the whole interval.

While one can derive the global error for a method, the local truncation error tends to be relatively easy to get. In general, the order of error for the global error is one order lower than the local truncation error.

From the concept of global error, we can derive that the forward Euler's method is a first order method in terms of global error: $E_n = O(h)$. We can see this conclusion from the *error analysis*.

As for each n, we have the local error $\tau_{k+1} = y''(\xi_k)h^2/2$ derived above. **Roughly speaking**, we sum up the local errors for $k = 1, \ldots, n$, we get the accumulated error upto time t_n as

$$\sum_{k=1}^{n} |\tau_k| \leq \sum_{k=1}^{n} y''(\xi_k)h^2 \leq nMh^2 \leq (b-a)Mh$$

where M is the maximum value of $y''(t)$ on $[a, b]$. We also note that $n \approx (b-a)/h$, so the **global error** is $O(h)$, which is first order accurate.

2.5 Convergence, consistency and stability

Definition: a numerical method is called **consistent** if the LTE at each step is at least $o(h)$ as $h \to 0$. This means we need

$$\lim_{h \to 0} \max_{0 \leq n \leq N} \frac{\tau_n}{h} = 0.$$

Definition: A numerical method is said to be **convergent** if the global error E goes to zero as the stepsize h goes to zero. (This means all E_n go to zero as $h \to 0$.)

Example: for IVP $y' = f(t, y), y(0) = y_0$, what is the local order of error of the following explicit two-step method, and is it consistent?

$$\phi_{n+1} + 9\phi_n - 10\phi_{n-1} = \frac{h}{2}(13f(t_n, \phi_n) + 9f(t_{n-1}, \phi_{n-1})).$$

You can derive it, and the error is 2 and it is consistent method.

Stability: A consistent method might not be convergent because of the way numerical errors accumulate over time steps. For example, consider the method above multi-step method. It is consistent with order 2. However, it is unstable. An easy way to see it is to apply it to solving the IVP $y' = 0, y(0) = a$. To initiate the method, we need two initial values $u(0) = u_0$ and $u(h) = u_1$. Since the right-hand side is zero, the method becomes the following linear recurrent relationship:

$$\phi_{n+1} + 9\phi_n - 10\phi_{n-1} = 0$$

The general solution to above is $\phi_n = Ar_1^n + Br_2^n$ where r_1 and r_2 are the roots to the characteristic equation $r^2 + 9r - 10 = 0$, i.e., $r_1 = 1, r_2 = -10$. In order to obtain the constant solution $\phi_n = a$, we need $\phi_0 = \phi_1 = a$. Then $A = a$ and $B = 0$. If either of these values ϕ_0 or ϕ_1, will be slightly perturbed, the

coefficient B will be nonzero and hence the solution will blow up. Note that the smaller the time step h will be, the more the solution will blow up over a fixed interval of time.

This example shows that besides requiring that the errors committed at each time step be small, they also need to accumulate stably.

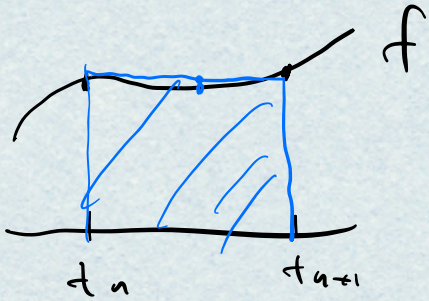
Indeed, we will need the so called zero-stability condition to ensure the stability of the method: for IVP $y' = f(t, y), y(t_0) = y_0$, if $\{\phi_n\}$ and $\{\psi_n\}$ are the numerical solutions to the IVP with the initial conditions ϕ_0 and ψ_0, if it holds that

$$\max_{0\le n\le N} |\phi_n - \psi_n| \le C \max_{0\le n\le N} |\phi_0 - \psi_0|,$$

where C is a constant independent of h (so independent of n), then the method is said to be zero-stable.

Convergence Theorem: A typical numerical method is convergent if it is zero-stable and consistent.

numerical solution ϕ_n will approach real sol $y(t)$



Heun's Method

Heun's method is a two-stage method, and it is also known as the improved Euler's method. We can still use the integral in (2) $y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))dt$ to start. But now we want to use the trapezoidal rule to approximate the integral, and this leads to $y(t_{n+1}) \approx y(t_n) + \frac{h}{2}(f(t_n, y(t_n)) + f(t_{n+1}, y(t_{n+1}))$. The term $f(t_{n+1}, y(t_{n+1}))$ is unknown, but we can approximate it using the forward Euler's method. If we use ϕ_n to denote the approximate solution of the IVP at t_n, then the **Heun's method** has formula

$$\phi_{n+1} = \phi_n + \frac{h}{2}(f(t_n, \phi_n) + f(t_{n+1}, \phi_n + hf(t_n, \phi_n))). \tag{6}$$

The Heun method is two stage method, and it is *explicit in time* method. It is also called midpoint method. We note it can be broken down to two steps: a **predictor** step and a **corrector** step. The predictor step is

$$\phi^*_{n+1} = \phi_n + hf(t_n, \phi_n),$$

and the corrector step is

$$\phi_{n+1} = \phi_n + \frac{h}{2}(f(t_n, \phi_n) + f(t_{n+1}, \phi^*_{n+1})).$$

The Heun method has locally 3rd order accurate, as the trapezoidal rule is used in approximating the integral. So the method is 2nd order accurate in

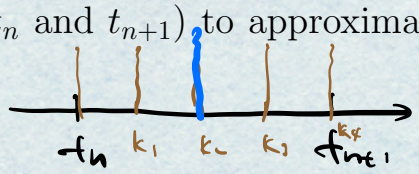
terms of global error. This is also called improved Euler's method.

3 Runge-Kutta Methods

3.1 Introduction

The Runge-Kutta methods are a family of methods that are based on the idea of using several stages to approximate the integral in (2). Indeed both Euler's method and Heun's method are special cases of the Runge-Kutta methods.

The Runge-Kutta method for advancing from t_n to t_{n+1} is to use multiple stages (f at intermiediate time points between t_n and t_{n+1}) to approximate the integral in (2), and it has the general form



$$y(t_{n+1}) \approx y(t_n) + h \sum_{i=1}^{s} b_i f_i, \tag{7}$$

where f_i are the slopes (f) at the stages $t_n + c_i h$, and b_i are the weights. The slopes f_i are computed as

$$f_i = f(t_n + c_i h, y(t_n) + h \sum_{j=1}^{i-1} a_{ij} f_j),$$

where a_{ij} are the coefficients (note to make sure k_i are at the right estimates, we need to ensure $c_i = \sum_{j=1}^{i-1} a_{ij}$). The Runge-Kutta method is explicit if the coefficients a_{ij} are zero for $i \geq j$, and it is implicit if the coefficients a_{ij} are non-zero for $i \geq j$. It is important to note that the sumation in (7) is to approximate the integral in (2): $y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f(t, y(t))$.

To approximate the integral, we have the goal (the Runge-Kutta method) to use function evaluations

$$f(t_h a, y + hb)$$

to replace the derivatives $y_n'', y'''n, \cdots$ and get a method with the desired order of accuracy. We note the number of stages as explained above is the number of function evaluations in the formula. Why? Let's see.

Recall that Euler's method was derived by expanding y_{n+1} in a Taylor series:

$$y_{n+1} = y_n + hy'(t_n) + O(h^2) \Longrightarrow y_{n+1} = y_n + hf(t_n, y_n) + O(h^2).$$

Since $y'(t) = f(t, y(t))$, we can derive a one step method (involving function evaluations only at t_N) by using

$$y_{n+1} = y_n + hy'(t_n) + \cdots \frac{h^p}{p!} y^{(p)}(t_n) + O(h^{p+1}).$$

We want to point out

$$y'' = \frac{d}{dt}(f(t, y)) = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial y} y' = f_t + f_y f.$$

We can see if we want to compute higher derivatives for y, there will be some

ugly formula to compute. And if you do, then you can derive the **Taylor series method** for the ODE.

3.2 Simple Runge-Kutta Methods

Now we see that $y' = f(t_n, y_n)$ are reasonable to compute, but $y'' = f_t + f_y f$ are not. The Runge-Kutta methods are designed to approximate the integral in (2) by using interpolant of $f(t, y(t))$ at several stages between t_n and t_{n+1}, and the interpolant is constructed by using the slopes f_i at these stages. The simplest Runge-Kutta method is the second order Runge-Kutta method, which is also known as the midpoint method. The method can be illustrated as the following

$$\begin{cases} f_1 = f(t_n, y_n), \quad f_2 = f(t_n + ch, y_n + haf_1), \\ y_{n+1} = y_n + h(b_1 f_1 + b_2 f_2). \end{cases}$$

with constants b_1, b_2, a, c to be determined. We note the method is second order accurate and the term $h(b_1 f_1 + b_2 f_2)$ is the approximation of the integral in $y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f(t, y(t))dt$.

The goal is to choose the constants so that

$$LHS = \frac{1}{h}(y_{n+1} - y_n) = b_1 f_1 + b_2 f_2 + \tau_{n+1} = RHS, \ \tau_{n+1} \sim O(h^2). \qquad (8)$$

The strategy is to expand in a Taylor series around t_n and y_n to get the terms of f and its derivatives and **match terms**!

For LHS

$$\begin{aligned}\frac{1}{h}(y_{n+1}-y_n) =& y'_n + \frac{h}{2}y''_n + O(h^2)\\ =& f + \frac{h}{2}(f_t + f_y f) + O(h^2)\end{aligned}$$

For the RHS, first expand f_2 using a Taylor series around (t_n, y_n):

$$\begin{aligned}f_2 =& f_n + (ch)\frac{\partial f(t_n,y_n)}{\partial t} + (haf_1)\frac{\partial f(t_n,y_n)}{\partial y} + O(h)\\ =& f + chf_t + haff_y + O(h^2)\end{aligned}$$

Then we can get the RHS:

$$RHS = b_1 f + b_2(f + chf_t + hbff_y) + \tau_{n+1} + O(h^2)$$

Both LHS and RHS of equation (8) are now available and we can match the terms to get the coefficients b_1, b_2:

$$\begin{aligned}b_1 + b_2 =& 1,\\ cb_2 = ab_2 =& \frac{1}{2}\end{aligned}$$

If we choose $b_2 = \theta$, then $b_1 = 1-\theta$, $a = c = 1/(2\theta)$, so for any $\theta \neq 0$, we

have

$$\begin{aligned} f_1 =& f(t_n, y_n), \\ f_2 =& f(t_n + h, y_n + h\theta f_1), \\ y_{n+1} =& y_n + h((1-\theta)f_1 + \theta f_2). \end{aligned}$$

The most popular choice is $\theta = 1/2$, and we can get the coefficients $b_1 = \frac{1}{2}$, $b_2 = \frac{1}{2}$, $a = 1$, $c = \frac{1}{2}$.

3.3 Typical Runge-Kutta Methods

We note the general form of the Runge-Kutta method in (7), and we can see that the method is determined by the coefficients a_{ij}, b_i, c_i. The most popular Runge-Kutta methods are the second order Runge-Kutta method (midpoint method), the fourth order Runge-Kutta method (RK4), and the fifth order Runge-Kutta method (RK5).

$$\begin{aligned} f_1 =& f(t_n, \phi_n), \\ f_2 =& f(t_n + c_2 h, \phi_n + h a_{21} f_1), \\ f_3 =& f(t_n + c_3 h, \phi_n + h a_{31} f_1 + h a_{32} f_2), \\ & \vdots \\ f_m =& f(t_n + c_m h, \phi_n + h a_{m1} f_1 + h a_{m2} f_2 + \cdots + h a_{m,m-1} f_{m-1}), \\ \phi_{n+1} =& \phi_n + h(b_1 f_1 + b_2 f_2 + \cdots + b_m f_m). \end{aligned}$$