Initial Value Problems: ODEs

Runge-Kutta Schemes

M.Sc. in Mathematical Modelling & Scientific Computing,
Practical Numerical Analysis

Michaelmas Term 2025, Lecture 6

The Problem

We continue to study the scalar first order initial value problem: find $u(t)$ such that

$$\begin{aligned} u'(t) &= f(t,u)\,, \quad t>0\,, \\ u(0) &= u_0\,. \end{aligned} \tag{1}$$

Recall that in lecture 5, in order to solve (1) numerically over the time interval $[0,T]$, we defined a set of time points at which we wish to approximate the solution. We set $t_n = n\Delta t$ for $n = 0,1,\ldots,N$ where $\Delta t = T/N$. We then used the θ-method to approximate $u(t_n)$ by U_n satisfying

$$\frac{U_{n+1}-U_n}{\Delta t} = \theta f(t_{n+1},U_{n+1}) + (1-\theta) f(t_n,U_n) \tag{2}$$

for $n = 0,1,\ldots$ and with $U_0 = u_0$.

Modifications of θ-Methods

As we have already stated, unless $\theta = 0$, the θ-method requires us to solve a nonlinear equation at each timestep. However, there exist modified methods to avoid this.

Recall the Crank Nicolson scheme

$$\frac{U_{n+1} - U_n}{\Delta t} = \frac{1}{2}\left(f(t_{n+1}, U_{n+1}) + f(t_n, U_n)\right) .$$

We can approximate U_{n+1} using the explicit Euler scheme. This leads to the improved Euler method

$$\frac{U_{n+1} - U_n}{\Delta t} = \frac{1}{2}\left(f(t_{n+1}, U_n + \Delta t f(t_n, U_n)) + f(t_n, U_n)\right) .$$

This has a truncation error $T_n = \mathcal{O}(\Delta t^2)$.

Explicit Runge-Kutta Schemes

The improved Euler method is a specific example of an explicit Runge-Kutta scheme. Such schemes take the general form

$$\frac{U_{n+1} - U_n}{\Delta t} = \sum_{i=1}^{s} b_i k_i$$

where

$$k_1 = f(t_n, U_n)$$

and

$$k_i = f\left(t_n + c_i \Delta t, U_n + \Delta t \sum_{j=1}^{i-1} a_{i,j} k_j\right),$$

for $i = 2, \ldots, s$. The k_is are known as the stages of the method and the method is often referred to as an s-stage method.

Runge-Kutta Schemes — Butcher Tableaux

The coefficients of explicit Runge-Kutta schemes are chosen to make the methods as high order as possible and are often stored as Butcher tableaux in the form

$$
\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{2,1} & & & & \\
c_3 & a_{3,1} & a_{3,2} & & & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_s & a_{s,1} & a_{s,2} & \dots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \dots & b_{s-1} & b_s
\end{array}
$$

Example: Improved Euler

The improved Euler scheme

$$\frac{U_{n+1} - U_n}{\Delta t} = \frac{1}{2}\left(f(t_{n+1}, U_n + \Delta t f(t_n, U_n)) + f(t_n, U_n)\right) ,$$

can be written in the form

$$\frac{U_{n+1} - U_n}{\Delta t} = \frac{1}{2}(k_1 + k_2) ,$$

where

$$k_1 = f(t_n, U_n)$$

and

$$k_2 = f(t_n + \Delta t, U_n + \Delta t k_1) .$$

Thus the Butcher tableau takes the form

0		
1	1	
	$\frac{1}{2}$	$\frac{1}{2}$

Example: Modified Euler

Another commonly used 2-stage scheme is the modified Euler scheme, given by

$$\frac{U_{n+1} - U_n}{\Delta t} = f\left(t_n + \frac{1}{2}\Delta t, U_n + \frac{1}{2}\Delta t f(t_n, U_n)\right) .$$

The Butcher tableau for the modified Euler scheme takes the form

$$\begin{array}{c|cc} 0 & & \\ \frac{1}{2} & \frac{1}{2} & \\ \hline & 0 & 1 \end{array}$$

Example: RK4

Finally, a very common 4-stage method is the so-called RK4 scheme, defined by the Butcher tableau

$$
\begin{array}{c|cccc}
0 & & & & \\
\frac{1}{2} & \frac{1}{2} & & & \\
\frac{1}{2} & 0 & \frac{1}{2} & & \\
1 & 0 & 0 & 1 & \\
\hline
 & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6}
\end{array}
$$

Derivation of Explicit Runge-Kutta Schemes

As already stated, the coefficients of explicit Runge-Kutta schemes are chosen to make the methods as high order as possible. We can do this using Taylor series expansions of the truncation error.

The truncation error for an explicit Runge-Kutta scheme is given by

$$T_n = \frac{u(t_{n+1}) - u(t_n)}{\Delta t} - \sum_{i=1}^{s} b_i \tilde{k}_i \ ,$$

where

$$\tilde{k}_1 = f(t_n, u(t_n))$$

and

$$\tilde{k}_i = f\left(t_n + c_i \Delta t, u(t_n) + \Delta t \sum_{j=1}^{i-1} a_{i,j} \tilde{k}_j \right) \ ,$$

for $i = 2, \ldots, s$.

Derivation of Explicit Runge-Kutta Schemes: 2-Stage Example

If $s = 2$ we have

$$T_n = \frac{u(t_{n+1}) - u(t_n)}{\Delta t} - (b_1 \tilde{k}_1 + b_2 \tilde{k}_2) ,$$

where

$$\begin{aligned} \tilde{k}_1 &= f(t_n, u(t_n)) , \\ \tilde{k}_2 &= f(t_n + c_2 \Delta t, u(t_n) + \Delta t a_{1,1} \tilde{k}_1) . \end{aligned}$$

Performing a Taylor series expansion of $u(t_{n+1})$ about the point t_n gives

$$u(t_{n+1}) = u(t_n) + \Delta t u'(t_n) + \frac{1}{2}\Delta t^2 u''(t_n) + \frac{1}{6}\Delta t^3 u'''(t_n) + \mathcal{O}(\Delta t^4) .$$

Derivation of Explicit Runge-Kutta Schemes: 2-Stage Example

We also perform a Taylor series expansion of $\tilde{k}_2$ about the point $(t_n, u(t_n))$ to get

$$\begin{aligned}
\tilde{k}_2 &= f(t_n + c_2\Delta t, u(t_n) + \Delta t a_{1,1}\tilde{k}_1) \\
&= f(t_n + c_2\Delta t, u(t_n) + \Delta t a_{1,1} f(t_n, u(t_n))) \\
&= f(t_n, u(t_n)) + c_2\Delta t f_t(t_n, u(t_n)) + \Delta t a_{1,1} f(t_n, u(t_n)) f_u(t_n, u(t_n)) \\
&\quad + \frac{1}{2} c_2^2 \Delta t^2 f_{tt}(t_n, u(t_n)) + c_2 a_{1,1} \Delta t^2 f(t_n, u(t_n)) f_{tu}(t_n, u(t_n)) \\
&\quad + \frac{1}{2} a_{1,1}^2 \Delta t^2 (f(t_n, u(t_n)))^2 f_{uu}(t_n, u(t_n)) + \mathcal{O}(\Delta t^3) \, .
\end{aligned}$$

To simplify the notation, we will suppress the arguments so that u represents $u(t_n)$ and f represents $f(t_n, u(t_n))$ (and similarly for all (partial) derivatives).

Derivation of Explicit Runge-Kutta Schemes: 2-Stage Example

Now we substitute these expansions into the expression for the truncation error

$$\begin{aligned} T_n &= \frac{u(t_{n+1}) - u(t_n)}{\Delta t} - (b_1 f(t_n, u(t_n)) \\ &\quad + b_2 f\left(t_n + c_2 \Delta t, u(t_n) + \Delta t a_{1,1} f(t_n, u(t_n))\right)) \ , \end{aligned}$$

simplify and reorder to get

$$\begin{aligned} T_n &= \left(u' - (b_1 + b_2) f\right) + \Delta t \left[\frac{1}{2} u'' - b_2 (c_2 f_t + a_{1,1} f \ f_u)\right] \\ &\quad + \Delta t^2 \left[\frac{1}{6} u''' - \frac{b_2}{2} \left(c_2^2 f_{tt} + 2 c_2 a_{1,1} f \ f_{tu} + a_{1,1}^2 f^2 f_{uu}\right)\right] \\ &\quad + \mathcal{O}(\Delta t^3) \ . \end{aligned}$$

Derivation of Explicit Runge-Kutta Schemes: 2-Stage Example

Now recall that

$$u'(t) \;=\; f(t, u(t))\,,$$

so that the chain rule gives

$$\begin{aligned} u''(t) &= f_t + f\ f_u\,, \\ u'''(t) &= f_{tt} + 2f\ f_{tu} + f_t f_u + f\ f_u^2 + f^2 f_{uu}\,. \end{aligned}$$

Then we have

$$\begin{aligned} T_n \;=\; & (1-(b_1+b_2))\, f + \Delta t \left[\left(\frac{1}{2} - b_2 c_2 \right) f_t + \left(\frac{1}{2} - b_2 a_{1,1} \right) f\ f_u \right] \\ & + \Delta t^2 \left[\left(\frac{1}{6} - \frac{b_2}{2} c_2^2 \right) f_{tt} + \left(\frac{1}{3} - b_2 c_2 a_{1,1} \right) f\ f_{tu} \right. \\ & \left. + \left(\frac{1}{6} - \frac{b_2}{2} a_{1,1}^2 \right) f^2 f_{uu} + \frac{1}{6} \left(f_u f_t + f\ f_u^2 \right) \right] \\ & + \mathcal{O}(\Delta t^3)\,. \end{aligned}$$

Derivation of Explicit Runge-Kutta Schemes: 2-Stage Example

We can make the method 2nd order (i.e. $T_n = \mathcal{O}(\Delta t^2)$) by picking

$$\begin{aligned} 1 - (b_1 + b_2) &= 0 \\ \frac{1}{2} - b_2 c_2 &= 0 \\ \frac{1}{2} - b_2 a_{1,1} &= 0 \,. \end{aligned}$$

Note improved Euler is the particular choice $b_1 = b_2 = 1/2$, $c_2 = a_{1,1} = 1$ and modified Euler is the particular choice $b_1 = 0$, $b_2 = 1$, $c_2 = a_{1,1} = 1/2$.

However, we cannot make the method 3rd order because we cannot eliminate the $f_u f_t$ and $f\ f_u^2$ terms.

Order of Runge-Kutta Schemes

Theorem

The order p of an explicit s-stage Runge-Kutta method is bounded by $p \leq s$.

It is possible to construct Runge-Kutta methods that achieve this maximal order.

Example 1

We revisit Example 1 from lecture 5, namely

$$\begin{aligned} u'(t) &= \lambda u\,, \quad t > 0\,, \\ u(0) &= 1\,. \end{aligned}$$

Numerical schemes for this are:

- Explicit Euler:

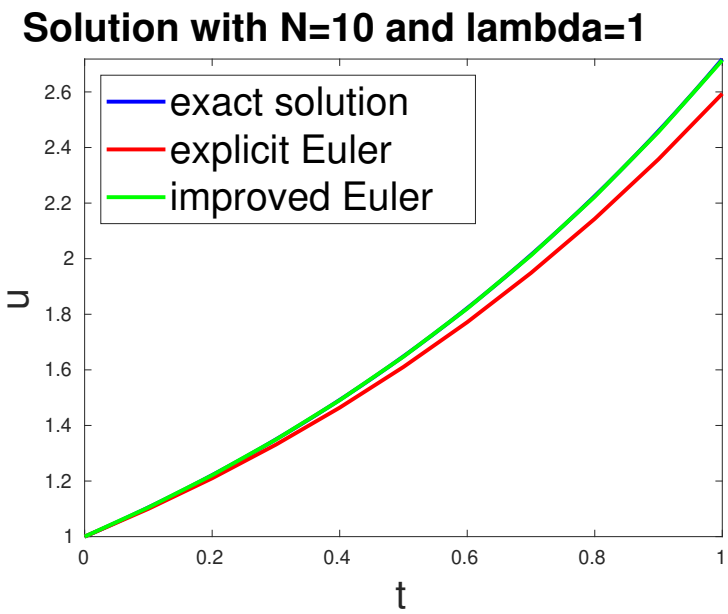
$$U_{n+1} = U_n + \lambda \Delta t U_n = (1 + \lambda \Delta t) U_n\,.$$

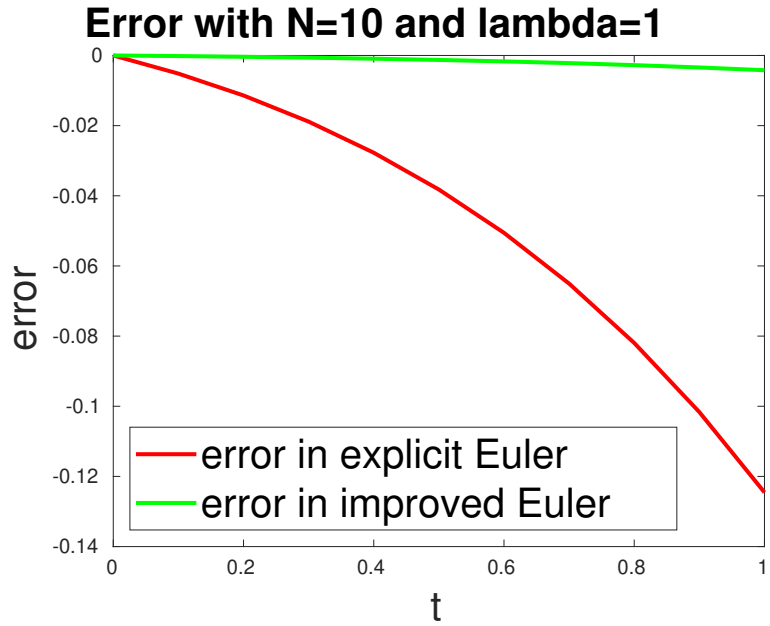
- Improved Euler:

$$U_{n+1} = \left(1 + \lambda \Delta t + \frac{1}{2}(\lambda \Delta t)^2\right) U_n\,.$$

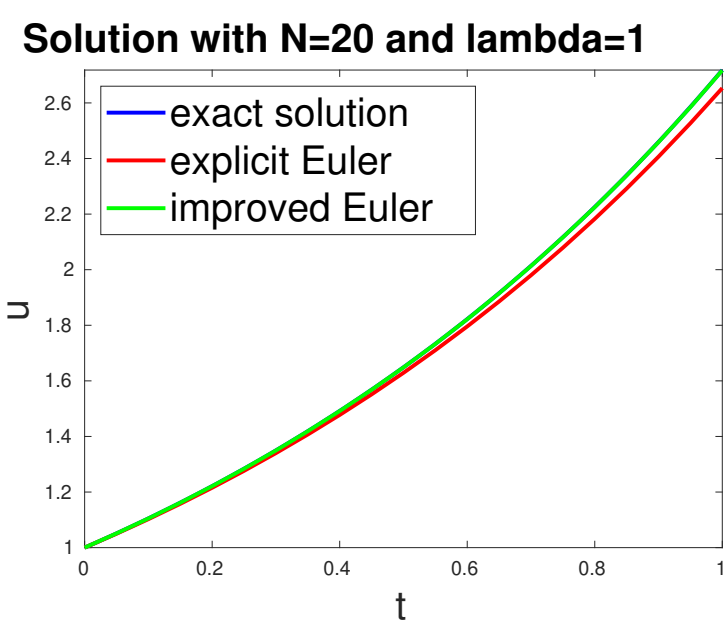
for $n = 0, 1, \ldots, N-1$ and with $U_0 = 1$.

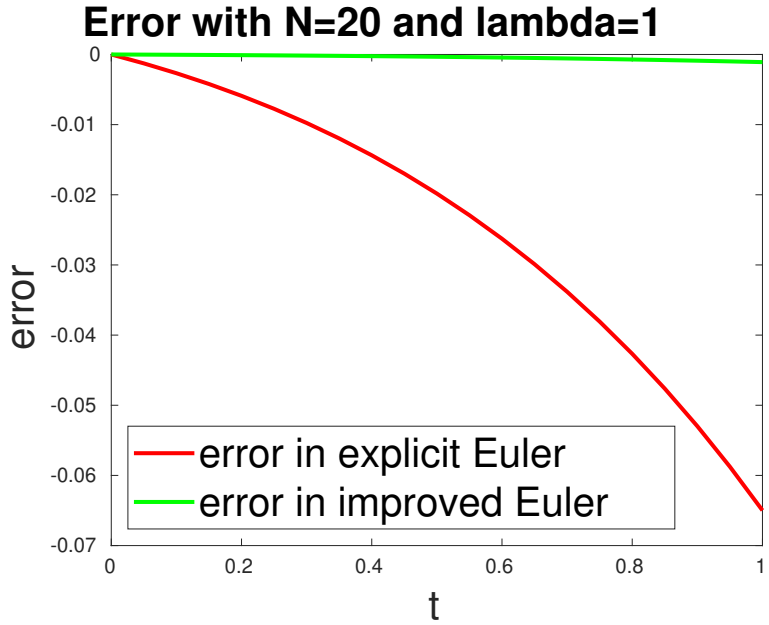
Example 1 Results



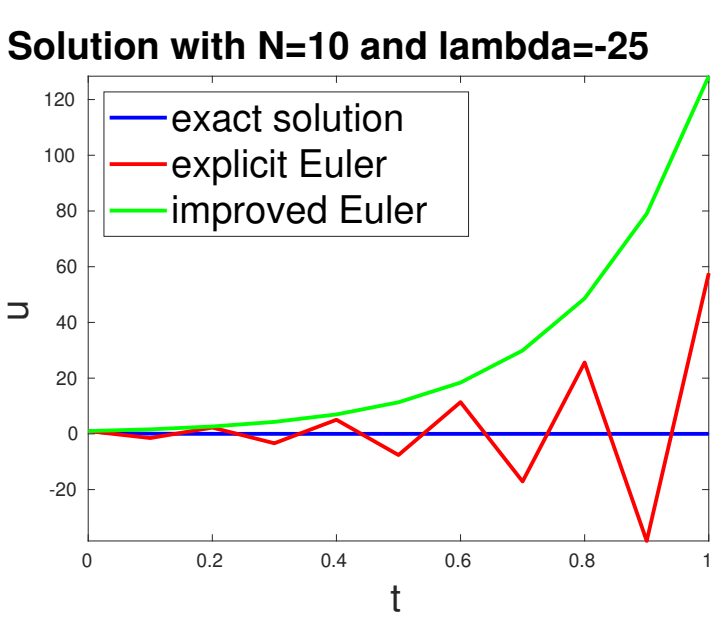


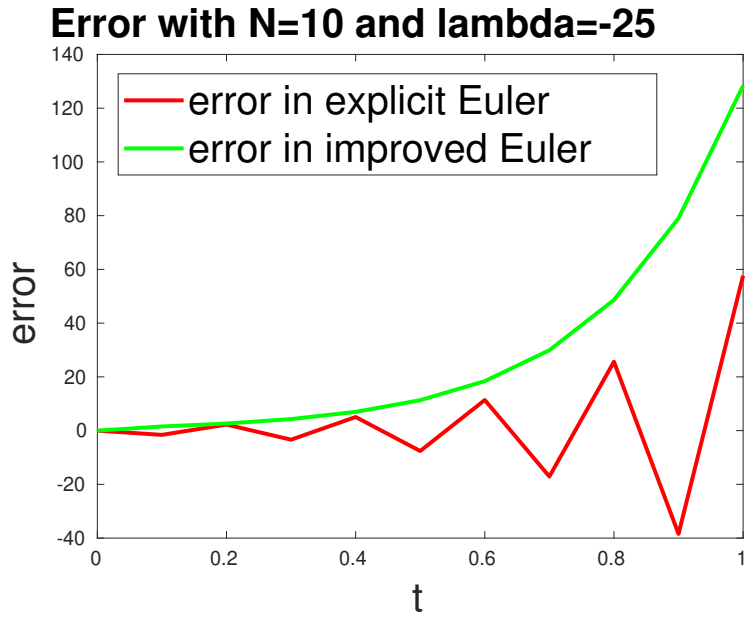
Example 1 Results



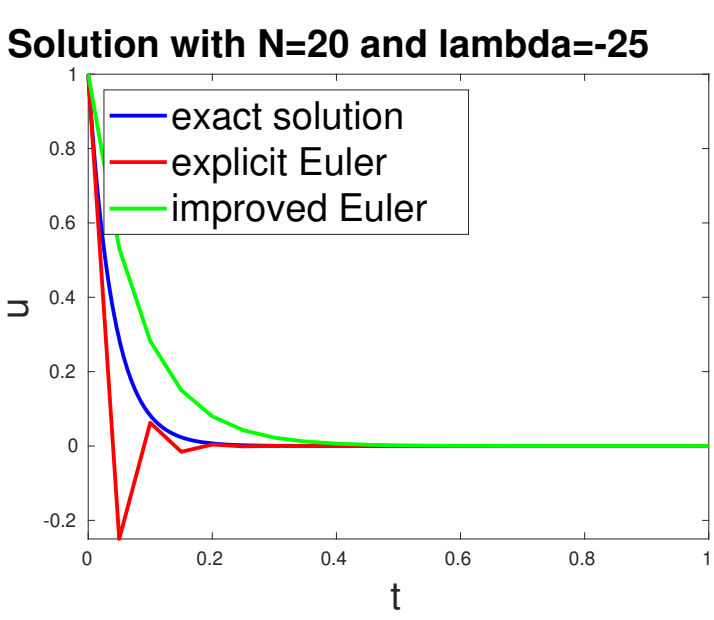


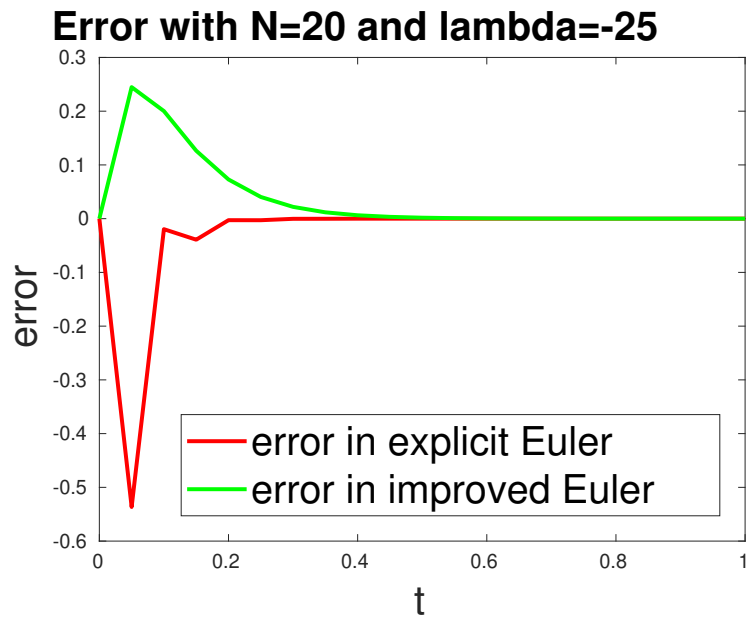
Example 1 Results





Example 1 Results





Explanation — Stability

We are solving

$$\begin{aligned} u'(t) &= \lambda u\,, \quad t > 0\,, \\ u(0) &= 1\,, \end{aligned}$$

which has exact solution $u(t) = \mathrm{e}^{\lambda t}$.

If $\lambda < 0$ the exact solution satisfies

$$\lim_{t\to\infty} u(t) = 0\,.$$

We would like our numerical scheme to mimic this behaviour.

Explanation — Stability

The explicit Euler scheme for this problem is

$$U_{n+1} \;=\; (1+\lambda\Delta t)U_n\,,$$

for $n = 0, 1, \ldots, N-1$ and with $U_0 = 1$.

Thus the explicit Euler solution is

$$U_n \;=\; (1+\lambda\Delta t)^n\,.$$

For $\lambda < 0$ we require

$$\lim_{n\to\infty} U_n \;=\; 0\,.$$

This holds if

$$-1 \;<\; 1+\lambda\Delta t \;<\; 1\,,$$

or equivalently if

$$\Delta t \;<\; \frac{2}{|\lambda|}\,.$$

Explanation — Stability

We can perform a similar analysis for the improved Euler scheme.

Improved Euler:

$$U_{n+1} \;=\; \left(1 + \lambda\Delta t + \frac{1}{2}(\lambda\Delta t)^2\right) U_n \,,$$

for $n = 0, 1, \ldots, N-1$ and with $U_0 = 1$. Then

$$U_n \;=\; \left(1 + \lambda\Delta t + \frac{1}{2}(\lambda\Delta t)^2\right)^n$$

so that

$$\lim_{n\to\infty} U_n \;=\; 0 \,,$$

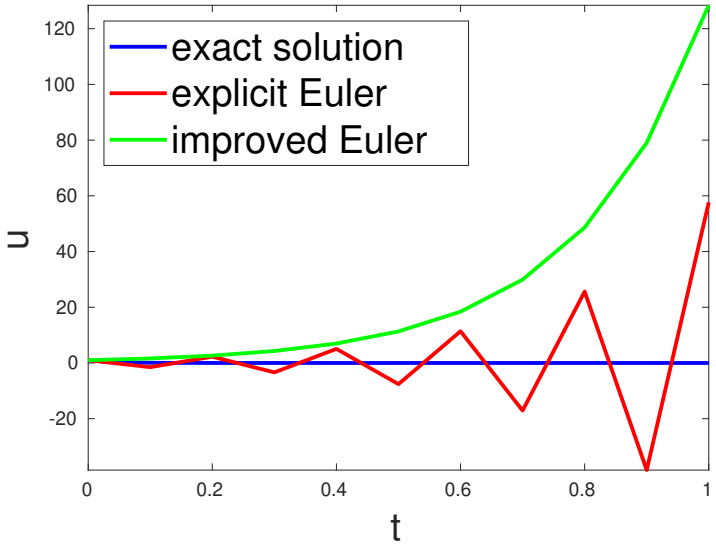
if

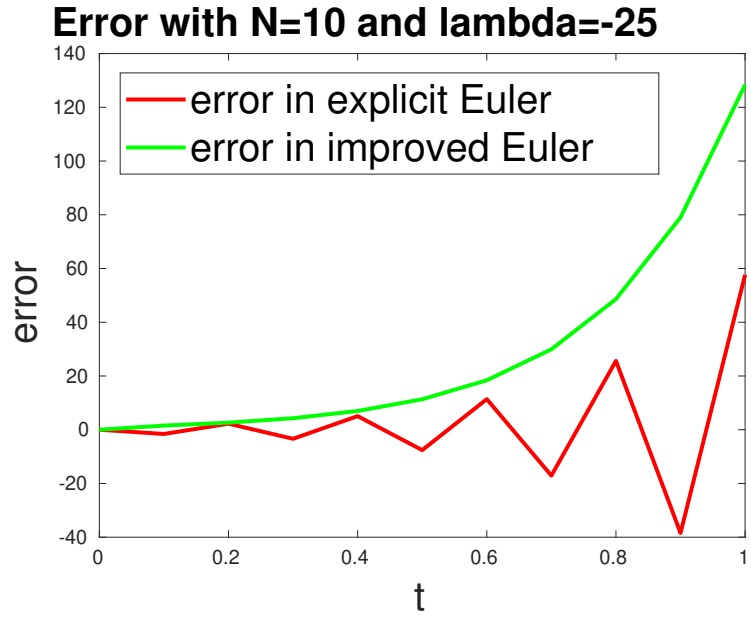
$$-1 \;<\; 1 + \lambda\Delta t + \frac{1}{2}(\lambda\Delta t)^2 \;<\; 1 \,,$$

or equivalently if

$$\Delta t \;<\; \frac{2}{|\lambda|} \,.$$

Example 1 Results Revisited



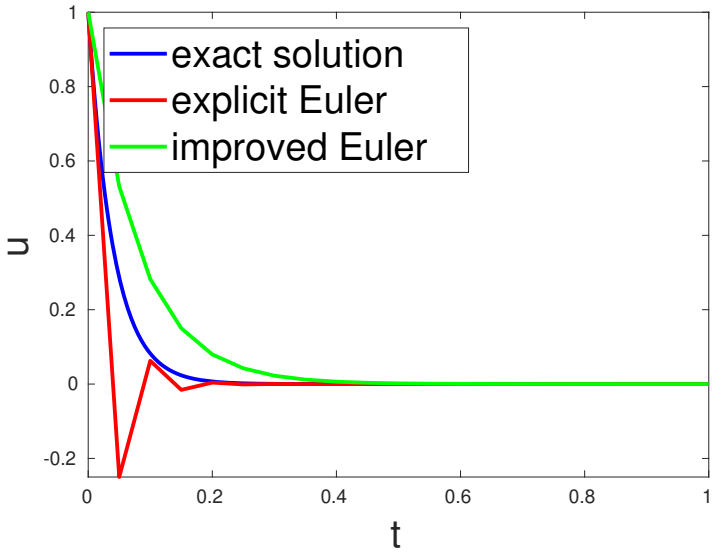


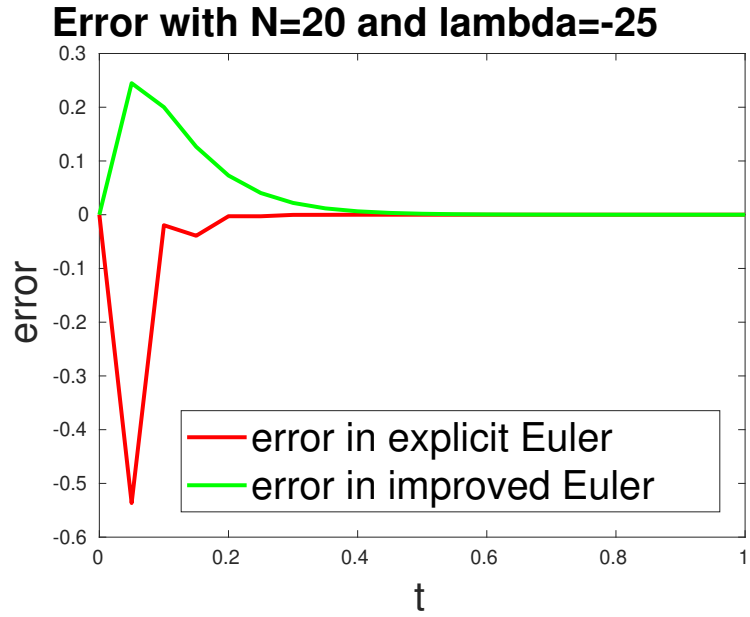
Here $\lambda = -25$ and $\Delta t = 0.1$ so $\Delta t > 2/|\lambda|$.

For explicit Euler we have $U_n = (1 + \lambda \Delta t)^n = (-1.5)^n$.

For improved Euler we have
$U_n = (1 + \lambda \Delta t + (\lambda \Delta t)^2/2)^n = 1.625^n$.

Example 1 Results Revisited





Here $\lambda = -25$ and $\Delta t = 0.05$ so $\Delta t < 2/|\lambda|$.

For explicit Euler we have $U_n = (1 + \lambda \Delta t)^n = (-0.25)^n$.

For improved Euler we have
$U_n = (1 + \lambda \Delta t + (\lambda \Delta t)^2/2)^n = 0.53125^n$.

The exact solution is $u(t_n) = \mathrm{e}^{\lambda t_n} = \mathrm{e}^{\lambda n \Delta t} = \left(\mathrm{e}^{\lambda \Delta t}\right)^n \approx 0.2865^n$.

Interval of Absolute Stability

The interval of absolute stability of a numerical scheme is the interval of values of $\lambda \Delta t$ such that, when applied to the problem

$$\begin{aligned} u'(t) &= \lambda u \,, \quad t > 0 \,, \\ u(0) &= 1 \,, \end{aligned}$$

the numerical solutions satisfy

$$\lim_{n \to \infty} U_n = 0 \,.$$

Thus the interval of absolute stability of both the explicit Euler and improved Euler schemes is $\lambda \Delta t \in (-2, 0)$.

Interval of Absolute Stability

It can be shown that for the RK4 scheme, the numerical solution satisfies

$$U_{n+1} = \left(1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24}\right) U_n \,,$$

where $z = \lambda \Delta t$.

The Python command `polyroots([1,1,1/2,1/6,1/24])` (from the numpy.polynomial.polynomial module) reveals only complex roots so

$$1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24} > 0 \,.$$

To find when

$$1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24} < 1 \,,$$

we can use `polyroots([0,1,1/2,1/6,1/24])` which gives real roots at $z = 0$ and $z = -2.7853$ (as well as two complex roots). Thus the RK4 scheme is stable for $\lambda \Delta t \in (-2.7853, 0)$.

Aside: Interval of Absolute Stability for θ-Method

We have been talking about explicit Runge-Kutta schemes but the idea of stability also holds for the θ-method discussed last time.

Recall that the θ-method is

$$\frac{U_{n+1} - U_n}{\Delta t} = \theta f(t_{n+1}, U_{n+1}) + (1-\theta) f(t_n, U_n) \,,$$

so that for the problem

$$\begin{aligned} u'(t) &= \lambda u \,, \quad t > 0 \,, \\ u(0) &= 1 \,, \end{aligned}$$

we have

$$\frac{U_{n+1} - U_n}{\Delta t} = \theta \lambda U_{n+1} + (1-\theta) \lambda U_n \,.$$

Aside: Interval of Absolute Stability for θ-Method

Re-arranging gives

$$U_{n+1} \quad = \quad \frac{1+(1-\theta)\lambda\Delta t}{1-\theta\lambda\Delta t} U_n \, .$$

Thus the interval of absolute stability is the range of values of $\lambda\Delta t$ for which

$$-1 \quad < \quad \frac{1+(1-\theta)\lambda\Delta t}{1-\theta\lambda\Delta t} \quad < \quad 1 \, .$$

After some algebra we find that this is satisfied if

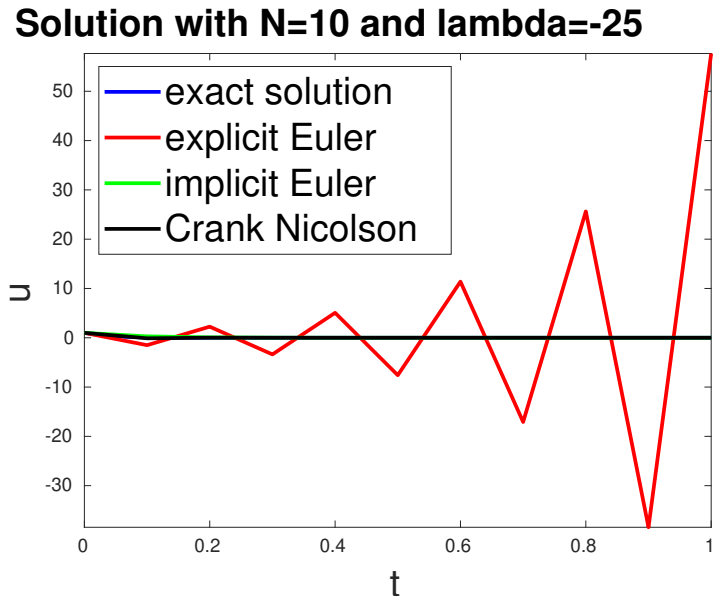
$$2+(1-2\theta)\lambda\Delta t \quad > \quad 0 \, .$$

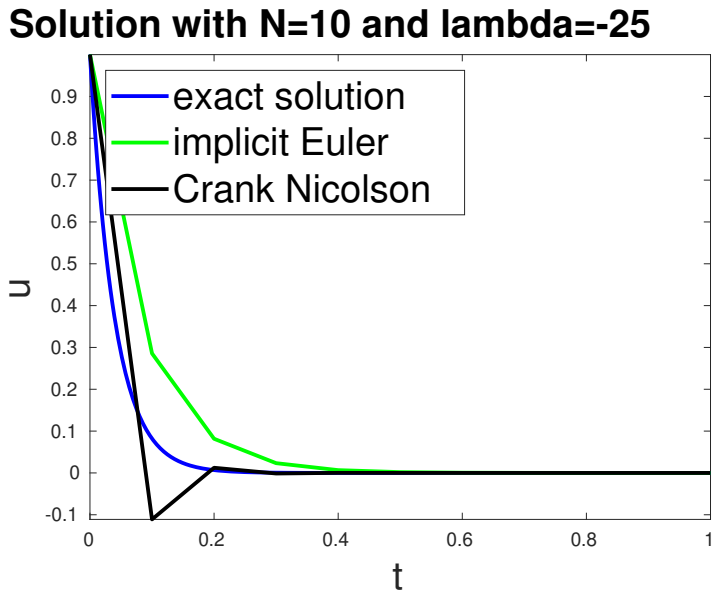
Two cases:

- If $\theta \in [1/2, 1]$, this holds for any negative value of $\lambda\Delta t$.
- If $\theta \in [0, 1/2)$, then we require

$$\lambda\Delta t \quad \in \quad \left(-\frac{2}{1-2\theta}, 0\right) \, .$$

Aside: Interval of Absolute Stability for θ-Method





Numerical Verification of Convergence Rates

We have already shown that the improved Euler and modified Euler schemes are second order. A (tedious) truncation error analysis would show that the RK4 scheme is fourth order. We verify this numerically as follows:

If a method has order p then we know that

$$|U_n - u(t_n)| \quad \lesssim \quad C\Delta t^p \ .$$

In particular

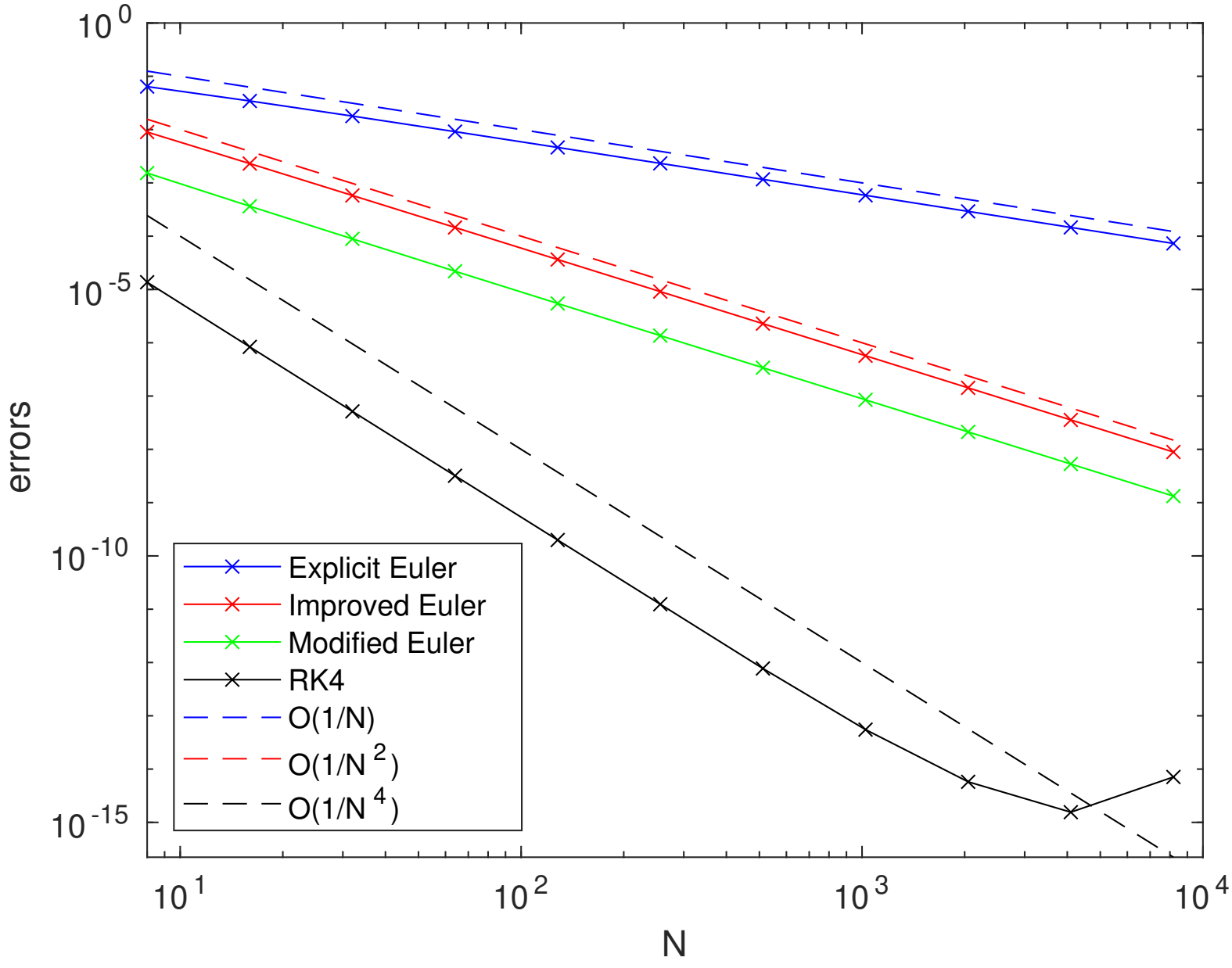
$$|U_N - u(T)| \quad \lesssim \quad C\Delta t^p \ ,$$

where $N\Delta t = T$. Taking logs

$$\log|U_N - u(T)| \quad \lesssim \quad p\log(\Delta t) + \log C \ .$$

Thus on a log-log scale the errors lie on a straight line with gradient p (or gradient $-p$ if x-axis is N rather than Δt).

Numerical Verification of Convergence Rates



Next Time ...

Adaptive Runge-Kutta schemes which do not use uniform time steps, and linear multistep methods.