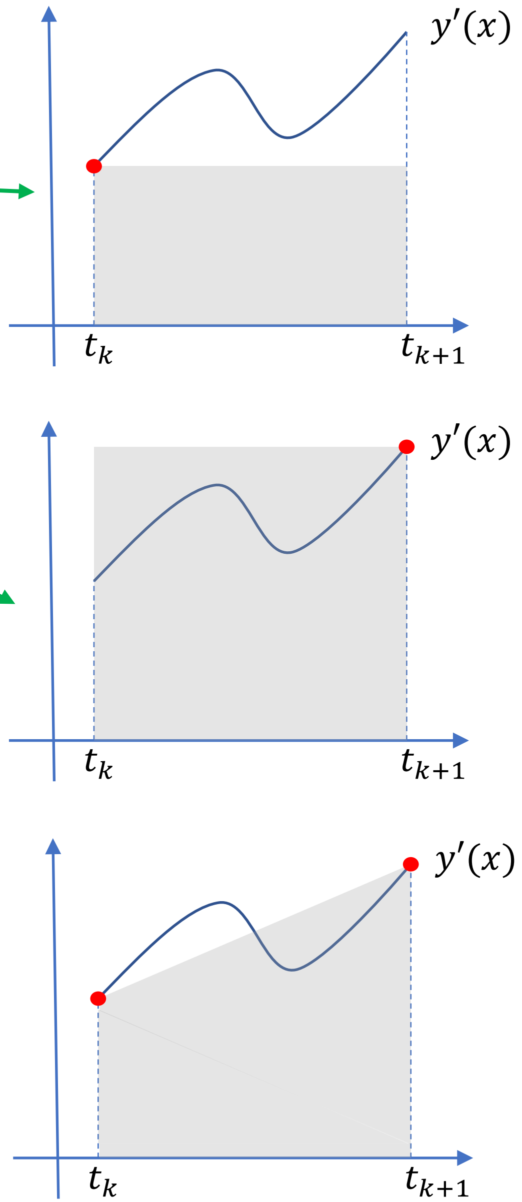
Trapezoidal method

To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$



Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

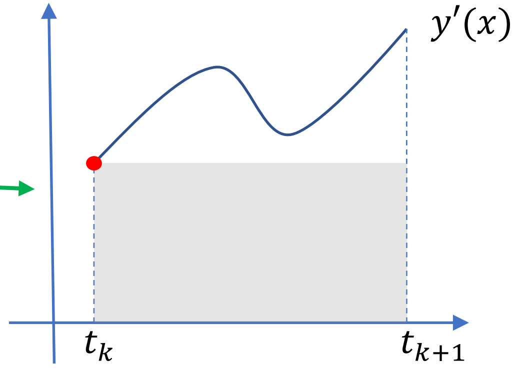
How do we obtain a higher-order method?

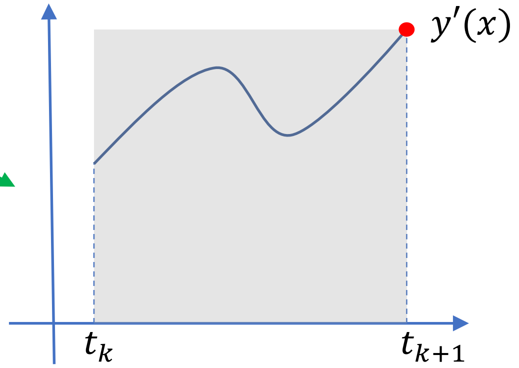
So far, to solve $y' = f(t,y), y(t_0) = y_0$, *we have used*

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} y'(s)ds$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_k, y_k)$$

$$y(t_{k+1}) = y(t_k) + h_k f(t_{k+1}, y_{k+1})$$





Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$

Trapezoidal method

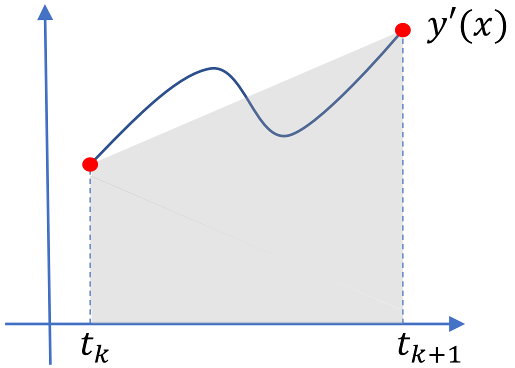
To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k (y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2) y_{k+1} = (1 + \lambda h_k/2) y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

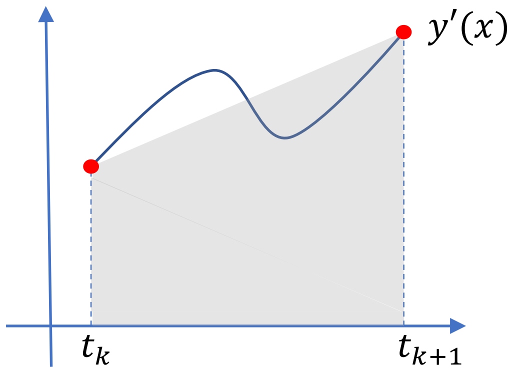


Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

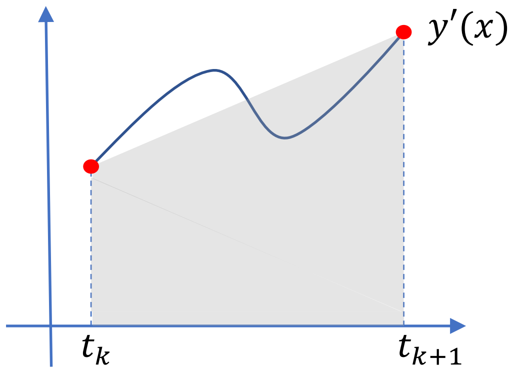
$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

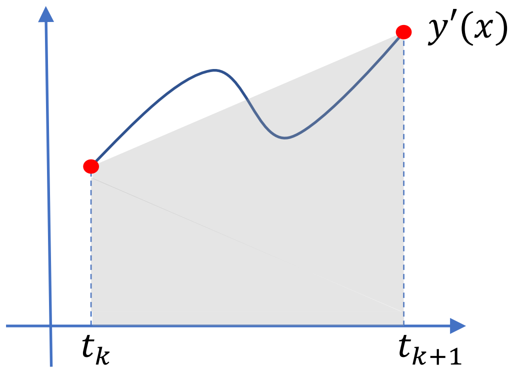
For stability, we need $|(1 + \lambda h_k/2)/(1 - \lambda h_k/2)| \leq 1.$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

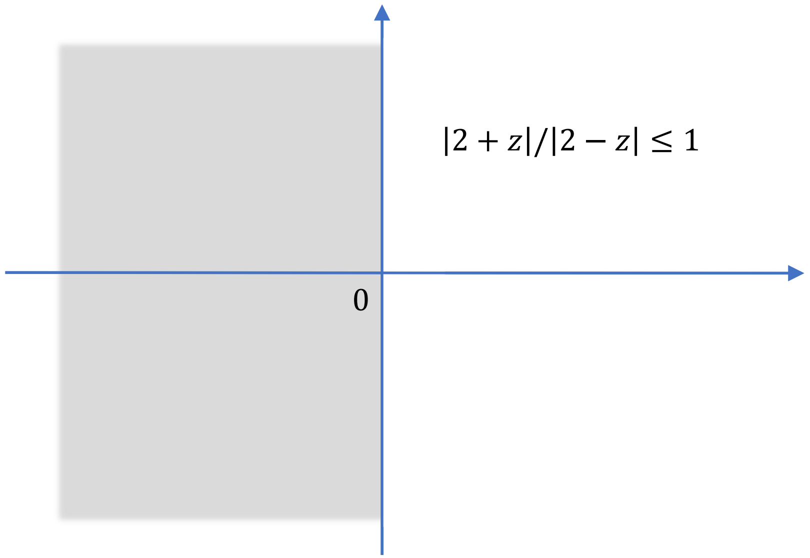
$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

For stability, we need $|(1 + \lambda h_k/2)/(1 - \lambda h_k/2)| \le 1.$

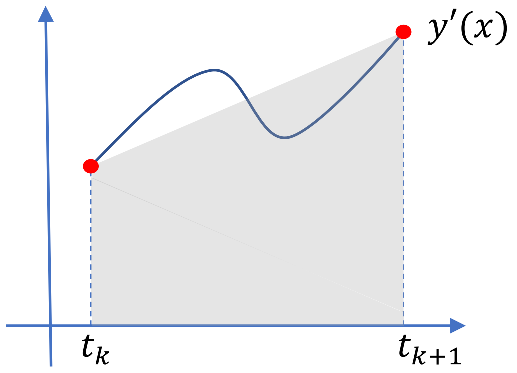


Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

For stability, we need $|(1 + \lambda h_k/2)/(1 - \lambda h_k/2)| \leq 1.$

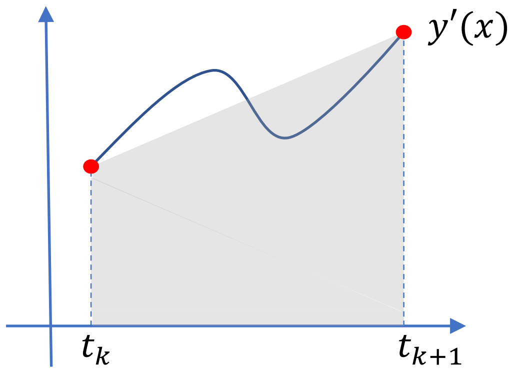
For order of accuracy, we look at the local error

$$\ell_{k+1} = y(t_k) + h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - y(t_{k+1})$$

Initial Value Problems: Implicit Methods

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

For stability, we need $|(1 + \lambda h_k/2)/(1 - \lambda h_k/2)| \leq 1.$

For order of accuracy, we look at the local error

$$\ell_{k+1} = y(t_k) + h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - y(t_{k+1})$$

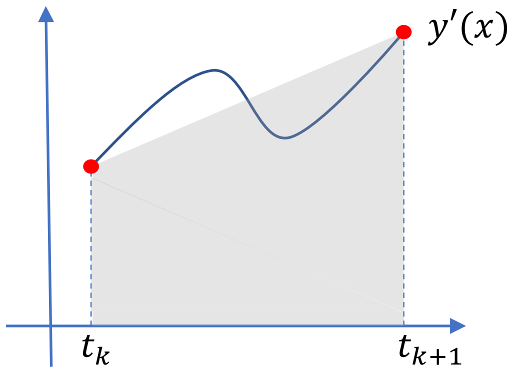
$$= h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - (y(t_{k+1}) - y(t_k))/2 + \big(y(t_k) - y(t_{k+1})\big)/2$$

Initial Value Problems: Implicit Methods

Akash Anand
MATH, IIT KANPUR

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

For stability, we need $|(1 + \lambda h_k/2)/(1 - \lambda h_k/2)| \leq 1.$

For order of accuracy, we look at the local error

$$\ell_{k+1} = y(t_k) + h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - y(t_{k+1})$$

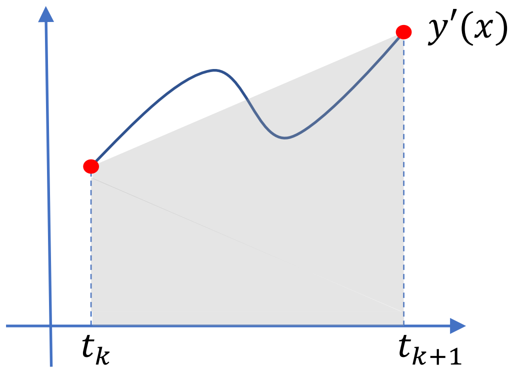
$$= h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - (y(t_{k+1}) - y(t_k))/2 + \big(y(t_k) - y(t_{k+1})\big)/2$$

$$= h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - \left(h_k y'(t_k) + h_k^2 y''(t_k)/2 + O(h_k^3)\right)/2$$

$$+ \left(-h_k y'(t_{k+1}) + h_k^2 y''(t_{k+1})/2 + O(h_k^3)\right)/2$$

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k(y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2)y_{k+1} = (1 + \lambda h_k/2)y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

For stability, we need $|(1 + \lambda h_k/2)/(1 - \lambda h_k/2)| \leq 1.$

For order of accuracy, we look at the local error

$$\ell_{k+1} = y(t_k) + h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - y(t_{k+1})$$

$$= h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - (y(t_{k+1}) - y(t_k))/2 + \big(y(t_k) - y(t_{k+1})\big)/2$$

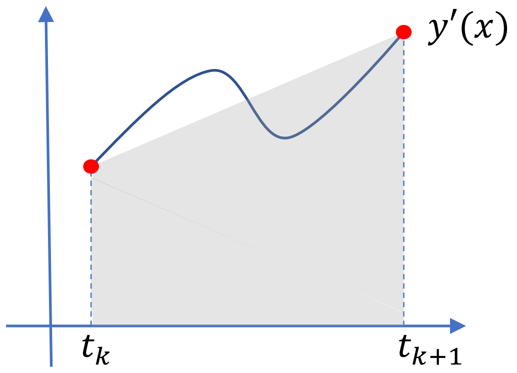
$$= h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - \left(h_k y'(t_k) + h_k^2 y''(t_k)/2 + O\big(h_k^3\big)\right)/2$$

$$+ \left(-h_k y'(t_{k+1}) + h_k^2 y''(t_{k+1})/2 + O\big(h_k^3\big)\right)/2 \quad = \; O\big(h_k^3\big)$$

Initial Value Problems: Implicit Methods

Approximating the integral by trapezoidal rule, we get

$$y(t_{k+1}) = y(t_k) + h_k\big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\big)/2$$



To study the stability, we apply the method to $y' = \lambda y$:

$$y_{k+1} = y_k + \lambda h_k (y_k + y_{k+1})/2$$

$$(1 - \lambda h_k/2) y_{k+1} = (1 + \lambda h_k/2) y_k$$

$$y_{k+1} = y_k\big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)$$

$$y_k = \big((1 + \lambda h_k/2)/(1 - \lambda h_k/2)\big)^k y_0$$

For stability, we need $|(1 + \lambda h_k/2)/(1 - \lambda h_k/2)| \le 1.$

For order of accuracy, we look at the local error

$$\ell_{k+1} = y(t_k) + h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - y(t_{k+1})$$

$$= h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - (y(t_{k+1}) - y(t_k))/2 + \big(y(t_k) - y(t_{k+1})\big)/2$$

$$= h_k\big(f(t_k, y(t_k)) + f(t_{k+1}, y(t_{k+1}))\big)/2 - \left(h_k y'(t_k) + h_k^2 y''(t_k)/2 + O(h_k^3)\right)/2$$

$$+ \left(-h_k y'(t_{k+1}) + h_k^2 y''(t_{k+1})/2 + O(h_k^3)\right)/2 \quad = \quad O(h_k^3)$$

second-order accurate!

Module 2

Initial Value Problems

2.3 Euler's method

2.4 Implicit method

2.5 Stiffness

Akash Anand
MATH, IIT KANPUR

Example

Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, where

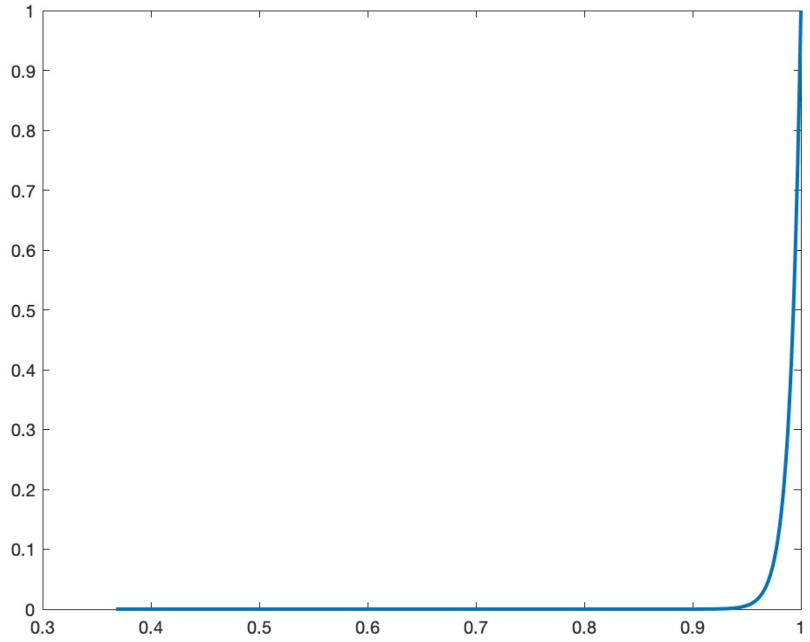
$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Example

Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Exact solution

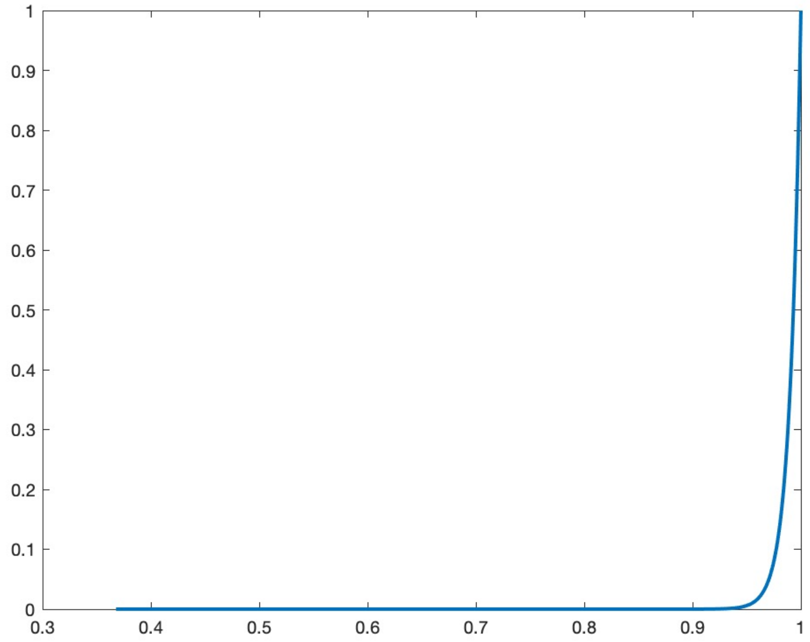


Example

Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Exact solution



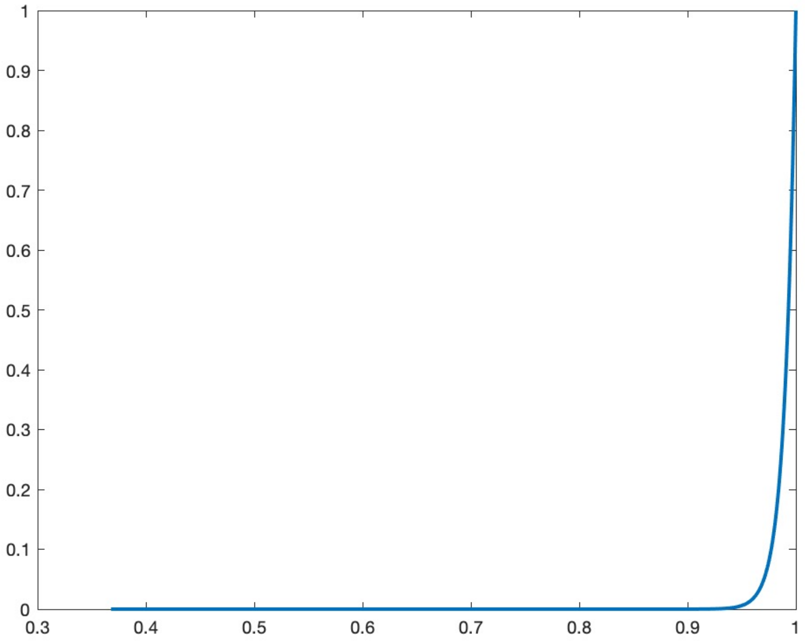
Euler's method with $h = 0.04$

Example

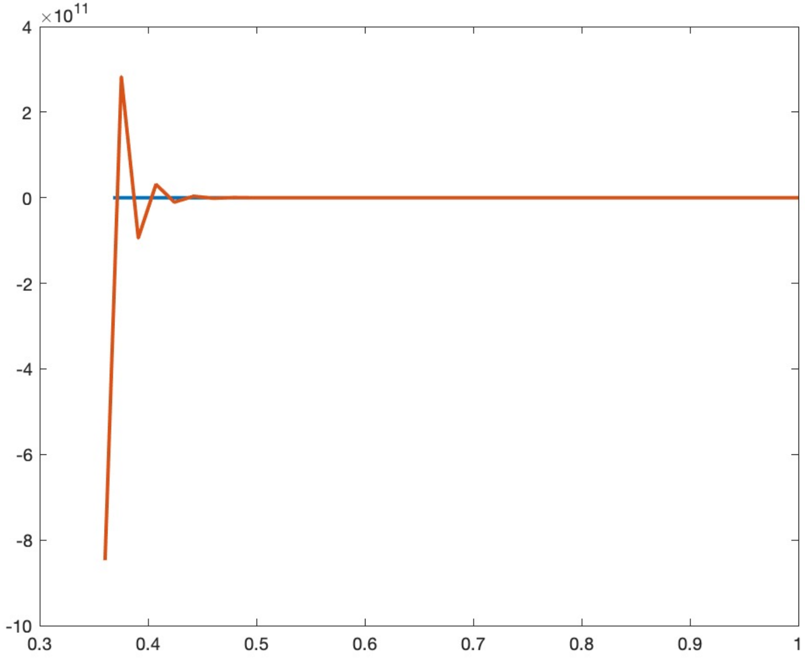
Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \text{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Exact solution



Euler's method with $h = 0.04$



Initial Value Problems: Stiffness

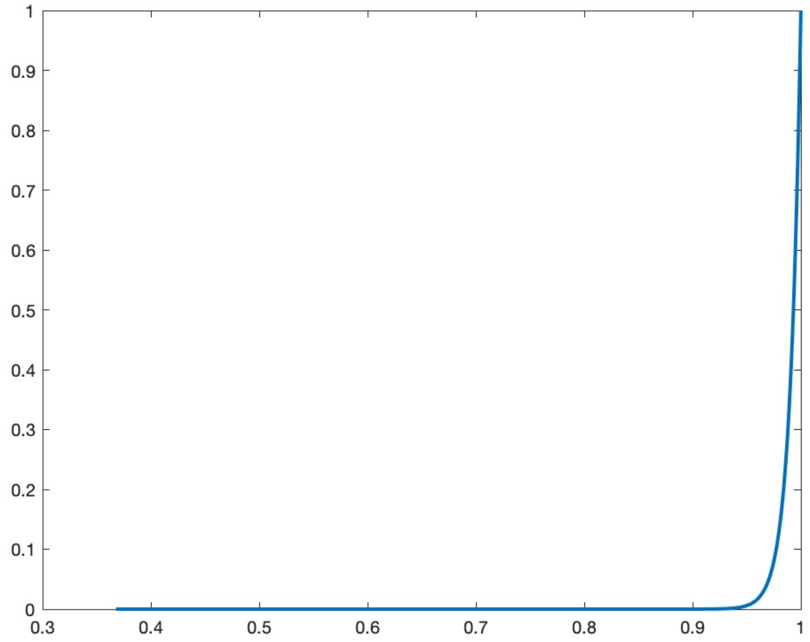
Akash Anand
MATH, IIT KANPUR

Example

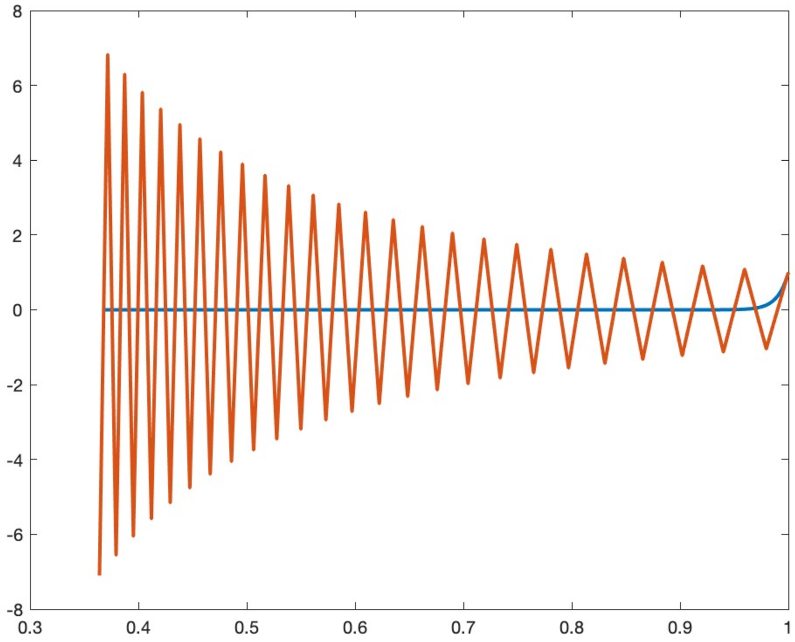
Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Exact solution



Euler's method with $h = 0.0204$

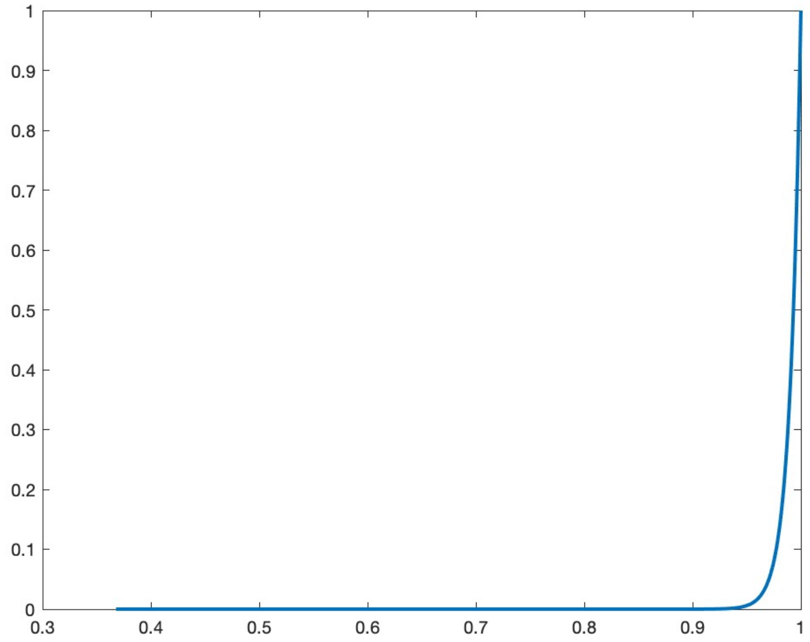


Example

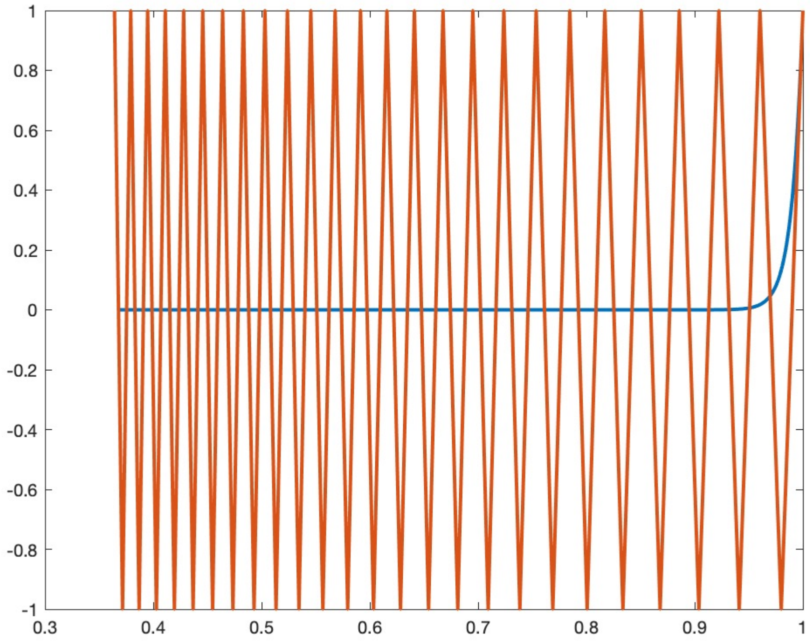
Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Exact solution



Euler's method with $h = 0.02$

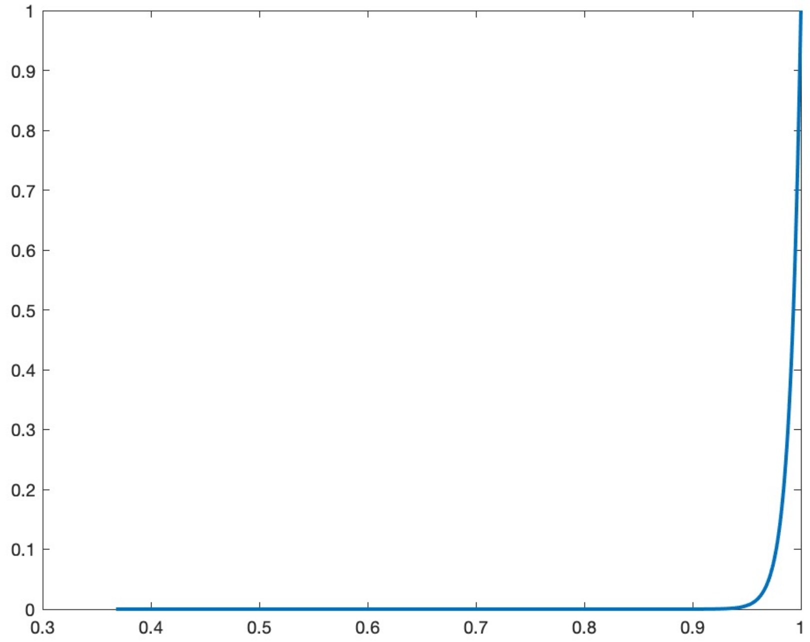


Example

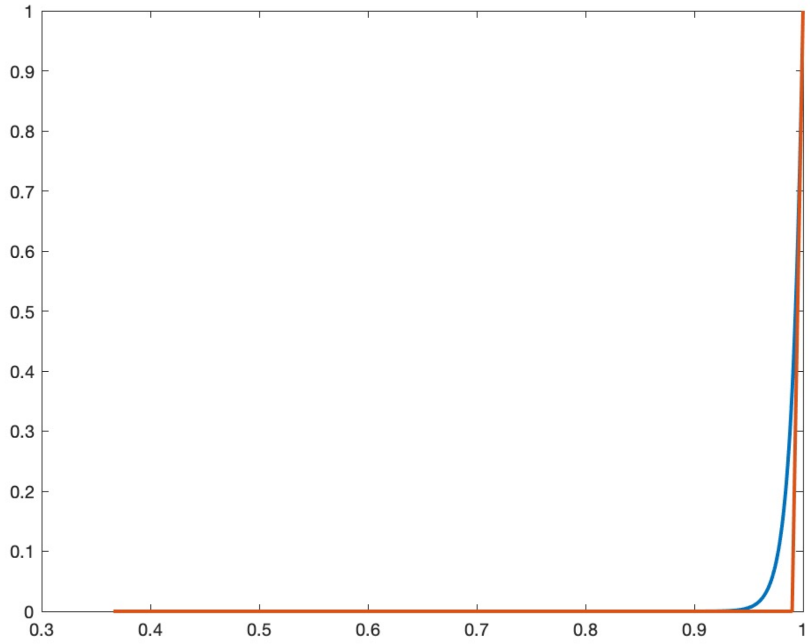
Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Exact solution



Euler's method with $h = 0.01$



Initial Value Problems: Stiffness

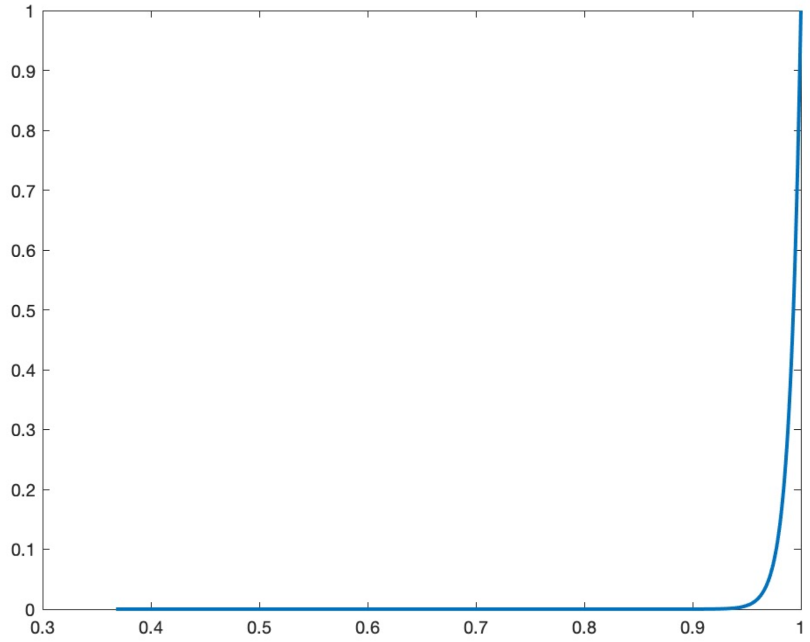
Akash Anand
MATH, IIT KANPUR

Example

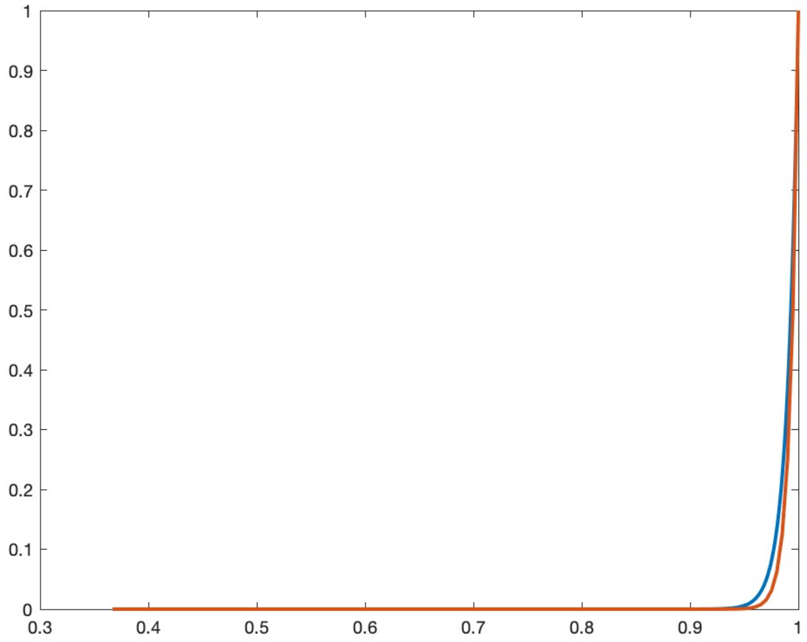
Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Exact solution



Euler's method with $h = 0.005$



Example

Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$*, where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \qquad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \qquad \textit{and} \qquad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

How do we explain this?

Example

Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

How do we explain this?

$$f'(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}$$

Example

Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

How do we explain this?

$$f'(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}, \ I + hf' = \begin{bmatrix} 1 - h & 0 \\ 0 & 1 - 100h \end{bmatrix}$$

Initial Value Problems: Stiffness

Example

Consider the following IVP, $y' = f(t, y),\; y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

How do we explain this?

$$f'(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}, \;\; I + hf' = \begin{bmatrix} 1-h & 0 \\ 0 & 1-100h \end{bmatrix}$$

$$\rho(I + hf') = \max\left(|1-h|, |1-100h|\right)$$

Initial Value Problems: Stiffness

Example

Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

How do we explain this?

$$f'(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}, \ I + hf' = \begin{bmatrix} 1-h & 0 \\ 0 & 1-100h \end{bmatrix}$$

$$\rho(I + hf') = \max\left(|1-h|, |1-100h|\right)$$

For $-1 \le 1 - 100h \le 1$, *we need* $h \le 1/50 = 0.02$.

Example

Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

How do we explain this?

$$f'(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}, \ I + hf' = \begin{bmatrix} 1-h & 0 \\ 0 & 1-100h \end{bmatrix}$$

$$\rho(I + hf') = \max\left(|1-h|, |1-100h|\right)$$

For $-1 \le 1 - 100h \le 1$, *we need* $h \le 1/50 = 0.02$.

Note that, for solving the first equation $y_1' = -y_1$ *only, the Euler's method is stable for all* $h \le 2$.

Example

Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$*, where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

How do we explain this?

$$f'(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix}, \ \ I + hf' = \begin{bmatrix} 1-h & 0 \\ 0 & 1-100h \end{bmatrix}$$

$$\rho(I + hf') = \max\left(|1-h|, |1-100h|\right)$$

For $-1 \le 1 - 100h \le 1$*, we need* $h \le 1/50 = 0.02.$

Note that, for solving the first equation $y_1' = -y_1$ *only, the Euler's method is stable for all* $h \le 2.$

The more severe restriction $h \le 0.02$ *is primarily due to the second equation* $y_2' = -100y_2$ *which governs the component that varies much more rapidly than the first component* y_1*.*

Stiffness

A stable ODE $y' = f(t, y)$ *is stiff if the Jacobian matrix* f' *has eigenvalues that differ greatly in magnitude.*

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Stiffness

A stable ODE $y' = f(t, y)$ is stiff if the Jacobian matrix f' has eigenvalues that differ greatly in magnitude.

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

Stiffness

A stable ODE $y' = f(t, y)$ is stiff if the Jacobian matrix f' has eigenvalues that differ greatly in magnitude.

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

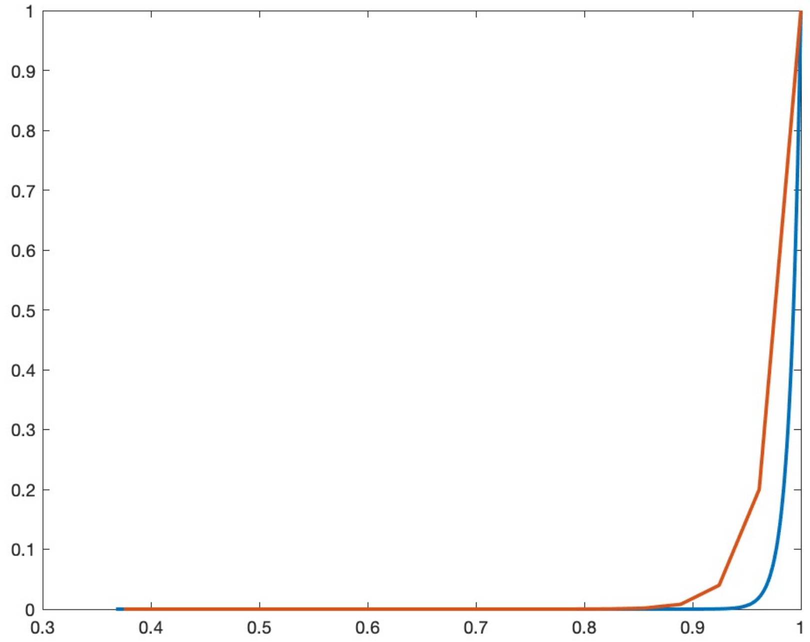
Going back to the previous example and using backward Euler, we see that …

Example

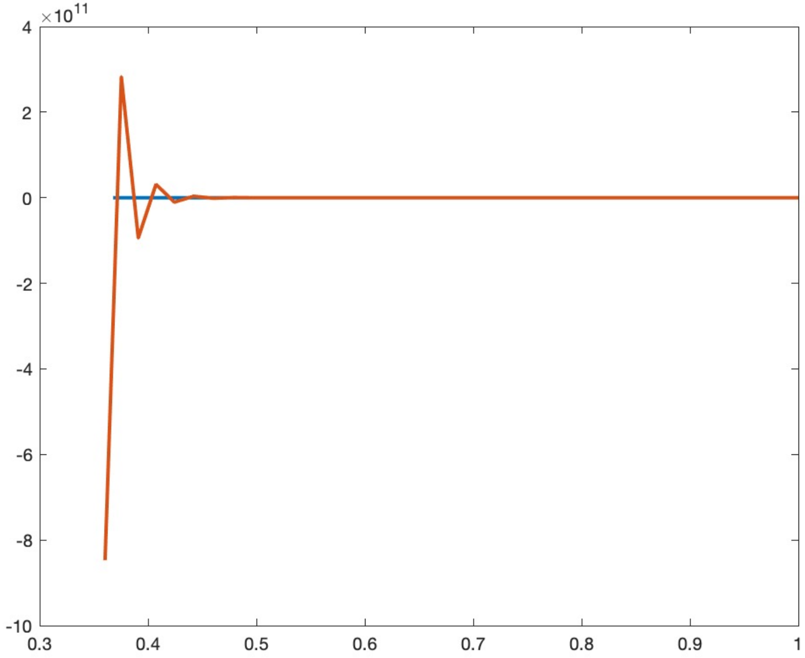
Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Backward Euler with $h = 0.04$



Euler's method with $h = 0.04$



Initial Value Problems: Stiffness

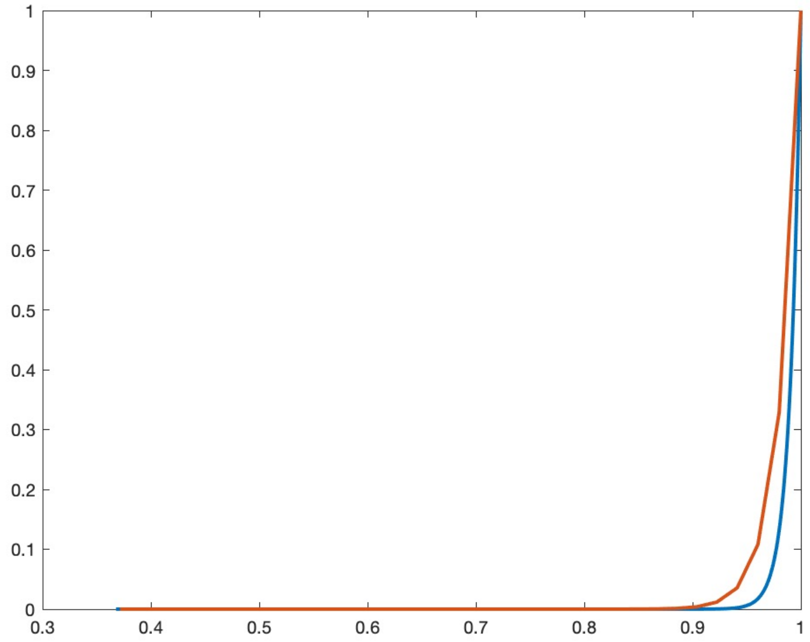
Akash Anand
MATH, IIT KANPUR

Example

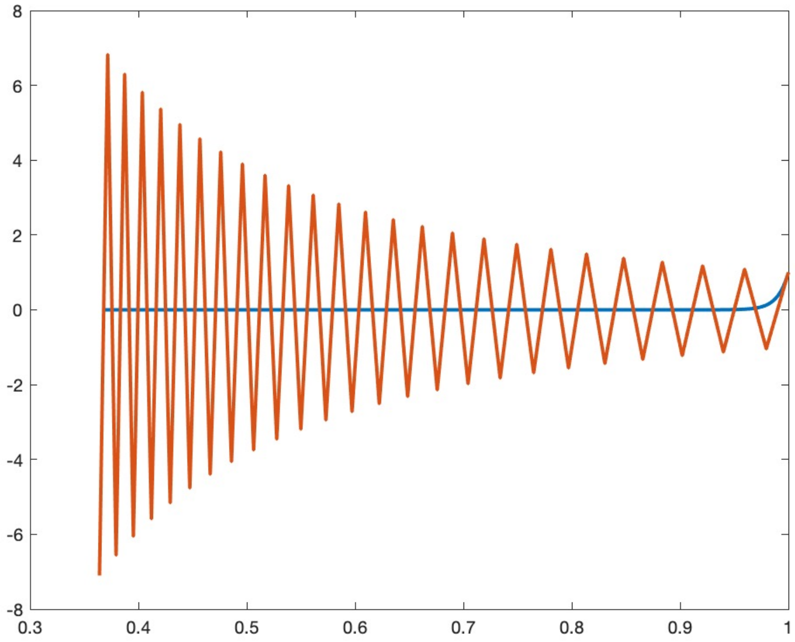
Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Backward Euler with $h = 0.0204$



Euler's method with $h = 0.0204$



Initial Value Problems: Stiffness

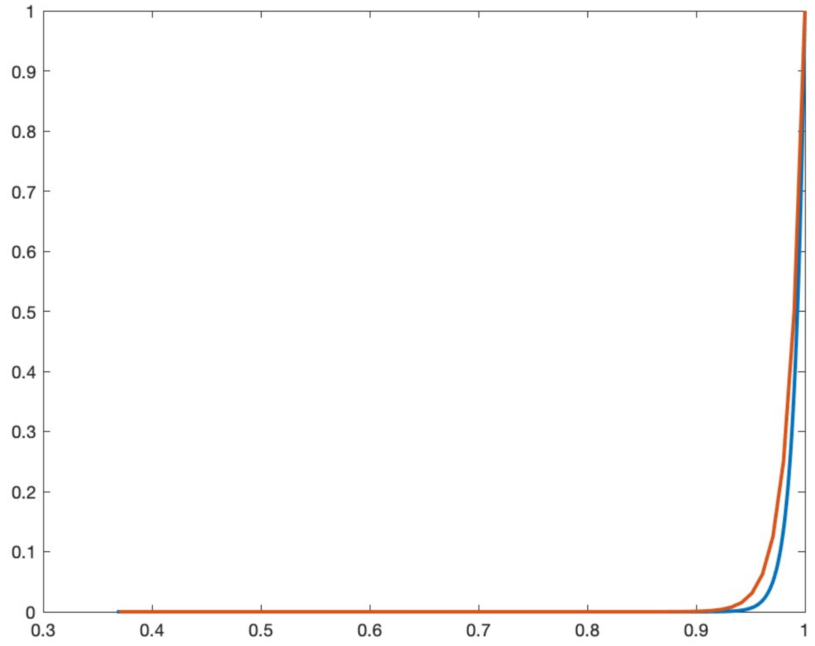
Akash Anand
MATH, IIT KANPUR

Example

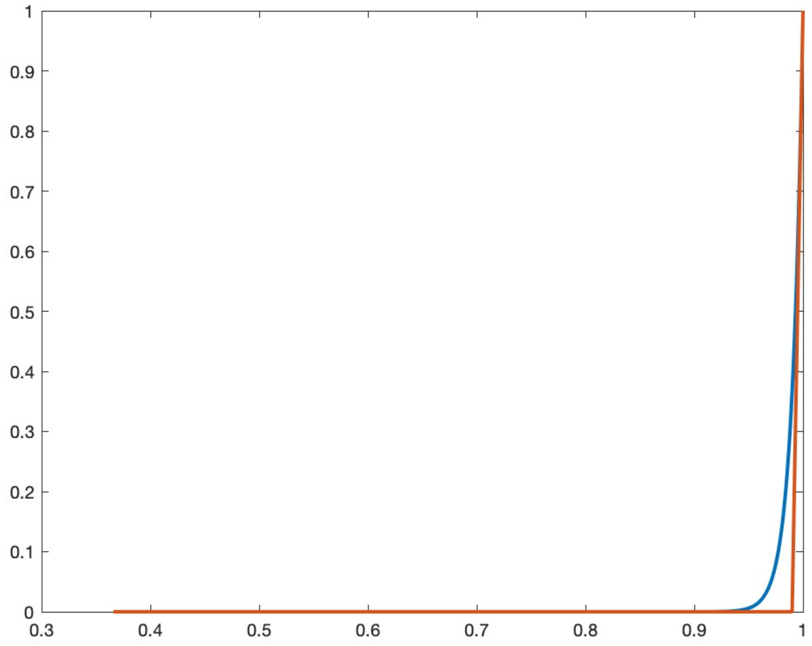
Consider the following IVP, $y' = f(t, y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t, y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad and \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Backward Euler with $h = 0.01$



Euler's method with $h = 0.01$

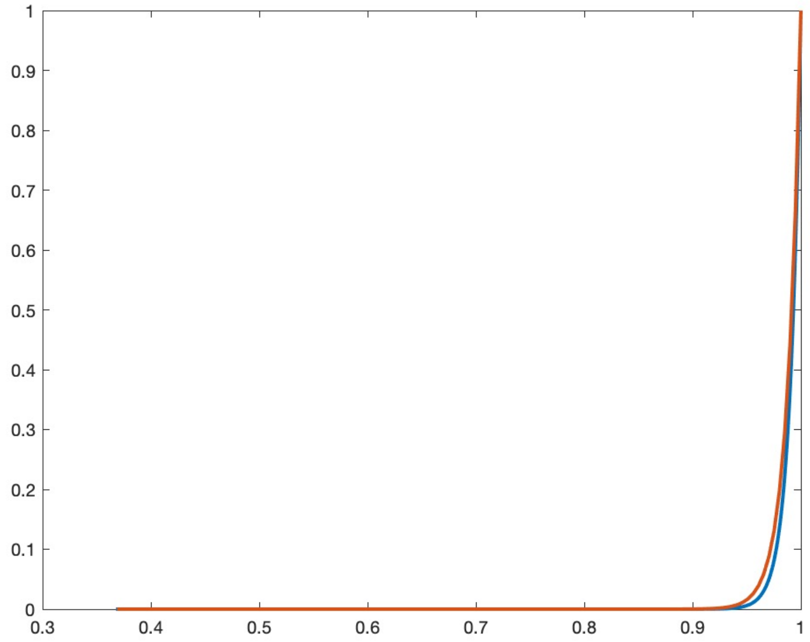


Example

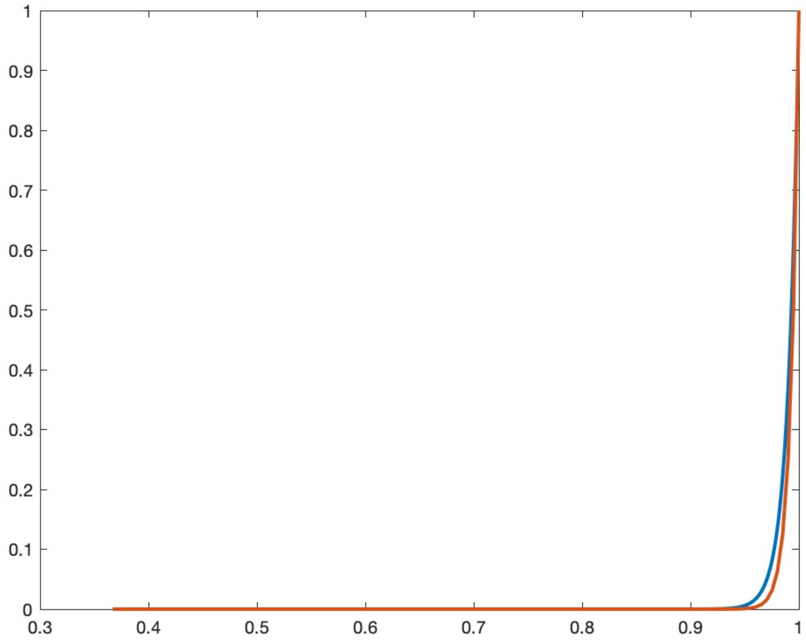
Consider the following IVP, $y' = f(t,y),\ y(0) = y_0$, *where*

$$\mathrm{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad f(t,y) = \begin{bmatrix} -1 & 0 \\ 0 & -100 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \textit{and} \quad y_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Backward Euler with $h = 0.005$



Euler's method with $h = 0.005$



Stiffness

A stable ODE $y' = f(t, y)$ is stiff if the Jacobian matrix f' has eigenvalues that differ greatly in magnitude.

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

Going back to the previous example and using backward Euler, we see that …

Initial Value Problems: Stiffness

Stiffness

A stable ODE $y' = f(t, y)$ is stiff if the Jacobian matrix f' has eigenvalues that differ greatly in magnitude.

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

Going back to the previous example and using backward Euler, we see that …

… is stable throughout. This is consistent with the unconditional stability of the method.

Initial Value Problems: Stiffness

Akash Anand
MATH, IIT KANPUR

Stiffness

A stable ODE $y' = f(t, y)$ is stiff if the Jacobian matrix f' has eigenvalues that differ greatly in magnitude.

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

Going back to the previous example and using backward Euler, we see that …

… is stable throughout. This is consistent with the unconditional stability of the method.

Therefore, implicit methods are required for stiff ODEs.

Initial Value Problems: Stiffness

Akash Anand
MATH, IIT KANPUR

Stiffness

A stable ODE $y' = f(t, y)$ is stiff if the Jacobian matrix f' has eigenvalues that differ greatly in magnitude.

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

Going back to the previous example and using backward Euler, we see that …

… is stable throughout. This is consistent with the unconditional stability of the method.

Therefore, implicit methods are required for stiff ODEs. Thus, a nonlinear (generally) equation must be solved at each step.

Stiffness

A stable ODE $y' = f(t, y)$ *is stiff if the Jacobian matrix* f' *has eigenvalues that differ greatly in magnitude.*

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

Going back to the previous example and using backward Euler, we see that …

… is stable throughout. This is consistent with the unconditional stability of the method.

Therefore, implicit methods are required for stiff ODEs. Thus, a nonlinear (generally) equation must be solved at each step. For fixed point iterations to converge, step size h *must be small which defeats the purpose of using implicit method at the first place.*

Initial Value Problems: Stiffness

Akash Anand
MATH, IIT KANPUR

Stiffness

A stable ODE $y' = f(t, y)$ is stiff if the Jacobian matrix f' has eigenvalues that differ greatly in magnitude.

There may be eigenvalues with relatively large negative real parts (corresponding to strongly damped components of the solution) or relatively large imaginary parts (corresponding to rapidly oscillating components of the solution).

Some methods are very inefficient for stiff equations because the rapidly varying components forces very small step sizes to be used to maintain stability.

Going back to the previous example and using backward Euler, we see that …

… is stable throughout. This is consistent with the unconditional stability of the method.

Therefore, implicit methods are required for stiff ODEs. Thus, a nonlinear (generally) equation must be solved at each step. For fixed point iterations to converge, step size h must be small which defeats the purpose of using implicit method at the first place. As a result, Newton's method or its variants are used.

Module 2

Initial Value Problems

2.4 Implicit method

2.5 Stiffness

2.6 Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

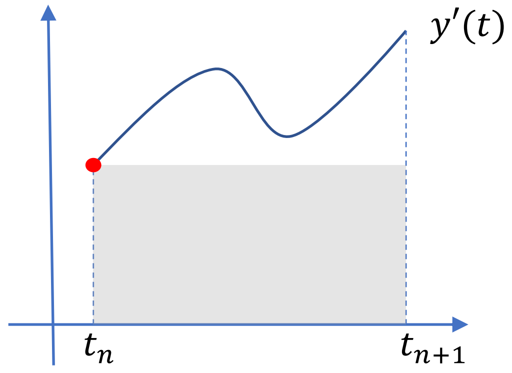
Initial Value Problems: Linear Multistep Methods

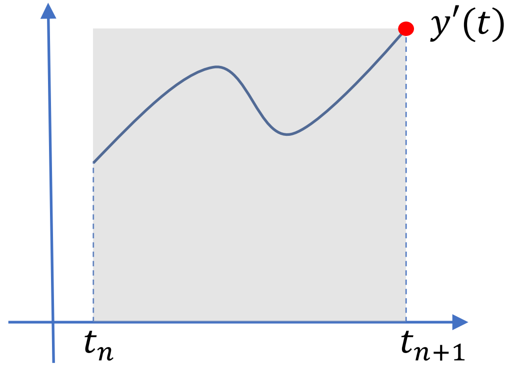
Akash Anand
MATH, IIT KANPUR

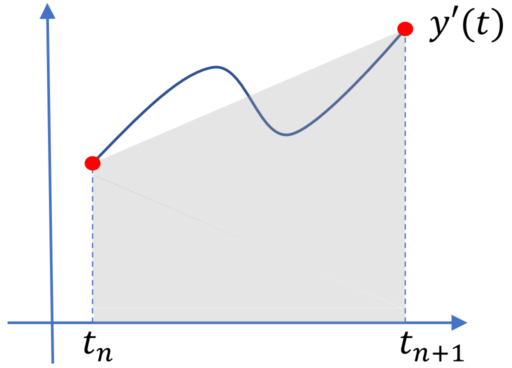
Can we make the method higher order?

Initial Value Problems: Linear Multistep Methods

Can we make the method higher order?



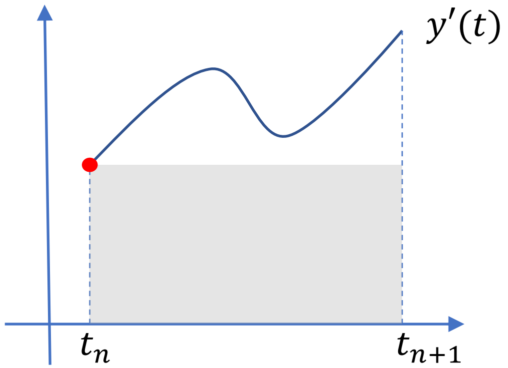


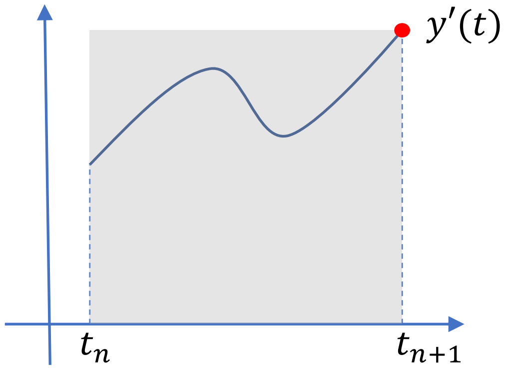


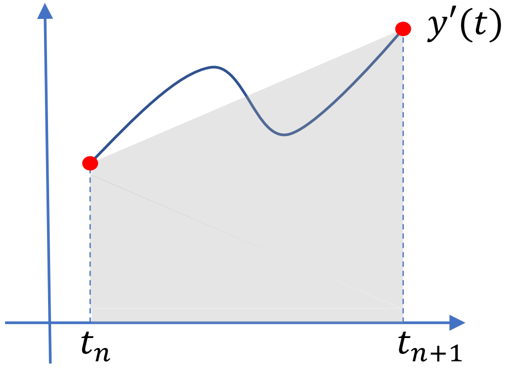
Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.



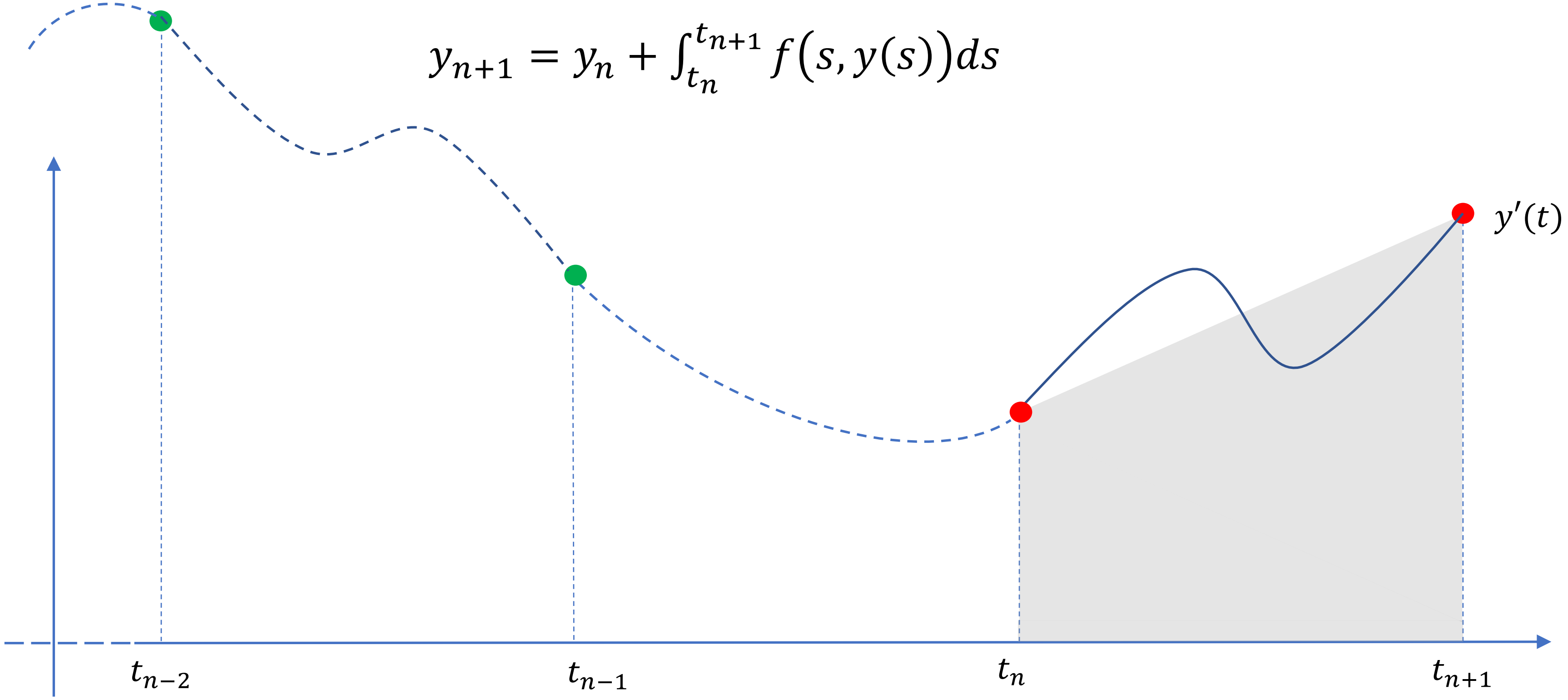




Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds$$



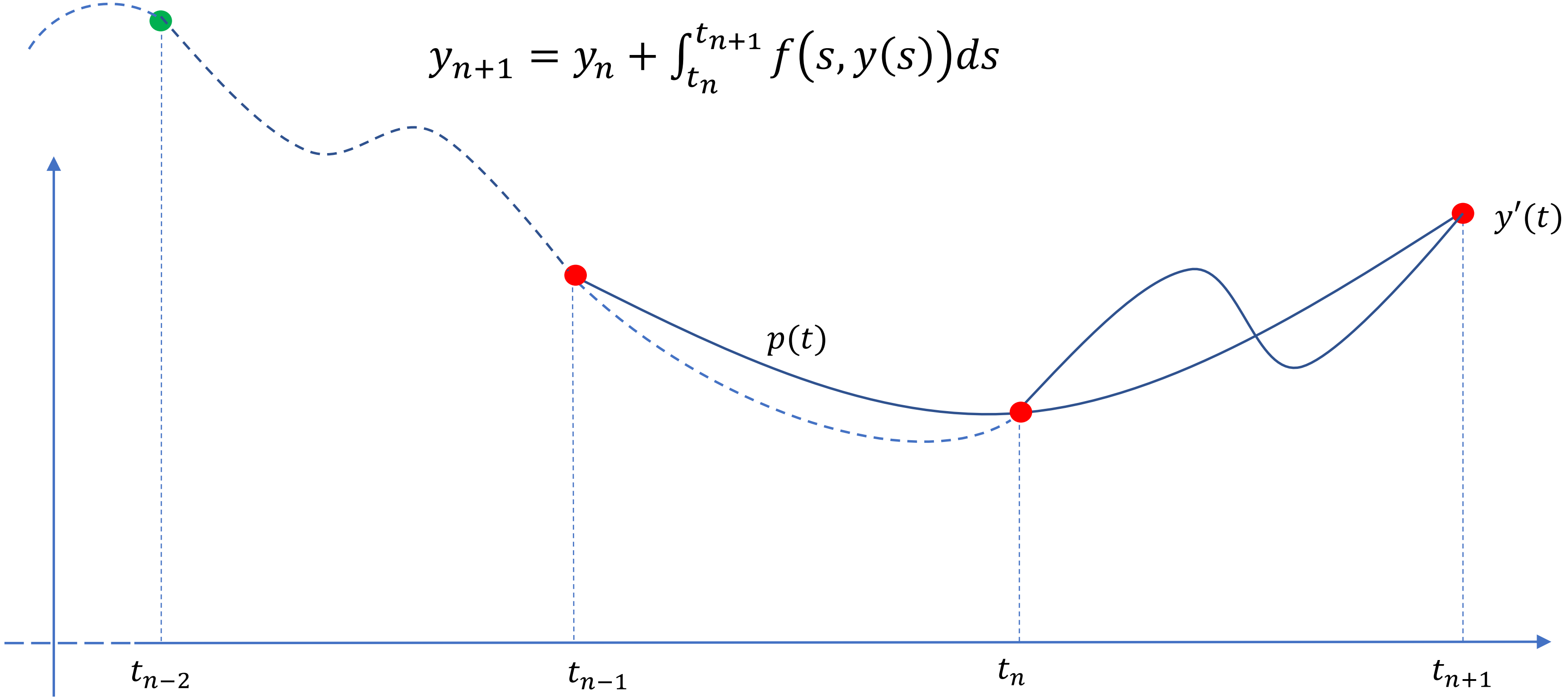
Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds$$



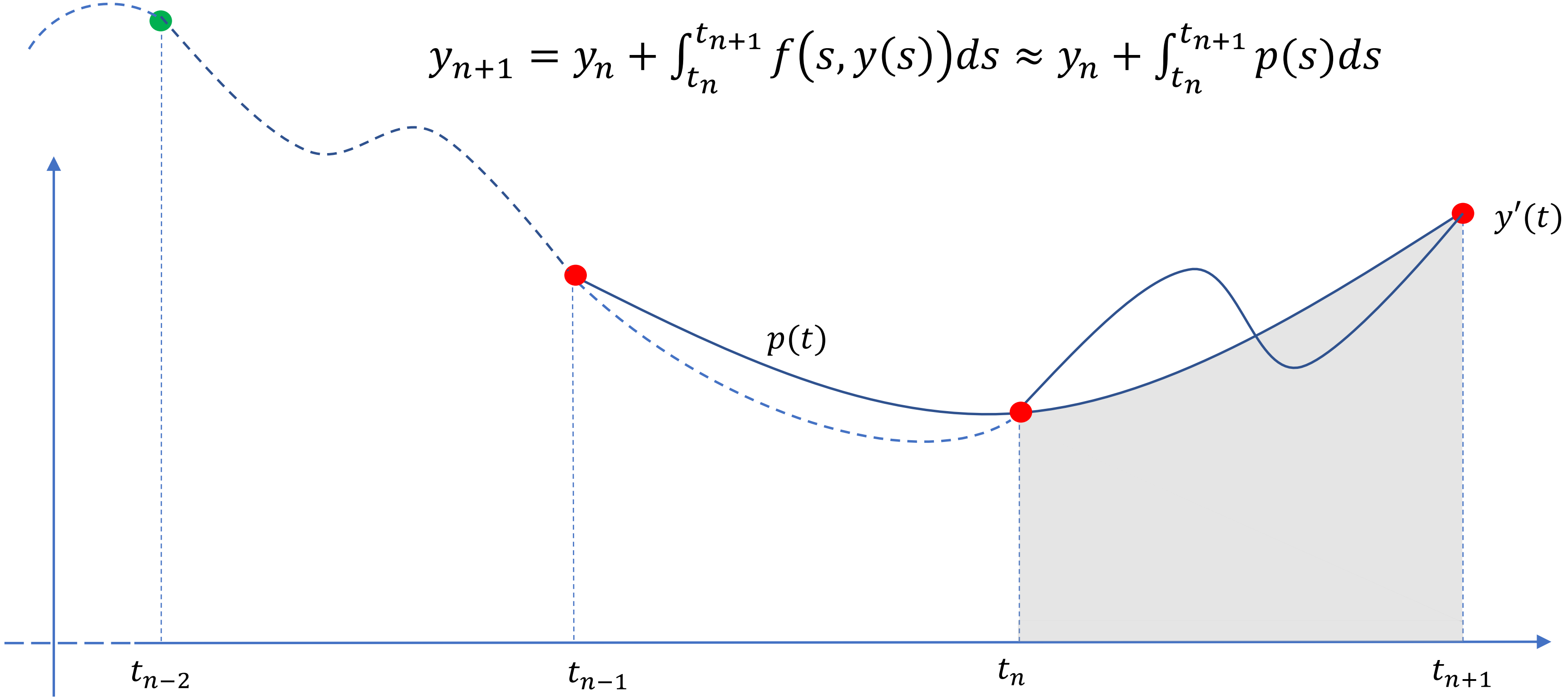
Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds \approx y_n + \int_{t_n}^{t_{n+1}} p(s)ds$$



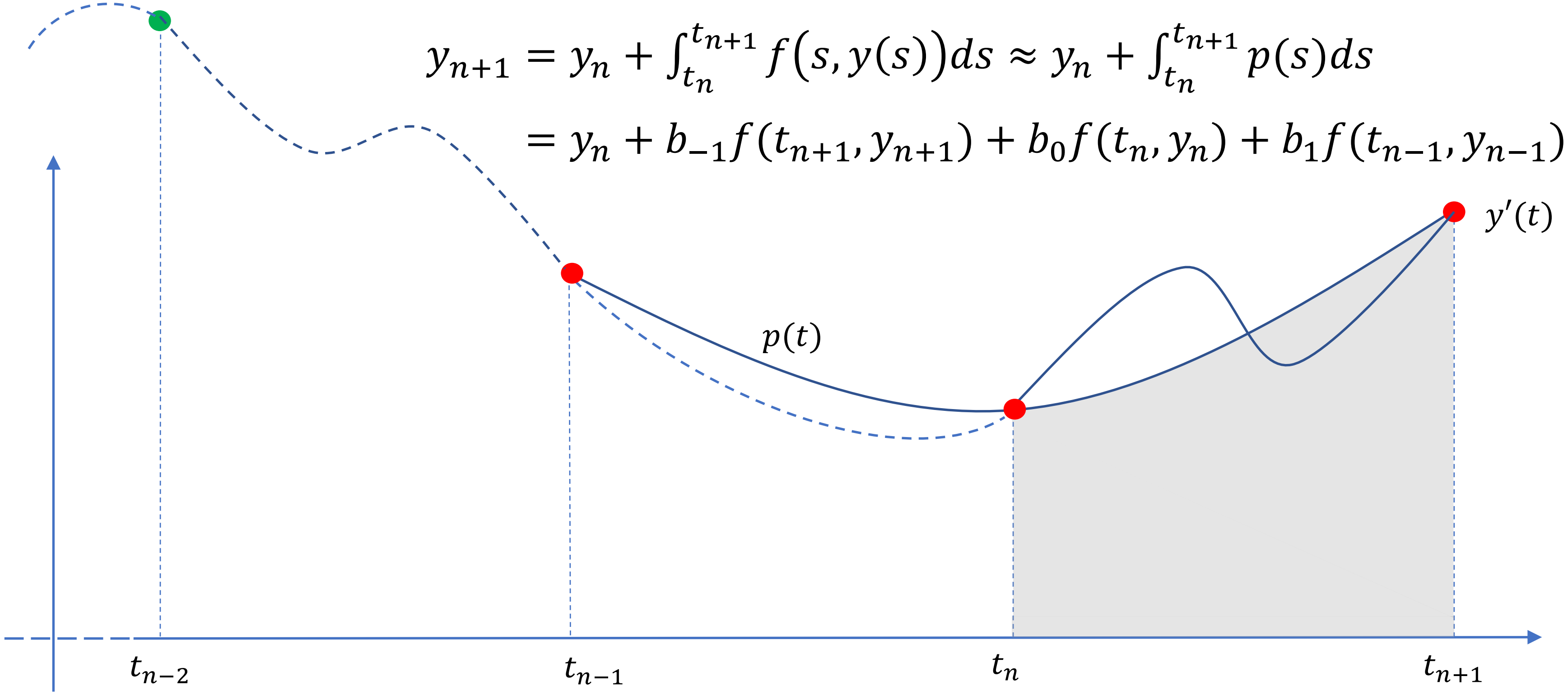
Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds \approx y_n + \int_{t_n}^{t_{n+1}} p(s)ds$$

$$= y_n + b_{-1} f(t_{n+1}, y_{n+1}) + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1})$$



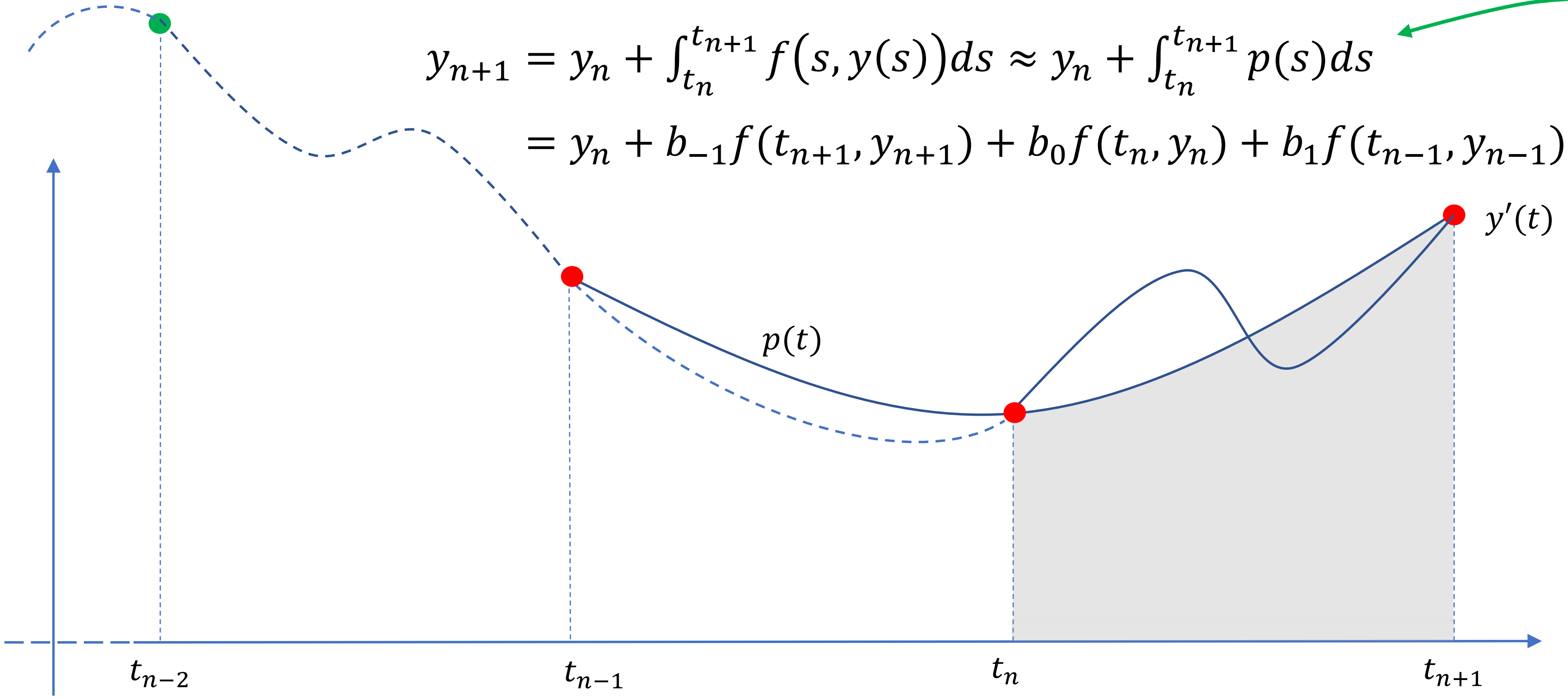
Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds \approx y_n + \int_{t_n}^{t_{n+1}} p(s)ds$$

$$= y_n + b_{-1} f(t_{n+1}, y_{n+1}) + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1})$$

Adams—Moulton method



Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

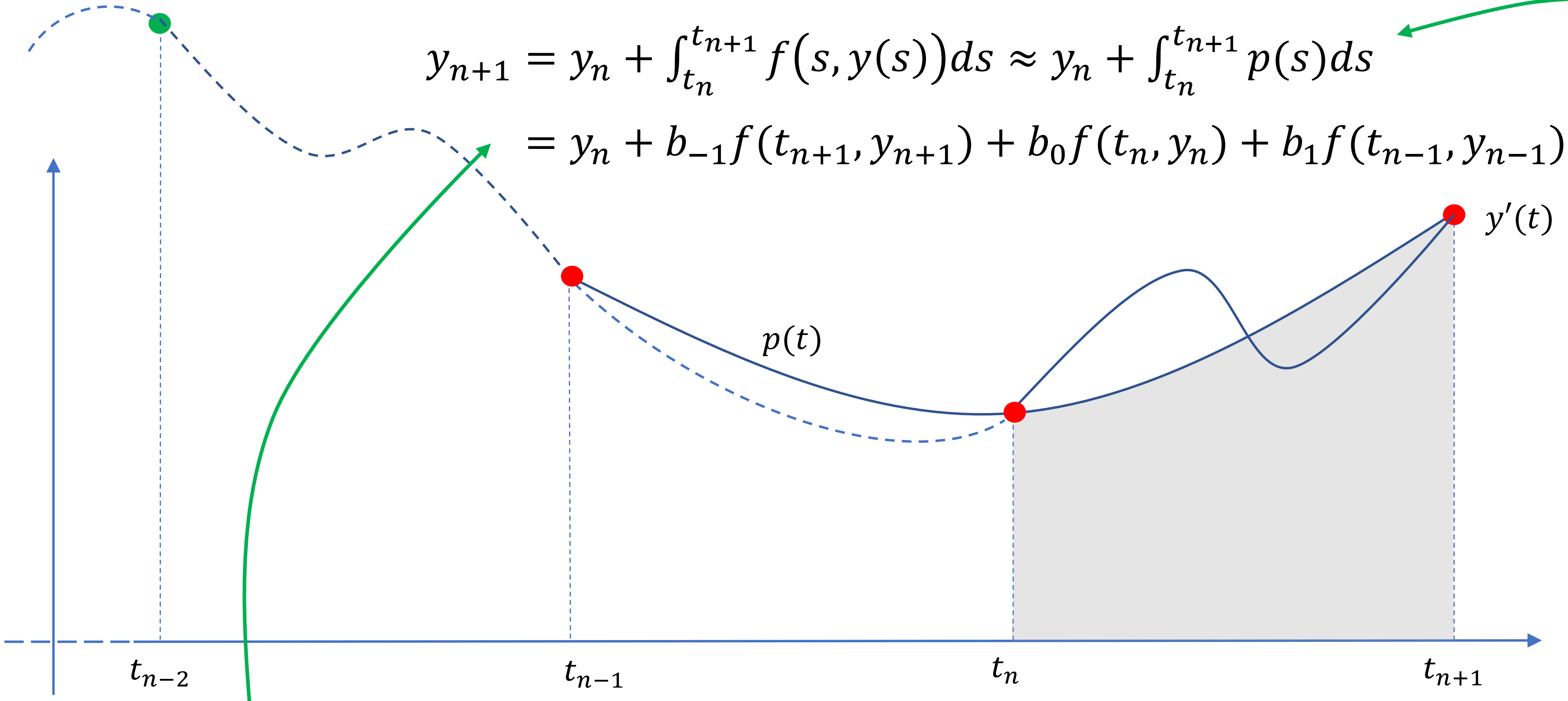
Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds \approx y_n + \int_{t_n}^{t_{n+1}} p(s)ds$$

$$= y_n + b_{-1} f(t_{n+1}, y_{n+1}) + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1})$$

Adams—Moulton method

An implicit method



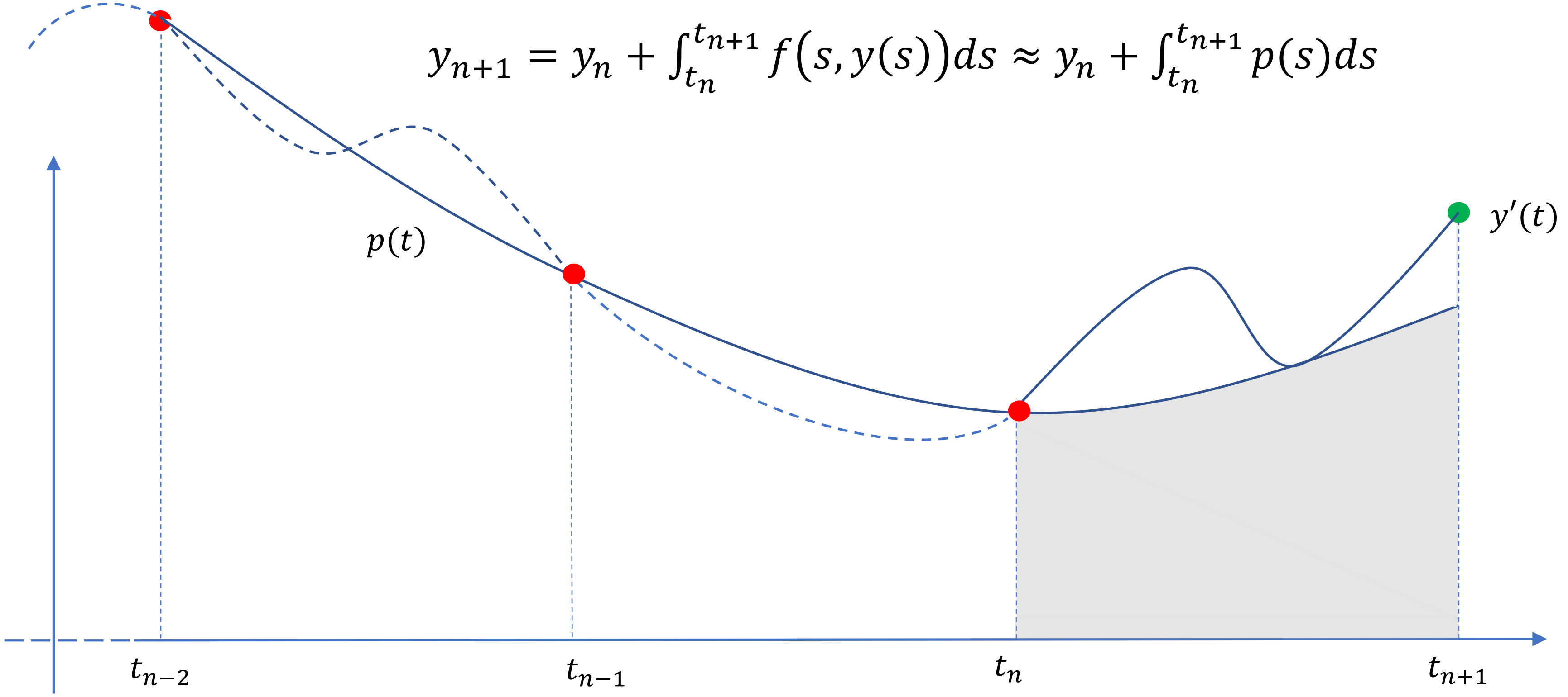
Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds \approx y_n + \int_{t_n}^{t_{n+1}} p(s)ds$$



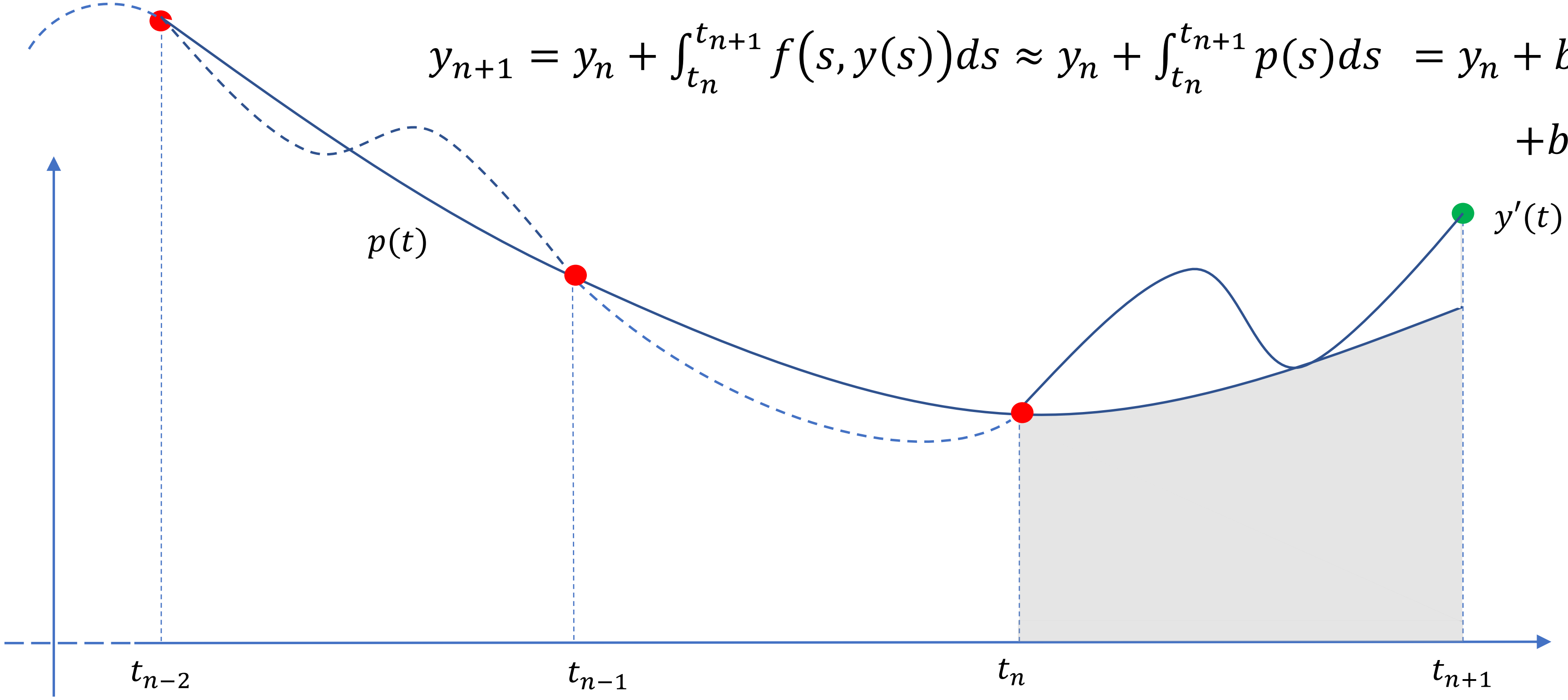
Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds \approx y_n + \int_{t_n}^{t_{n+1}} p(s)ds \;= y_n + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1})$$
$$+ b_2 f(t_{n-2}, y_{n-2})$$



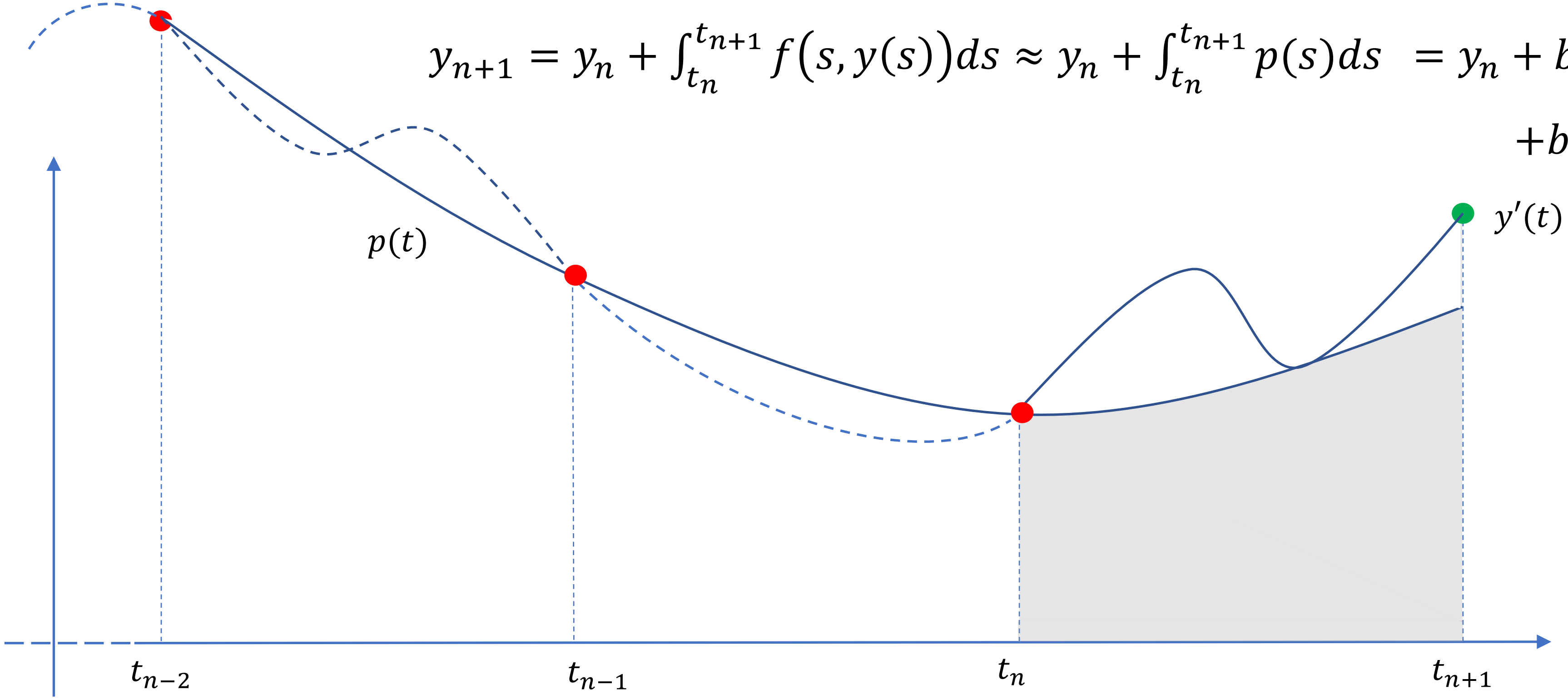
Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

Can we make the method higher order?

Following the idea that we used to derive trapezoidal method, we could use higher order polynomial approximations.

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f\big(s, y(s)\big)ds \approx y_n + \int_{t_n}^{t_{n+1}} p(s)ds \;= y_n + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1})$$
$$+ b_2 f(t_{n-2}, y_{n-2})$$



Adams-Bashford method

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider methods that take constant step size h and determine y_{n+1} using the values from several preceding steps:

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \dots, y_{n-k}, h).$$

Here y_{n+1} depends on $k+1$ previous values, so this is called a $(k+1)$-step method.

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider methods that take constant step size h *and determine* y_{n+1} *using the values from several preceding steps:*

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \dots, y_{n-k}, h).$$

Here y_{n+1} *depends on* $k+1$ *previous values, so this is called a* $(k+1)$*-step method.*

For $k \geq 1$*, that is, for a* 2 *or more step method, how do we start the time marching? Note that we need to know* $y_0, \dots, y_k$*, to compute* y_{k+1} *and we typically only know* y_0*!*

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider methods that take constant step size h and determine y_{n+1} using the values from several preceding steps:

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \dots, y_{n-k}, h).$$

Here y_{n+1} depends on $k+1$ previous values, so this is called a $(k+1)$-step method.

For $k \geq 1$, that is, for a 2 or more step method, how do we start the time marching? Note that we need to know $y_0, \dots, y_k$, to compute y_{k+1} and we typically only know y_0!

… we use some other method such as a single step method.

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider methods that take constant step size h and determine y_{n+1} using the values from several preceding steps:

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \dots, y_{n-k}, h).$$

Here y_{n+1} depends on $k+1$ previous values, so this is called a $(k+1)$-step method.

For $k \geq 1$, that is, for a 2 or more step method, how do we start the time marching? Note that we need to know $y_0, \dots, y_k$, to compute y_{k+1} and we typically only know y_0!

… we use some other method such as a single step method.

Examples:

1. $y_{n+1} = y_n + hf(t_n, y_n)$ … *an explicit one step method*
2. $y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$ … *an implicit one step method*
3. $y_{n+1} = y_n + h\,(f(t_n, y_n) + f(t_{n+1}, y_{n+1}))/2$ … *an implicit one step method*

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider methods that take constant step size h and determine y_{n+1} using the values from several preceding steps:

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \dots, y_{n-k}, h).$$

Here y_{n+1} depends on $k+1$ previous values, so this is called a $(k+1)$-step method.

For $k \geq 1$, that is, for a 2 or more step method, how do we start the time marching? Note that we need to know $y_0, \dots, y_k$, to compute y_{k+1} and we typically only know y_0!

… we use some other method such as a single step method.

Examples:

1. $y_{n+1} = y_n + hf(t_n, y_n)$ *… an explicit one step method*
2. $y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$ *… an implicit one step method*
3. $y_{n+1} = y_n + h\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right)/2$ *… an implicit one step method*
4. $y_{n+1} = y_n + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1}) + b_2 f(t_{n-2}, y_{n-2})$ *… an explicit three step method*

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider methods that take constant step size h and determine y_{n+1} using the values from several preceding steps:

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \ldots, y_{n-k}, h).$$

Here y_{n+1} depends on $k+1$ previous values, so this is called a $(k+1)$-step method.

For $k \geq 1$, that is, for a 2 or more step method, how do we start the time marching? Note that we need to know $y_0, \ldots, y_k$, to compute y_{k+1} and we typically only know y_0!

… we use some other method such as a single step method.

Examples:

1. $y_{n+1} = y_n + hf(t_n, y_n)$ … *an explicit one step method*
2. $y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$ … *an implicit one step method*
3. $y_{n+1} = y_n + h\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right)/2$ … *an implicit one step method*
4. $y_{n+1} = y_n + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1}) + b_2 f(t_{n-2}, y_{n-2})$ … *an explicit three step method*
5. $y_{n+1} = y_n + h\left(f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))\right)/2$ … *an explicit one step method*

We consider methods that take constant step size h *and determine* y_{n+1} *using the values from several preceding steps:*

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \ldots, y_{n-k}, h).$$

Here y_{n+1} *depends on* $k+1$ *previous values, so this is called a* $(k+1)$*-step method.*

For $k \geq 1$*, that is, for a* 2 *or more step method, how do we start the time marching? Note that we need to know* $y_0, \ldots, y_k$*, to compute* y_{k+1} *and we typically only know* y_0*!*

… we use some other method such as a single step method.

Improved Euler Method

Examples:

1. $y_{n+1} = y_n + hf(t_n, y_n)$ … *an explicit one step method*
2. $y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$ … *an implicit one step method*
3. $y_{n+1} = y_n + h\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right)$ … *an implicit one step method*
4. $y_{n+1} = y_n + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1}) + b_2 f(t_{n-2}, y_{n-2})$ … *an explicit three step method*
5. $y_{n+1} = y_n + h\left(f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))\right)/2$ … *an explicit one step method*

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider methods that take constant step size h and determine y_{n+1} using the values from several preceding steps:

$$y_{n+1} = \Phi(f, t_n, y_{n+1}, y_n, y_{n-1}, \dots, y_{n-k}, h).$$

Here y_{n+1} depends on $k+1$ previous values, so this is called a $(k+1)$-step method.

For $k \geq 1$, that is, for a 2 or more step method, how do we start the time marching? Note that we need to know $y_0, \dots, y_k$, to compute y_{k+1} and we typically only know y_0!

… we use some other method such as a single step method.

Improved Euler Method

Examples:

1. $y_{n+1} = y_n + hf(t_n, y_n)$ … *an explicit one step method*
2. $y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$ … *an implicit one step method*
3. $y_{n+1} = y_n + h\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right)$ … *an implicit one step method*
4. $y_{n+1} = y_n + b_0 f(t_n, y_n) + b_1 f(t_{n-1}, y_{n-1}) + b_2 f(t_{n-2}, y_{n-2})$ … *an explicit three step method*
5. $y_{n+1} = y_n + h\left(f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))\right)/2$ … *an explicit one step method*

Φ Is linear in y_n, $f(t_n, y_n)$, $f(t_{n+1}, y_{n+1})$, etc.

non-linear Φ

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider linear multistep methods with constant step size, which by definition, are methods of the form

$$y_{n+1} = -\mathrm{a}_0\mathrm{y}_\mathrm{n} - \mathrm{a}_1\mathrm{y}_{\mathrm{n}-1} - \cdots - a_k y_{n-k} + h[b_{-1}f_{n+1} + b_0 f_n + \cdots + b_k f_{n-k}]$$

where f_n denotes $f(t_n, y_n)$ (for brevity) and a_j, b_j are constants which must be given and determine the specific method.

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider linear multistep methods with constant step size, which by definition, are methods of the form

$$y_{n+1} = -\mathrm{a}_0\mathrm{y}_\mathrm{n} - \mathrm{a}_1\mathrm{y}_{\mathrm{n}-1} - \cdots - a_k y_{n-k} + h[b_{-1}f_{n+1} + b_0 f_n + \cdots + b_k f_{n-k}]$$

where f_n *denotes* $f(t_n, y_n)$ *(for brevity) and* a_j*,* b_j *are constants which must be given and determine the specific method.*

For explicit linear multistep method, $b_{-1} = 0$.

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider linear multistep methods with constant step size, which by definition, are methods of the form

$$y_{n+1} = -\mathrm{a}_0 \mathrm{y}_\mathrm{n} - \mathrm{a}_1 \mathrm{y}_{\mathrm{n}-1} - \cdots - a_k y_{n-k} + h[b_{-1} f_{n+1} + b_0 f_n + \cdots + b_k f_{n-k}]$$

where f_n *denotes* $f(t_n, y_n)$ *(for brevity) and* a_j, b_j *are constants which must be given and determine the specific method.*

For explicit linear multistep method, $b_{-1} = 0$.

It is also convenient to define $a_{-1} = 1$, *so that the method can we written more concisely as*

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f_{n-j} \,.$$

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider linear multistep methods with constant step size, which by definition, are methods of the form

$$y_{n+1} = -\mathrm{a}_0 \mathrm{y}_\mathrm{n} - \mathrm{a}_1 \mathrm{y}_{\mathrm{n}-1} - \cdots - a_k y_{n-k} + h[b_{-1} f_{n+1} + b_0 f_n + \cdots + b_k f_{n-k}]$$

where f_n *denotes* $f(t_n, y_n)$ *(for brevity) and* a_j, b_j *are constants which must be given and determine the specific method.*

For explicit linear multistep method, $b_{-1} = 0$.

It is also convenient to define $a_{-1} = 1$, *so that the method can we written more concisely as*

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f_{n-j}\,.$$

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ *where* L *is the Lipschitz constant for* f. *Then for any* $h < h_0$ *and any* $y_n, y_{n-1}, \dots, y_{n-k}$, *there is a unique* y_{n+1} *such that*

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j})\,.$$

Initial Value Problems: Linear Multistep Methods

Akash Anand
MATH, IIT KANPUR

We consider linear multistep methods with constant step size, which by definition, are methods of the form

$$y_{n+1} = -\mathrm{a}_0\mathrm{y}_\mathrm{n} - \mathrm{a}_1\mathrm{y}_{\mathrm{n}-1} - \cdots - a_k y_{n-k} + h[b_{-1}f_{n+1} + b_0 f_n + \cdots + b_k f_{n-k}]$$

where f_n denotes $f(t_n, y_n)$ (for brevity) and a_j, b_j are constants which must be given and determine the specific method.

For explicit linear multistep method, $b_{-1} = 0$.

It is also convenient to define $a_{-1} = 1$, so that the method can we written more concisely as

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f_{n-j}\,.$$

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ where L is the Lipschitz constant for f. Then for any $h < h_0$ and any $y_n, y_{n-1}, \ldots, y_{n-k}$, there is a unique y_{n+1} such that

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j})\,.$$

Proof.

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ *where* L *is the Lipschitz constant for* f. *Then for any* $h < h_0$ *and any* $y_n, y_{n-1}, \dots, y_{n-k}$, *there is a unique* y_{n+1} *such that*

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j}).$$

Proof.

Define

$$F(z) = -\sum_{j=0}^{k} a_j y_{n-j} + h \sum_{j=0}^{k} b_j f(t_{n-j}, y_{n-j}) + h b_{-1} f(t_{n+1}, z).$$

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ where L is the Lipschitz constant for f. Then for any $h < h_0$ and any $y_n, y_{n-1}, \dots, y_{n-k}$, there is a unique y_{n+1} such that

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j}).$$

Proof.

Define

$$F(z) = -\sum_{j=0}^{k} a_j y_{n-j} + h \sum_{j=0}^{k} b_j f(t_{n-j}, y_{n-j}) + h b_{-1} f(t_{n+1}, z).$$

Now, $F(z)$ is Lipschitz with Lipschitz constant less than or equal to $h|b_{-1}|L$ (why?).

Akash Anand
MATH, IIT KANPUR

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ *where* L *is the Lipschitz constant for* f. *Then for any* $h < h_0$ *and any* $y_n, y_{n-1}, \dots, y_{n-k}$, *there is a unique* y_{n+1} *such that*

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j}).$$

Proof.

Define

$$F(z) = -\sum_{j=0}^{k} a_j y_{n-j} + h \sum_{j=0}^{k} b_j f(t_{n-j}, y_{n-j}) + h b_{-1} f(t_{n+1}, z).$$

Now, $F(z)$ *is Lipschitz with Lipschitz constant less than or equal to* $h|b_{-1}|L$ *(why?). By hypothesis* $(h < h_0)$, *the Lipschitz constant is strictly less than 1, that is,* $F(z)$ *is a contraction.*

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ *where* L *is the Lipschitz constant for* f. *Then for any* $h < h_0$ *and any* $y_n, y_{n-1}, \dots, y_{n-k}$, *there is a unique* y_{n+1} *such that*

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j}).$$

Proof.

Define

$$F(z) = -\sum_{j=0}^{k} a_j y_{n-j} + h \sum_{j=0}^{k} b_j f(t_{n-j}, y_{n-j}) + h b_{-1} f(t_{n+1}, z).$$

Now, $F(z)$ *is Lipschitz with Lipschitz constant less than or equal to* $h|b_{-1}|L$ *(why?). By hypothesis* $(h < h_0)$, *the Lipschitz constant is strictly less than 1, that is,* $F(z)$ *is a contraction. The contraction mapping theorem then guarantees a unique fixed point, say* y_{n+1}.

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ where L is the Lipschitz constant for f. Then for any $h < h_0$ and any $y_n, y_{n-1}, \dots, y_{n-k}$, there is a unique y_{n+1} such that

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j}) \, .$$

Proof.

Define

$$F(z) = -\sum_{j=0}^{k} a_j y_{n-j} + h \sum_{j=0}^{k} b_j f(t_{n-j}, y_{n-j}) \; + h b_{-1} f(t_{n+1}, z).$$

Now, $F(z)$ is Lipschitz with Lipschitz constant less than or equal to $h|b_{-1}|L$ (why?). By hypothesis $(h < h_0)$, the Lipschitz constant is strictly less than 1, that is, $F(z)$ is a contraction. The contraction mapping theorem then guarantees a unique fixed point, say y_{n+1}. Thus, we have $y_{n+1} = F(y_{n+1})$, as desired.

Theorem

Let $h_0 = 1/(|b_{-1}|L)$ where L is the Lipschitz constant for f. Then for any $h < h_0$ and any $y_n, y_{n-1}, \dots, y_{n-k}$, there is a unique y_{n+1} such that

$$\sum_{j=-1}^{k} a_j y_{n-j} = h \sum_{j=-1}^{k} b_j f(t_{n-j}, y_{n-j}).$$

Proof.

Define

$$F(z) = -\sum_{j=0}^{k} a_j y_{n-j} + h \sum_{j=0}^{k} b_j f(t_{n-j}, y_{n-j}) + h b_{-1} f(t_{n+1}, z).$$

Now, $F(z)$ is Lipschitz with Lipschitz constant less than or equal to $h|b_{-1}|L$ (why?). By hypothesis $(h < h_0)$, the Lipschitz constant is strictly less than 1, that is, $F(z)$ is a contraction. The contraction mapping theorem then guarantees a unique fixed point, say y_{n+1}. Thus, we have $y_{n+1} = F(y_{n+1})$, as desired.

Remark

The contraction mapping theorem also implies that the solution can be computed by fixed point iteration as is often done in practice. Moreover, only a fixed (small) number of iterations are made (introducing an additional error).