Numerical methods for solving IVPs for ODEs

Consider the IVP-ODE $y' = f(t, y),\ \ y(t_0) = y_0$. Most numerical methods approximate the solution of an IVP for an ODE by generating a sequence $y_1, y_2, y_3, \cdots$ of approximations to $y(t)$ at a set of points $t_1, t_2, t_3, \cdots$, respectively, such that $t_{i-1} < t_i$. That is, $y_1 \approx y(t_1)$, $y_2 \approx y(t_2)$, $y_3 \approx y(t_3)$, etc. If we wish to approximate $y(t)$ at intermediate points, we can fit an interpolant through $(t_0, y(t_0))$, $(t_1, y(t_1))$, $(t_2, y(t_2))$, $(t_3, y(t_3))$, etc. Other methods may compute other quantities, such as coefficients of a spline representation of approximations to $y(t)$ at $t_1, t_2, t_3, \cdots$, and from these compute approximations $y_1, y_2, y_3, \cdots$

A simple way to proceed from t_i to t_{i+1} is to use values of the approximation to y at one or more previous points, as well as f.

As far as the use of one or more previous values of the approximation to y, we have the following classification of methods:
One-step methods use y_i to construct y_{i+1}.
Multi-step methods use $y_i, y_{i-1}, \cdots$ to construct y_{i+1}.

As will be seen later, another classification of methods will arise. As far as the use of f on previous and/or the current step, we have the following classification of methods:
Explicit methods use f at previous step(s), for which the approximation to y is already computed.
Implicit methods use f at the current step (i.e. $f(t_{i+1}, y_{i+1})$, "*before*" y_{i+1} is computed. They may also use past values of f.

(Forward) Euler's method

Consider the IVP-ODE $y' = f(t, y),\ \ y(t_0) = y_0$, and Taylor's expansion about t_i with stepsize $h_i = t_{i+1} - t_i$

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

Assume y'' exists and is bounded and that h_i is small. Then the remainder $\frac{h_i^2}{2!} y''(\xi_i)$ is small. If we discard the remainder and substitute approximations $y_{i+1} \approx y(t_{i+1})$ and $y_i \approx y(t_i)$, we have

$$y_{i+1} = y_i + h_i y'(t_i),\ \ i = 0, 1, \cdots$$

or, if we also substitute $f(t_i, y_i) \approx f(t_i, y(t_i)) = y'(t_i)$

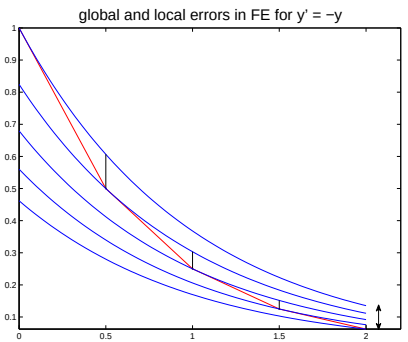
$$y_{i+1} = y_i + h_i f(t_i, y_i),\ \ i = 0, 1, \cdots \qquad \text{(FE)}$$

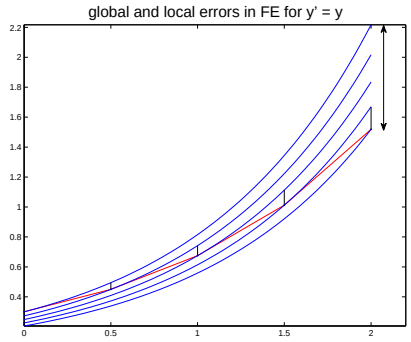
which gives *(forward) Euler's (FE) method*. That is, using y_i and $f(t_i, y_i)$, we construct y_{i+1}. Geometrically, starting at y_i, we proceed to the direction of the tangent $f = y'$ at (t_i, y_i) for a stepsize h_i.

Consider the first few steps. Starting at $y_0 = y(t_0)$, we proceed to the direction of the tangent $f = y'$ at (t_0, y_0) for a stepsize h_0. When we arrive at $y_1 \approx y(t_1)$, we are no longer on the original curve, but on a nearby curve (of the same family of solutions of the ODE $y' = f(t, y)$).

(Forward) Euler's method

For the second step, starting at y_1, we proceed to the direction of the tangent $f = y'$ at (t_1, y_1) for a stepsize h_1. Note that we start with some error and take the tangent of the nearby curve. When we arrive at $y_2 \approx y(t_2)$, we are no longer on the second curve, but on another nearby curve.





Note:
Global error indicated by double arrows, local errors by vertical lines.

(Forward) Euler's method

Thus, although we start with an exact initial value $y_0 = y(t_0)$, subsequent steps have some error, i.e. $y_i \neq y(t_i)$, in general. In the first step the source of the error is the discarded term $\frac{h_0^2}{2!} y''(\xi_0)$ in Taylor's expansion. In the second (and any subsequent) step there are two sources of errors: (a) the discarded term $\frac{h_i^2}{2!} y''(\xi_i)$ in Taylor's expansion, and (b) the accumulation of previous errors which forces us to start the step on the "wrong" curve.

Stability and accuracy of FE

To study the error(s) in FE, assume that all computations are done in exact arithmetic (so that no round-off errors occur), and consider again

$$y(t_{i+1}) = y(t_i) + h_i f(t_i, y(t_i)) + \frac{h_i^2}{2!} y''(\xi_i),\ \ \xi_i \in (t_i, t_{i+1}) \qquad (1)$$

and the FE method

$$y_{i+1} = y_i + h_i f(t_i, y_i). \qquad (2)$$

Consider subtracting (2) from (1).

(Forward) Euler's method -- stability and accuracy

Subtracting (2) from (1) we get

$$y(t_{i+1}) - y_{i+1} = y(t_i) - y_i + h_i(f(t_i, y(t_i)) - f(t_i, y_i)) + \frac{h_i^2}{2!} y''(\xi_i). \quad (3)$$

The quantity $y(t_{i+1}) - y_{i+1}$ is called the **global truncation error (GTE) at** t_{i+1} (from t_0 to t_{i+1}).

The quantity $\frac{h_i^2}{2!} y''(\xi_i)$ is called the **local truncation error (LTE) from** t_i **to** t_{i+1}, and is equivalent to the GTE, *had we started the step with the exact value*, i.e. if $y_i = y(t_i)$.

In general, $y_i \neq y(t_i)$. Regarding $f(t_i, y)$ as a function of y and applying the MVT, we get

$$\frac{f(t_i, y(t_i)) - f(t_i, y_i)}{y(t_i) - y_i} = \frac{\partial f}{\partial y}(t_i, \tau_i), \ \ \tau_i \in \text{ospr}\{y_i, y(t_i)\} \quad (4)$$

Using (4), relation (3) becomes

$$y(t_{i+1}) - y_{i+1} = y(t_i) - y_i + h_i(y(t_i) - y_i)\frac{\partial f}{\partial y}(t_i, \tau_i) + \frac{h_i^2}{2!} y''(\xi_i)$$

or, equivalently,

$$y(t_{i+1}) - y_{i+1} = (y(t_i) - y_i)(1 + h_i \frac{\partial f}{\partial y}(t_i, \tau_i)) + \frac{h_i^2}{2!} y''(\xi_i). \quad (5)$$

(Forward) Euler's method -- stability and accuracy

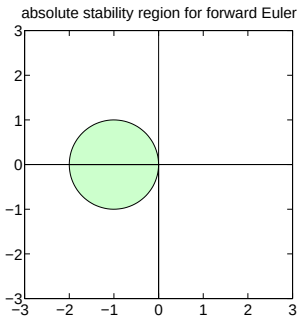
Relation (5) indicates that the GTE at t_{i+1} equals the GTE at t_i times an amplification factor $\text{AF}_i = 1 + h_i \frac{\partial f}{\partial y}(t_i, \tau_i)$, plus the LTE from t_i to t_{i+1}.

(GTE at t_{i+1}) = (GTE at t_i) × (AF from t_i to t_{i+1}) + (LTE from t_i to t_{i+1})

Note that $\frac{\partial f}{\partial y}(t_i, \tau_i)$ is the Jacobian at (t_i, τ_i). Denoting $J_i = \frac{\partial f}{\partial y}(t_i, \tau_i)$, we have $\text{AF}_i = 1 + h_i J_i$.

If $|1 + h_i J_i| < 1$, past errors are not magnified and the numerical method (FE) is called **stable**.
If $|1 + h_i J_i| > 1$, past errors are magnified and the numerical method (FE) is called **unstable**.

The relation $|1 + h_i J_i| < 1$ says that $h_i J_i$ must lie inside a circle in the complex plane of radius 1 centered at −1. Thus, the region (called **region of absolute stability**) where $h_i J_i$ must lie in order to maintain stability of FE is finite.



(Forward) Euler's method -- stability and accuracy

Consider the case that J_i is real. The region of absolute stability of FE is the interval $(-2, 0)$.

If $J_i > 0$, the FE method is unstable (if $h_i > 0$), i.e., if the ODE is unstable, FE is unstable too (as long as we proceed forward in time).

If $J_i < 0$, the FE method is stable, iff

$$-1 < 1 + h_i J_i < 1 \Rightarrow -2 < h_i J_i < 0 \Rightarrow 2 > -h_i J_i > 0 \Rightarrow 0 < h_i < -\frac{2}{J_i}.$$

Thus, for FE to be stable, h_i is restricted to be less than $-\frac{2}{J_i}$ (and positive).

A smart implementation of FE would pick a stepsize h_i small enough to satisfy the stability restriction $h_i < -\frac{2}{J_i}$ at each step.

If $J_i \ll 0$, then h_i has to be very small for FE to be stable, i.e. if the ODE is stiff, the stepsize restriction is very stringent. Thus, the FE method for a stiff ODE will take many steps to reach the final point of the interval of interest.

(Forward) Euler's method -- stability and accuracy

The magnitude of the stepsize h_i can also be used to control the LTE from t_i to t_{i+1}. Assume we wish to have |LTE from t_i to t_{i+1}| ≤ *tol*, i.e. $\frac{h_i^2}{2!} y''(\xi_i) \leq tol$. Then we should pick $h_i \leq \sqrt{2\frac{tol}{|y''(\xi_i)|}}$. Since neither ξ_i is known, nor y'' is readily available, we usually pick $h_i \leq \sqrt{2\frac{tol\ h_{i-1}}{|y'_i - y'_{i-1}|}}$, since $y''(\xi_i) \approx \frac{y'_{i+1} - y'_i}{h_i} \approx \frac{y'_i - y'_{i-1}}{h_{i-1}}$.

By choosing h_i small in the FE method, we achieve two goals: (a) numerical stability of the method and (b) small LTEs. However, we cannot affect the stability of the ODE.

For a system of ODEs, the GTE (a vector) is multiplied at each step by the amplification factor (matrix) $\mathbf{I} + h_i J_i$, where J_i is the Jacobian matrix at (t_i, τ_i). Past errors are not magnified, if the spectral radius of $\mathbf{I} + h_i J_i$ is less than 1, i.e., $\rho(\mathbf{I} + h_i J_i) \leq 1$. This condition is satisfied if all eigenvalues of $h_i J_i$ lie inside a circle in the complex plane of radius 1 centered at −1.

(Forward) Euler's method -- stability and accuracy

For the FE method, with $h = \max_i \{h_i\}$, we have LTE = $O(h^2)$, assuming y'' exists and is bounded. We can also show that GTE = $O(h)$. That is, FE is of first order.
We say that a numerical method for ODEs is **of pth order**, when the LTE is $O(h^{p+1})$ (or when the GTE is $O(h^p)$). For a pth order method, we expect the GTE to decrease by 2^p, when h is halved.

To show that the GTE of FE is $O(h)$, consider again relation (3)

$$y(t_{i+1}) - y_{i+1} = y(t_i) - y_i + h_i(f(t_i, y(t_i)) - f(t_i, y_i)) + \frac{h_i^2}{2!} y''(\xi_i). \tag{3}$$

Assume that f satisfies a Lipschitz condition in y with Lipschitz constant L. Then,

$$|f(t_i, y(t_i)) - f(t_i, y_i)| \le L|y(t_i) - y_i| \tag{6}$$

and from (3) and (6) we have

$$|y(t_{i+1}) - y_{i+1}| \le |y(t_i) - y_i| + h_i L|y(t_i) - y_i| + \frac{h_i^2}{2!}|y''(\xi_i)|, \tag{7}$$

or |GTE at t_{i+1}| ≤ |GTE at t_i| × $(1 + h_i L)$ + |LTE from t_i to t_{i+1}|.

(Forward) Euler's method -- stability and accuracy

For simplicity, consider the case $h_i = h = \frac{t_n - t_0}{n}$. Let $A = 1 + hL$, $M = \max_{t_0 \le t \le t_n} |y''|$, $B = \frac{h^2}{2} M$, and e_{i+1} = GTE at t_{i+1}. We then have a recurrence relation (inequality) of the form

$$|e_{i+1}| \le |e_i| A + B. \tag{8}$$

It is easy to show (by induction) that

$|e_{i+1}| \le |e_0| A^{i+1} + (1 + A + A^2 + \cdots + A^i) B = |e_0| A^{i+1} + \frac{A^{i+1} - 1}{A - 1} B$. Thus

$$|e_n| \le |e_0| A^n + \frac{A^n - 1}{A - 1} B. \tag{9}$$

Taking into account that

$A^n = (1 + hL)^n \le (1 + hL + \frac{(hL)^2}{2} + \cdots)^n = (e^{hL})^n = e^{nhL} = e^{(t_n - t_0)L}$, and $A - 1 = hL$, from (9) we get

$$|e_n| \le |e_0| e^{(t_n - t_0)L} + \frac{e^{(t_n - t_0)L} - 1}{hL} B = |e_0| e^{(t_n - t_0)L} + \frac{e^{(t_n - t_0)L} - 1}{L} \frac{h}{2} M. \tag{10}$$

If we assume that we start with zero error ($e_0 = 0$), then $e_n = O(h)$.

(Forward) Euler's method -- stability and accuracy

The above discussion assumed that there are no round-off errors. In realistic computations, this is not the case.

If we assume that, at each time step, instead of computing

$$y_{i+1} = y_i + h_i f(t_i, y_i),$$

we compute

$$u_{i+1} = u_i + h_i f(t_i, u_i) + \delta_{i+1},$$

where δ_{i+1} is some round-off error, and that we start we some possibly perturbed initial condition

$$u_0 = y(t_0) + \delta_0,$$

the global (including computational errors) error at point t_n can be very large, even unbounded. More specifically, it can be shown that, if $|\delta_i| \le \delta$, for $i = 0, 1, \cdots$, then

$$|y(t_n) - u_n| \le |\delta_0| e^{(t_n - t_0)L} + \frac{e^{(t_n - t_0)L} - 1}{L} (\frac{h}{2} M + \frac{\delta}{h}). \tag{11}$$

Thus the error bound is no longer $O(h)$. In fact, due to the $\frac{\delta}{h}$ term, the error can become arbitrarily large as the stepsize decreases.

(Forward) Euler's method -- stability and accuracy

Studying the quantity $E(h) = \frac{h}{2} M + \frac{\delta}{h}$ as a function of h, we see that $E'(h) = \frac{M}{2} - \frac{\delta}{h^2}$, and thus $E(h)$ decreases if $h < \sqrt{2 \frac{\delta}{M}}$ and increases if $h > \sqrt{2 \frac{\delta}{M}}$. Therefore, $E(h)$ is minimum when $h = \sqrt{2 \frac{\delta}{M}}$.

Normally, δ is very small. The relation $h = \sqrt{2 \frac{\delta}{M}}$ may restrict h to be quite small (hence the method would have to take many steps to reach the final point in time). Fortunately, the quantity $E(h)$ decreases sharply if $h < \sqrt{2 \frac{\delta}{M}}$, while it increases mildly (almost linearly) if $h > \sqrt{2 \frac{\delta}{M}}$. Therefore, we can choose a stepsize h reasonably larger than $\sqrt{2 \frac{\delta}{M}}$, and still get reasonable results from Forward Euler's method.

Backward Euler's method -- Implicit methods

Consider the IVP-ODE $y' = f(t, y)$, $y(t_0) = y_0$, and Taylor's expansion about t_{i+1} with stepsize $-h_i = t_i - t_{i+1}$ (i.e. h_i is still $t_{i+1} - t_i$)

$$y(t_i) = y(t_{i+1}) - h_i y'(t_{i+1}) + \frac{h_i^2}{2!} y''(\rho_i), \quad \rho_i \in (t_i, t_{i+1}).$$

Assume y'' exists and is bounded and that h_i is small. Then the remainder $\frac{h_i^2}{2!} y''(\rho_i)$ is small. If we discard the remainder and substitute approximations $y_{i+1} \approx y(t_{i+1})$ and $y_i \approx y(t_i)$, we have

$$y_i = y_{i+1} - h_i y'(t_{i+1}), \quad i = 0, 1, \cdots$$

or, if we also substitute $f(t_{i+1}, y_{i+1}) \approx f(t_{i+1}, y(t_{i+1})) = y'(t_{i+1})$

$$y_i = y_{i+1} - h_i f(t_{i+1}, y_{i+1}), \quad i = 0, 1, \cdots$$

or

$$y_{i+1} = y_i + h_i f(t_{i+1}, y_{i+1}), \quad i = 0, 1, \cdots \qquad \text{(BE)}$$

which gives *backward Euler's (BE) method*. That is, using y_i and $f(t_{i+1}, y_{i+1})$, we construct y_{i+1}.

Note that y_{i+1} appears twice in the above relation, once in the left side, and once in the right side as an argument to f. Thus we cannot compute y_{i+1} by (BE) explicitly, unless f is of some very simple form, e.g. linear with respect to y. In the general case, when f is nonlinear, we need to solve for y_{i+1} *implicitly*.

Backward Euler's method -- Implicit methods

Methods computing y_{i+1} in terms of previously computed values are called **explicit**. Methods computing y_{i+1} in terms of $f(\cdot, y_{i+1})$ and possibly previously computed values are called **implicit**.

Another example (linear): Let $y' = -\lambda y$, $y(0) = 2$ (i.e. $f(t, y) = -\lambda y$), and λ is a given scalar. FE computes

$$y_{i+1} = y_i - h_i \lambda y_i.$$

Thus, for example, $y_1 = 2 - 2h_0\lambda$, which is easy to evaluate, for a chosen h_0 and a given λ.
BE computes

$$y_{i+1} = y_i - h_i \lambda y_{i+1}$$

which can be rearranged as

$$y_{i+1}(1 + h_i\lambda) = y_i \Rightarrow y_{i+1} = \frac{y_i}{1 + h_i\lambda}.$$

Thus, for example, $y_1 = \frac{2}{1 + h_0\lambda}$, which is also easy to evaluate, for a chosen h_0 and a given λ.

Backward Euler's method -- Implicit methods

The equation to solve is

$$y_{i+1} - y_i - h_i f(t_{i+1}, y_{i+1}) = 0,$$

and since f is, in general, nonlinear, the above equation is also nonlinear. We need to apply some nonlinear solver, such as Newton's method, or fixed-point iteration.

Example: Let $y' = e^y(t + 1)$, $y(0) = -2$ (i.e. $f(t, y) = e^y(t + 1)$).
FE computes

$$y_{i+1} = y_i + h_i e^{y_i}(t_i + 1)$$

Thus, for example, $y_1 = -2 + h_0 e^{-2}$, which is easy to evaluate, for a chosen h_0.
BE computes

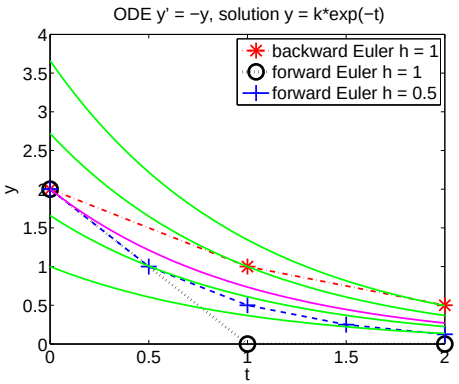
$$y_{i+1} = y_i + h_i e^{y_{i+1}}(t_{i+1} + 1).$$

This is a non-linear equation to be solved for y_{i+1}. It can take the form

$$y_{i+1} - h_i(t_{i+1} + 1)e^{y_{i+1}} - y_i = 0.$$

For example, in the first timestep, for a chosen h_0, we need to solve $y_1 - e^{y_1} h_0(h_0 + 1) + 2 = 0$ for y_1.

Backward Euler's method -- stability and accuracy



Stability and accuracy of BE

To study the error(s) in BE, assume that all computations are done in exact arithmetic (so that no round-off errors occur), and consider again

$$y(t_i) = y(t_{i+1}) - h_i f(t_{i+1}, y(t_{i+1})) + \frac{h_i^2}{2!} y''(\rho_i), \qquad (1)$$

and

$$y_i = y_{i+1} - h_i f(t_{i+1}, y_{i+1}). \qquad (2)$$

Backward Euler's method -- stability and accuracy

Subtracting (2) from (1) we get

$$y(t_i) - y_i = y(t_{i+1}) - y_{i+1} - h_i(f(t_{i+1}, y(t_{i+1})) - f(t_{i+1}, y_{i+1})) + \frac{h_i^2}{2!} y''(\rho_i). \qquad (3)$$

Regarding $f(t_{i+1}, y)$ as a function of y and applying the MVT, we get

$$\frac{f(t_{i+1}, y(t_{i+1})) - f(t_{i+1}, y_{i+1})}{y(t_{i+1}) - y_{i+1}} = \frac{\partial f}{\partial y}(t_{i+1}, \sigma_i), \quad \sigma_i \in \text{ospr}\{y_{i+1}, y(t_{i+1})\}. \qquad (4)$$

Using (4), relation (3) becomes

$$y(t_i) - y_i = y(t_{i+1}) - y_{i+1} - h_i(y(t_{i+1}) - y_{i+1}) \frac{\partial f}{\partial y}(t_{i+1}, \sigma_i) + \frac{h_i^2}{2!} y''(\rho_i)$$

or, equivalently,

$$y(t_i) - y_i = (y(t_{i+1}) - y_{i+1})(1 - h_i \frac{\partial f}{\partial y}(t_{i+1}, \sigma_i)) + \frac{h_i^2}{2!} y''(\rho_i),$$

or

$$y(t_i) - y_i = (y(t_{i+1}) - y_{i+1})(1 - h_i J_i) + \frac{h_i^2}{2!} y''(\rho_i),$$

or

$$y(t_{i+1}) - y_{i+1} = (y(t_i) - y_i)(1 - h_i J_i)^{-1} - \frac{h_i^2}{2!} y''(\rho_i)(1 - h_i J_i)^{-1}.$$

Thus

$$(\text{GTE at } t_{i+1}) = (\text{GTE at } t_i) \times (\text{AF}_i) + (\text{LTE from } t_i \text{ to } t_{i+1})\text{AF}_i$$

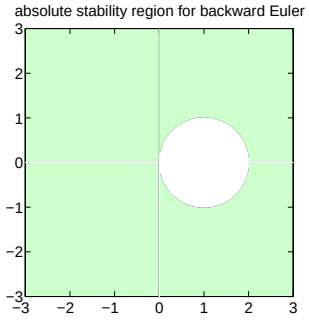
where, for BE we have $\text{AF}_i = (1 - h_i J_i)^{-1}$.

Backward Euler's method -- stability and accuracy

If $\frac{1}{|1 - h_i J_i|} < 1$, past errors are not magnified and the numerical method (BE) is **stable**.

If $\frac{1}{|1 - h_i J_i|} > 1$, past errors are magnified and the numerical method (BE) is **unstable**.

The relation $\frac{1}{|1 - h_i J_i|} < 1$ says that $h_i J_i$ must lie *outside* a circle in the complex plane of radius 1 centered at 1. Thus, the region of absolute stability of BE is infinite and includes the entire left half plane.



Methods whose region of absolute stability includes the entire left half plane are called **A-stable**.

Consider the case that J_i is real. The region of absolute stability of BE is $(-\infty, 0) \cup (2, \infty)$.

Backward Euler's method -- stability and accuracy

For stable ODEs, $J_i < 0$, and thus $\frac{1}{|1 - h_i J_i|} < 1$, for all $h_i > 0$.

Thus, for the BE method to be stable, h_i is not restricted (as long as we proceed forward in time).

However, in order to control the LTE from t_i to t_{i+1}, we still need to restrict the stepsize h, just as we did for FE, since the LTE formula for BE is similar to that of FE.

For a system of ODEs, the GTE (a vector) is multiplied at each step by the amplification factor (matrix) $(\mathbf{I} - h_i J_i)^{-1}$, where J_i is the Jacobian matrix at (t_i, τ_i). Past errors are not magnified, if the spectral radius of $(\mathbf{I} - h_i J_i)^{-1}$ is less than 1. This is satisfied if all eigenvalues of $h_i J_i$ lie outside a circle in the complex plane of radius 1 centered at 1.

For the BE method, with $h = \max_i \{h_i\}$, we have LTE = $O(h^2)$, assuming y'' exists and is bounded. We can also show that GTE = $O(h)$. Thus the BE method is of first order.

Note: We have seen that BE does not impose restrictions on the stepsize for stability. This is true for many implicit methods, though not for all. Some implicit methods may still impose some restrictions on the stepsize, however, those restrictions are often quite milder than the restrictions imposed by explicit methods.

Explicit versus implicit

Explicit	Implicit
explicit relation giving y_{i+1} in terms of previous values	implicit relation giving y_{i+1} in terms of $f(\cdot, y_{i+1})$ and possibly previous values
does not require solution of a nonlinear equation	requires solution of a nonlinear equation
for a system of ODEs does not require solution of a nonlinear system of equations	for a system of ODEs requires solution of a nonlinear system of equations
requires h_i to be small to preserve stability	often unconditionally stable or mild restrictions on h_i
lots of cheap steps	few expensive steps
tough restriction for stiff ODEs	preferred for stiff ODEs

The trapezoid method

Consider the IVP-ODE $y' = f(t, y)$, $y(t_0) = y_0$, and the two Taylor's expansions about t_i and t_{i+1} with stepsizes $h_i = t_{i+1} - t_i$ and $-h_i$, respectively

$$y(t_{i+1}) = y(t_i) + h_i y'(t_i) + \frac{h_i^2}{2!} y''(\xi_i), \ \xi_i \in (t_i, t_{i+1}),$$

and

$$y(t_i) = y(t_{i+1}) - h_i y'(t_{i+1}) + \frac{h_i^2}{2!} y''(\rho_i), \ \rho_i \in (t_i, t_{i+1}).$$

For convenience, we re-write the latter as

$$y(t_{i+1}) = y(t_i) + h_i y'(t_{i+1}) - \frac{h_i^2}{2!} y''(\rho_i), \ \rho_i \in (t_i, t_{i+1}).$$

Assume y'' exists and is bounded and that h_i is small. Discarding the remainders and substituting approximations $y_{i+1} \approx y(t_{i+1})$, $y_i \approx y(t_i)$, $f(t_i, y_i) \approx f(t_i, y(t_i)) = y'(t_i)$ and $f(t_{i+1}, y_{i+1}) \approx f(t_{i+1}, y(t_{i+1})) = y'(t_{i+1})$ we get

$$y_{i+1} = y_i + h_i f(t_i, y_i) \tag{31}$$

$$y_{i+1} = y_i + h_i f(t_{i+1}, y_{i+1}) \tag{32}$$

Adding (averaging) (31) and (32) we have

$$y_{i+1} = y_i + \frac{h_i}{2}(f(t_i, y_i) + f(t_{i+1}, y_{i+1})), \ i = 0, 1, \cdots \tag{TR}$$

The trapezoid method

This is the *trapezoid method*, the term being familiar from quadrature, and motivated by the averaging of two end-point values (times the stepsize, i.e. the length of the interval).

The TR method is clearly an implicit method, since it involves $f(t_{i+1}, y_{i+1})$.

Stability and accuracy of the trapezoid method

Assume now that y''' exists and is bounded. Write the two Taylor's expansions and the TR method once again in convenient format:

$$y(t_{i+1}) = y(t_i) + h_i y'(t_i) + \frac{h_i^2}{2!} y''(t_i) + \frac{h_i^3}{3!} y'''(\xi_i), \ \xi_i \in (t_i, t_{i+1}), \tag{33}$$

$$y(t_{i+1}) = y(t_i) + h_i y'(t_{i+1}) - \frac{h_i^2}{2!} y''(t_{i+1}) + \frac{h_i^3}{3!} y'''(\rho_i), \ \rho_i \in (t_i, t_{i+1}), \tag{34}$$

$$y_{i+1} = y_i + \frac{h_i}{2}(f(t_i, y_i) + f(t_{i+1}, y_{i+1})). \tag{TR}$$

For convenience, let $R_F = \frac{h_i^2}{2!} y''(t_i) + \frac{h_i^3}{3!} y'''(\xi_i)$ and $R_B = -\frac{h_i^2}{2!} y''(t_{i+1}) + \frac{h_i^3}{3!} y'''(\rho_i)$. Calculating $\frac{1}{2}[(33) + (34) - 2(\text{TR})]$ gives

The trapezoid method -- stability and accuracy

$$y(t_{i+1}) - y_{i+1} = y(t_i) - y_i + \frac{h_i}{2}(f(t_i, y(t_i)) - f(t_i, y_i)) + \frac{h_i}{2}(f(t_{i+1}, y(t_{i+1})) - f(t_{i+1}, y_{i+1})) + \frac{1}{2}(R_F + R_B).$$

Applying the MVT twice, we get

$$y(t_{i+1}) - y_{i+1} = y(t_i) - y_i + \frac{h_i}{2}(y(t_i) - y_i)\frac{\partial f}{\partial y}(t_i, \tau_i) + \frac{h_i}{2}(y(t_{i+1}) - y_{i+1})\frac{\partial f}{\partial y}(t_{i+1}, \sigma_{i+1}) + \frac{1}{2}(R_F + R_B).$$

Substituting $J_i = \frac{\partial f}{\partial y}(t_i, \tau_i)$ and $J_{i+1} = \frac{\partial f}{\partial y}(t_{i+1}, \sigma_{i+1})$, we get

$$y(t_{i+1}) - y_{i+1} = y(t_i) - y_i + \frac{h_i}{2}(y(t_i) - y_i)J_i + \frac{h_i}{2}(y(t_{i+1}) - y_{i+1})J_{i+1} + \frac{1}{2}(R_F + R_B)$$

from which

$$(y(t_{i+1}) - y_{i+1})(1 - \frac{h_i}{2} J_{i+1}) = (y(t_i) - y_i)(1 + \frac{h_i}{2} J_i) + \frac{1}{2}(R_F + R_B)$$

and

$$(y(t_{i+1}) - y_{i+1}) = (y(t_i) - y_i)(1 + \frac{h_i}{2} J_i)(1 - \frac{h_i}{2} J_{i+1})^{-1} + \frac{1}{2}(R_F + R_B)(1 - \frac{h_i}{2} J_{i+1})^{-1}$$

The trapezoid method -- stability and accuracy

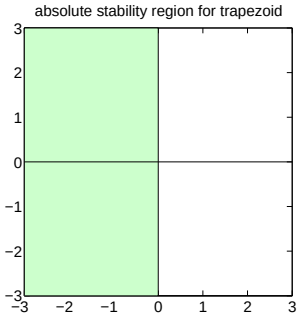
Thus

$$(\text{GTE at } t_{i+1}) = (\text{GTE at } t_i) \times (\text{AF from } t_i \text{ to } t_{i+1}) + \frac{(\text{LTE from } t_i \text{ to } t_{i+1})}{1 - \frac{h_i}{2} J_{i+1}}$$

where, for TR $\text{AF}_i = \dfrac{1 + \frac{h_i}{2} J_i}{1 - \frac{h_i}{2} J_{i+1}}$.

If $|1 + \frac{h_i}{2} J_i| < |1 - \frac{h_i}{2} J_{i+1}|$, past errors are not magnified and the numerical method (TR) is **stable**.

If $|1 + \frac{h_i}{2} J_i| > |1 - \frac{h_i}{2} J_{i+1}|$, past errors are magnified and the numerical method (TR) is **unstable**.



The relation $|1 + \frac{h_i}{2} J_i| < |1 - \frac{h_i}{2} J_{i+1}|$ says that $h_i J_i$ (and $h_i J_{i+1}$) must lie anywhere in the left half plane.

Thus, the region of absolute stability of TR is infinite and includes the entire left half plane. Therefore, the TR method is A-stable. However, note that the absolute stability region of TR is smaller than (more precisely, a proper subset of) the one of BE.

The trapezoid method -- stability and accuracy

Consider the case that J_* is real. The region of absolute stability of TR is $(-\infty, 0)$.

If $J_i < 0$ and $J_{i+1} < 0$, then $|1 + \frac{h_i}{2} J_i| < |1 - \frac{h_i}{2} J_{i+1}|$, for all $h_i > 0$.

Thus, for the TR method to be stable, h_i is not restricted (as long as we proceed forward in time).

However, in order to control the LTE from t_i to t_{i+1}, we still need to restrict the stepsize h. We will see that the LTE formula for TR is $O(h^3)$, thus the GTE is $O(h^2)$, therefore, the TR method is expected to be more accurate than FE (and BE).

For a system of ODEs, the GTE of TR (a vector) is multiplied at each step by the amplification factor (matrix) $(\mathbf{I} + \frac{h_i}{2} J_i)(\mathbf{I} - \frac{h_i}{2} J_{i+1})^{-1}$ where J_i is the Jacobian matrix at (t_i, τ_i), and J_{i+1} is the Jacobian matrix at (t_{i+1}, σ_{i+1}). Past errors are not magnified, if the spectral radius of $(\mathbf{I} + \frac{h_i}{2} J_i)(\mathbf{I} - \frac{h_i}{2} J_{i+1})^{-1}$ is less than 1.

To analyze the order of the LTE, consider that

$$\text{LTE} = \frac{1}{2}(R_F + R_B) = \frac{1}{2}\left(\frac{h_i^2}{2!}(y''(t_i) - y''(t_{i+1})) + \frac{h_i^3}{3!}(y'''(\xi_i) + y'''(\rho_i))\right).$$

The trapezoid method -- stability and accuracy

Applying the MVT to $y''(t_i) - y''(t_{i+1})$, we get

$$\text{LTE} = \frac{h_i^2}{4}(t_i - t_{i+1})y'''(\phi_i) + O(h_i^3) = -\frac{h_i^3}{4} y'''(\phi_i) + O(h_i^3) = O(h_i^3)$$

For the TR method, with $h = \max_i \{h_i\}$, we have LTE $= O(h^3)$, assuming y''' exists and is bounded. We can also show that GTE $= O(h^2)$, thus the TR method is of second order.

Example (linear): Let $y' = -\lambda y$, $y(0) = 2$ (i.e. $f(t, y) = -\lambda y$), and λ is a given scalar.
TR computes

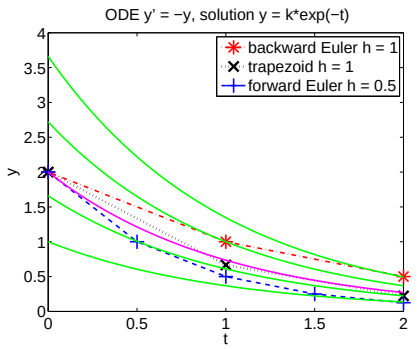
$$y_{i+1} = y_i - \frac{h_i}{2}\lambda(y_i + y_{i+1})$$

which can be rearranged as

$$y_{i+1}(1 + \frac{h_i\lambda}{2}) = y_i(1 - \frac{h_i\lambda}{2})$$

$$\Rightarrow y_{i+1} = y_i(1 - \frac{h_i\lambda}{2})/(1 + \frac{h_i\lambda}{2}).$$

Thus, for example, $y_1 = 2(1 - \frac{h_0\lambda}{2})/(1 + \frac{h_0\lambda}{2})$, which is easy to evaluate, for a chosen h_0 and a given λ.



The trapezoid method -- stability and accuracy

Example (non-linear): Let $y' = e^y(t+1)$, $y(0) = -2$ (i.e. $f(t, y) = e^y(t+1)$).
TR computes

$$y_{i+1} = y_i + \frac{h_i}{2}(e^{y_i}(t_i + 1) + e^{y_{i+1}}(t_{i+1} + 1)).$$

This is a non-linear equation to be solved for y_{i+1}. It can take the form

$$y_{i+1} - \frac{h_i}{2}(t_{i+1} + 1)e^{y_{i+1}} - y_i - \frac{h_i}{2}(t_i + 1)e^{y_i} = 0.$$

For example, in the first timestep, for a chosen h_0, we need to solve $y_1 - e^{y_1}\frac{h_0}{2}(h_0 + 1) - e^{-2}\frac{h_0}{2} + 2 = 0$ for y_1.

Linear multistep methods (LMMs)

We have seen three methods for solving IVPs for ODEs so far: FE, BE and TR. All three are using data within *one* time step, e.g. from t_i to t_{i+1}. For this reason, they are called *one-step* methods. The order of these methods was either one (FE, BE) or two (TR), and the L.T.E. was $O(h^2)$ and $O(h^3)$, respectively. It is worth considering methods of higher order.

The idea behind multistep methods is to use information from more than one step. This explains the term "multistep".

An m-step multistep method uses information from m steps, namely y_i, y_{i-1}, $\cdots$, y_{i-m+1}, as well as f_i, f_{i-1}, $\cdots$, f_{i-m+1}. It may also involve f_{i+1}, if the method is implicit. (*Note*: $f_k \equiv f(t_k, y_k)$.)
The general form of an m-step LMM is

$$y_{i+1} = \sum_{k=1}^{m} a_k y_{i-k+1} + h_i \sum_{k=0}^{m} b_k f_{i-k+1} \qquad (1)$$

where a_k, $k = 1, \cdots, m$, and b_k, $k = 0, \cdots, m$, are coefficients which characterize a particular method. Note that y_{i-k+1} and f_{i-k+1}, $k = 1, \cdots, m$, are known quantities at the step from t_i to t_{i+1}. But $f_{i+1} = f(t_{i+1}, y_{i+1})$ is unknown.
If $b_0 = 0$, then f_{i+1} does not get involved in (1) and the method is explicit. If $b_0 \neq 0$, the method is implicit, and a nonlinear equation needs to be solved for y_{i+1}.

Linear multistep methods (LMMs)

The three methods we have discussed are one-step methods which can be given by (1), with the following choice of coefficients:
FE: $m = 1$, $a_1 = 1$, $b_0 = 0$, $b_1 = 1$;
BE: $m = 1$, $a_1 = 1$, $b_0 = 1$, $b_1 = 0$;
TR: $m = 1$, $a_1 = 1$, $b_0 = \frac{1}{2}$, $b_1 = \frac{1}{2}$.

We are interested in methods with $m \geq 2$. How are the coefficients a_k and b_k, $k = 1, \cdots, m$, chosen for such methods?

A fundamental theorem of calculus states that

$$\int_{t_i}^{t_{i+1}} y'(t)dt = y(t_{i+1}) - y(t_i),$$

which, since $y' = f(t, y)$, can be re-written as

$$y(t_{i+1}) - y(t_i) = \int_{t_i}^{t_{i+1}} f(t, y(t))dt$$

or

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} f(t, y(t))dt. \quad (2)$$

Linear multistep methods (LMMs)

Repeat:

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} f(t, y(t))dt. \quad (2)$$

Consider replacing $y(t_{i+1})$ by y_{i+1}, $y(t_i)$ by y_i and $f(t, y(t))$ by the interpolating polynomial of degree at most $m-1$ that passes through (t_i, f_i), (t_{i-1}, f_{i-1}), $\cdots$, (t_{i-m+1}, f_{i-m+1}). This gives rise to an m-step explicit LMM (referred to as *Adams-Bashforth method*).

If $f(t, y(t))$ is replaced by the interpolating polynomial of degree at most m that passes through (t_{i+1}, f_{i+1}), (t_i, f_i), (t_{i-1}, f_{i-1}), $\cdots$, (t_{i-m+1}, f_{i-m+1}), we get an m-step implicit LMM (referred to as *Adams-Moulton method*).

Similarly, $f(t, y(t))$ may be replaced by the interpolating polynomial of degree at most $m-1$ that passes through (t_{i+1}, f_{i+1}), (t_i, f_i), (t_{i-1}, f_{i-1}), $\cdots$, (t_{i-m+2}, f_{i-m+2}), to give rise to an $(m-1)$-step implicit LMM.

Linear multistep methods (LMMs)

Example: A two-step explicit LMM.
Consider approximating $y(t_{i+1})$ by y_{i+1}, $y(t_i)$ by y_i and $f(t, y(t))$ by the interpolating polynomial $p_1(t)$ of degree at most 1, that passes through (t_i, f_i) and (t_{i-1}, f_{i-1}). Then (2) becomes

$$y_{i+1} = y_i + \int_{t_i}^{t_{i+1}} \left\{ \frac{t - t_i}{t_{i-1} - t_i} f_{i-1} + \frac{t - t_{i-1}}{t_i - t_{i-1}} f_i \right\} dt. \quad (3)$$

The integral in (3) can be calculated analytically, since the integrand is a linear function of t. For simplicity, assume constant stepsize, i.e. $t_i - t_{i-1} = t_{i+1} - t_i = h$. Then (3) gives rise to the two-step explicit method

$$y_{i+1} = y_i + \frac{h}{2}(3 f_i - f_{i-1}). \quad (4)$$

Linear multistep methods (LMMs)

Example: A two-step implicit LMM.
Consider replacing $y(t_{i+1})$ by y_{i+1}, $y(t_i)$ by y_i and $f(t, y(t))$ by the interpolating polynomial $p_2(t)$ of degree at most 2, that passes through (t_{i+1}, f_{i+1}), (t_i, f_i) and (t_{i-1}, f_{i-1}). Then (2) becomes

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} [\frac{(t - t_i)(t - t_{i+1})}{(t_{i-1} - t_i)(t_{i-1} - t_{i+1})} f_{i-1} \quad (5)$$
$$+ \frac{(t - t_{i-1})(t - t_{i+1})}{(t_i - t_{i-1})(t_i - t_{i+1})} f_i + \frac{(t - t_{i-1})(t - t_i)}{(t_{i+1} - t_{i-1})(t_{i+1} - t_i)} f_{i+1}] dt.$$

The integral in (5) can be calculated analytically, since the integrand is a quadratic function of t. For simplicity, assume constant stepsize, i.e. $t_i - t_{i-1} = t_{i+1} - t_i = h$. Then (5) gives rise to the two-step implicit method

$$y_{i+1} = y_i + \frac{h}{12}(-f_{i-1} + 8 f_i + 5 f_{i+1}) \quad (6)$$

Notice that y_{i-1} does not appear explicitly in (4) nor (6). That is, for the above two methods $a_2 = 0$.

Linear multistep methods (LMMs)

To analyze the accuracy of the LMM

$$y_{i+1} = y_i + \frac{h}{2}(3f_i - f_{i-1}), \tag{4}$$

we develop a formula for the LTE (and hence find the order).
Consider again relation (2) in the form

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} y'(t)dt, \tag{2a}$$

and assume we take a step similar to (3) but with exact past information

$$y_{i+1} = y(t_i) + \int_{t_i}^{t_{i+1}} p_1(t)dt \tag{7}$$

where $p_1(t)$ is the polynomial of degree at most 1 passing from $(t_i, y'(t_i))$ and $(t_{i-1}, y'(t_{i-1}))$. Subtracting (7) from (2a), we get

$$y(t_{i+1}) - y_{i+1} = \int_{t_i}^{t_{i+1}} (y'(t) - p_1(t))dt.$$

The polynomial interpolation error theorem gives

$$y'(t) - p_1(t) = \frac{y'^{(2)}(\xi)}{2!}(t - t_i)(t - t_{i-1}),$$

where $\xi \in \text{ospr}\{t, t_{i-1}, t_i\}$, and ξ is unknown and depends on t.

Linear multistep methods (LMMs)

Thus, taking also into account that $y'^{(2)} = y^{(3)}$, we have

$$y(t_{i+1}) - y_{i+1} = \int_{t_i}^{t_{i+1}} \frac{y^{(3)}(\xi)}{2!}(t - t_i)(t - t_{i-1})dt.$$

Notice that the variable t of the integral belongs to $[t_i, t_{i+1}]$ and therefore $(t - t_i)(t - t_{i-1})$ is one-signed in $[t_i, t_{i+1}]$. Thus, the Mean Value Theorem (MVT) for integrals is applicable, and we get

$$y(t_{i+1}) - y_{i+1} = \frac{y^{(3)}(\eta)}{2!} \int_{t_i}^{t_{i+1}} (t - t_i)(t - t_{i-1})dt, \tag{8}$$

for some $\eta \in (t_i, t_{i+1})$. The integral in (8) can be calculated analytically, since the integrand is quadratic. For simplicity, assume constant stepsize, i.e. $t_i - t_{i-1} = t_{i+1} - t_i = h$. Then (8) gives

$$y(t_{i+1}) - y_{i+1} = \frac{y^{(3)}(\eta)}{2!}\frac{5h^3}{6}. \tag{9}$$

Thus the LTE of (4) is $O(h^3)$, and therefore the GTE of (4) is $O(h^2)$, i.e. the method (4) is of second order.
The analysis of

$$y_{i+1} = y_i + \frac{h}{12}(-f_{i-1} + 8f_i + 5f_{i+1}) \tag{6}$$

is more complicated, so we include only the main result: The (Adams-Moulton) method given by (6) is of third order.

Linear multistep methods (LMMs)

A general LMM of the form

$$y_{i+1} = \sum_{k=1}^{m} a_k y_{i-k+1} + h_i \sum_{k=0}^{m} b_k f_{i-k+1} \tag{1}$$

has $2m+1$ free parameters, the coefficients a_k, $k = 1, \cdots, m$, and b_k, $k = 0, \cdots, m$. The coefficients can be chosen so that the LMM is of maximum order. This maximum order can be shown to be $2m$.

Explicit LMMs have stability restrictions for the stepsize, i.e. they require the stepsize be below a certain threshold to control the propagation of errors (even for stable ODEs). Some implicit LMMs also have stability restrictions for the stepsize. In particular, it can be shown that a zero-stable m-step LMM has order $\leq m + 2$.

An m-step LMM requires information from m previous steps. On the first step, we only have $y(t_0)$ and $f(t_0, y(t_0))$, so we cannot use a two-step method, for example. Most implementations of LMMs start with an one-step method, but take small enough initial step so that the order is preserved, then increase the number of steps, as soon as they get the past information required.

When the stepsize is variable, the formulae for the LMMs become quite complicated. (That is, the coefficients are not given by nice numbers.) There is a certain amount of overhead involved in finding the coefficients and various tricks may be used to reduce this overhead.

Linear multistep methods (LMMs)

It is common to use explicit and implicit LMMs in pairs. Such methods are called *predictor-corrector* methods. They consist of applying an explicit formula to predict y_{i+1}, then applying an implicit formula to correct y_{i+1}, using the prediction from the explicit formula within f_{i+1}.
Example:

Predict: $\tilde{y}_{i+1} = y_i + \frac{h}{2}(3f_i - f_{i-1})$,

Correct: $y_{i+1} = y_i + \frac{h}{12}(-f_{i-1} + 8f_i + 5\tilde{f}_{i+1})$

where $\tilde{f}_{i+1} = f(t_{i+1}, \tilde{y}_{i+1})$. Notice that the overall method is explicit, since we do not need to solve any nonlinear equation for y_{i+1}.

Notes on LMMs:

LMMs are efficient, in the sense that they use information (y and f values) already computed in past steps.

Proficient implementations of LMMs use variable stepsize and variable order formulae. That is, they adapt the stepsize so that they take the maximum stepsize such that the LTE is below a certain tolerance. They may also change the order of the LMM (e.g. change m) aiming for efficiency.

Runge-Kutta methods

Consider again the IVP-ODE $y' = f(t, y)$, $y(t_0) = y_0$, and the FE method

$$y_{i+1} = y_i + h_i f(t_i, y_i), \quad i = 0, 1, \cdots \tag{FE}$$

The change of y between t_i and t_{i+1} is determined by y', i.e. by $f(t, y)$, in the interval $[t_i, t_{i+1}]$. The quantity $h_i f(t_i, y_i)$ in (FE) can be viewed as an approximate increment of y between t_i and t_{i+1}. It is therefore reasonable to sample f at more than one point in $[t_i, t_{i+1}]$ to get a more accurate approximation to the increment of y.

Runge-Kutta (RK) methods consider an increment of the form

$$h_i(c_{i1} f(\theta_{i1}, \gamma_{i1}) + c_{i2} f(\theta_{i2}, \gamma_{i2}) + \cdots + c_{is} f(\theta_{is}, \gamma_{is})) = h_i \sum_{j=1}^{s} c_{ij} f(\theta_{ij}, \gamma_{ij})$$

i.e. an increment constructed as a weighted average of the slopes of y at points $(\theta_{ij}, \gamma_{ij}))$, $j = 1, \cdots, s$. The points $(\theta_{ij}, \gamma_{ij}))$ are called *stages*. Usually, θ_{ij} is between t_i and t_{i+1}.

How should the parameters c_{*j}, θ_{*j} and γ_{*j}, $j = 1, \cdots, s$, be chosen? (Another question: how large an s should we take?)

It is quite natural to set $(\theta_{*1}, \gamma_{*1}) = (t_i, y_i)$, and this is often the choice for $(\theta_{*1}, \gamma_{*1})$.

In general, we choose the parameters c_{ij}, θ_{ij} and γ_{ij} so that the LTE is of the highest order possible. That is, in the LTE expression coming from Taylor's expansion, we try to eliminate as many terms as possible.

Runge-Kutta methods

With (5), relation (4) becomes

$$\begin{aligned} y(t_{i+1}) &= y(t_i) + hf(t_i, y(t_i)) + \frac{h^2}{2!}[f_t(t_i, y(t_i)) + f_y(t_i, y(t_i)) f(t_i, y(t_i))] + Kh^3 \\ &= y(t_i) + h\left(f(t_i, y(t_i)) + \frac{h}{2}[f_t(t_i, y(t_i)) + f_y(t_i, y(t_i)) f(t_i, y(t_i))] \right) + Kh^3. \end{aligned} \tag{6}$$

Noting that (6) is the "exact" version of (3), we try to make the terms of (3) and (6) to agree as much as possible, given that we want $y_i = y(t_i)$ and $y_{i+1} = y(t_{i+1})$. This leads to

$$\left\{ \begin{array}{c} c_1 + c_2 = 1 \\ \alpha c_2 = \frac{1}{2} \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} c_1 = 1 - \frac{1}{2\alpha} \\ c_2 = \frac{1}{2\alpha} \end{array} \right\}$$

We still have a free parameter α, for which we must have $\alpha \neq 0$.

A natural choice for α is $\frac{1}{2}$, which results in $c_1 = 0$ and $c_2 = 1$. Then the method in (1) becomes *modified Euler's method*

$$y_{i+1} = y_i + h_i f(t_i + \frac{h_i}{2}, y_i + \frac{h_i}{2} f(t_i, y_i)), \quad i = 0, 1, \cdots \tag{ME}$$

which resembles the midpoint rule. In formula (ME), we have 2 function evaluations (for f), and we use FE with stepsize $h_i/2$ to get the y-point, where f is evaluated the second time.

Runge-Kutta methods

Example: Develop an RK method with two stages, namely (t_i, y_i) and $(t_i + \alpha h_i, y_i + \alpha h_i f(t_i, y_i))$, where α is to be determined. For notational simplicity, let $c_j = c_{ij}$, $h = h_i$, $f_t = \frac{\partial f}{\partial t}$, $f_y = \frac{\partial f}{\partial y}$. The method would be of the form

$$y_{i+1} = y_i + h(c_1 f(t_i, y_i) + c_2 f(t_i + \alpha h, y_i + \alpha h f(t_i, y_i))). \tag{1}$$

Consider the 2D Taylor expansion

$$f(t_i + \alpha h, y_i + \alpha h f(t_i, y_i)) = f(t_i, y_i) + \alpha h f_t(t_i, y_i) + \alpha h f(t_i, y_i) f_y(t_i, y_i) + Ch^2 \tag{2}$$

Substituting (2) into (1) we get

$$\begin{aligned} y_{i+1} &= y_i + h\left(c_1 f(t_i, y_i) + c_2[f(t_i, y_i) + \alpha h f_t(t_i, y_i) + \alpha h f(t_i, y_i) f_y(t_i, y_i) + Ch^2] \right) \\ &= y_i + h\left(c_1 f(t_i, y_i) + c_2[f(t_i, y_i) + \alpha h f_t(t_i, y_i) + \alpha h f(t_i, y_i) f_y(t_i, y_i)] \right) + Ch^3. \end{aligned} \tag{3}$$

Consider also the Taylor expansion

$$y(t_{i+1}) = y(t_i) + hy'(t_i) + \frac{h^2}{2!} y''(t_i) + Kh^3, \tag{4}$$

and note that

$$y'(t) = f(t, y), \quad y''(t) = \frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial y} \frac{dy}{dt} = f_t + f_y f. \tag{5}$$

Runge-Kutta methods

Another choice for α is 1, which results in $c_1 = \frac{1}{2}$ and $c_2 = \frac{1}{2}$. Then the method in (1) becomes *Heun's method* or *improved Euler's method*

$$\begin{aligned} y_{i+1} &= y_i + \frac{h_i}{2}(f(t_i, y_i) + f(t_i + h_i, y_i + h_i f(t_i, y_i))) \\ &= y_i + \frac{h_i}{2}(f(t_i, y_i) + f(t_{i+1}, y_i + h_i f(t_i, y_i))), \quad i = 0, 1, \cdots \end{aligned} \tag{HE}$$

which resembles the trapezoid rule but is explicit. Note that, in formula (HE), we have 2 function evaluations (for f), and we use FE with stepsize h_i to get the second y-point, where f is evaluated.

Yet another choice for α is $\frac{2}{3}$, which results in $c_1 = \frac{1}{4}$ and $c_2 = \frac{3}{4}$. Then the method in (1) becomes another *Heun's method*

$$y_{i+1} = y_i + \frac{h_i}{4}(f(t_i, y_i) + 3f(t_i + \frac{2}{3} h_i, y_i + \frac{2}{3} h_i f(t_i, y_i))), \quad i = 0, 1, \cdots \tag{HE23}$$

All the above methods can be shown to be of second order (i.e. the LTE is of third order).

Runge-Kutta methods -- notes

- By including more terms in (1) and in the Taylor's expansions (2) and (4) and trying to match the respective terms we can get higher order RK methods.
- It can be shown that we need *at least* $s = k$ stages to get a kth order explicit RK method. More specifically, for order $k > 4$, we always need $s > k$.
- Implicit RK methods can also be derived.
- As with FE and BE, implicit RK methods are more appropriate for stiff problems, while explicit ones work well on non-stiff problems.
- Certain implicit RK methods with s stages are of order $2s$.
- Clearly, the larger the s (# of stages) and the number of function evaluations, the higher the cost, but the higher the accuracy (order) as well.
- Implicit methods have better stability properties and (often) better ratio of function evaluations over accuracy (order), but involve the cost of solving non-linear equations (or systems).
- All factors, including number of function evaluations, order, stability, stiffness, stepsize, number of steps, etc, need to be taken into account, when solving differential equations.

Runge-Kutta methods -- Runge-Kutta tableau

Consider again the three RK methods developed so far:
Modified Euler (midpoint)

$$y_{i+1} = y_i + h_i f(t_i + \frac{h_i}{2}, y_i + \frac{h_i}{2} f(t_i, y_i)) \qquad \text{(ME)}$$

Improved Euler -- Heun

$$y_{i+1} = y_i + \frac{h_i}{2}(f(t_i, y_i) + f(t_i + h_i, y_i + h_i f(t_i, y_i))) \qquad \text{(HE)}$$

Other Heun

$$y_{i+1} = y_i + \frac{h_i}{4}(f(t_i, y_i) + 3f(t_i + \frac{2}{3}h_i, y_i + \frac{2}{3}h_i f(t_i, y_i))) \qquad \text{(HE23)}$$

These formulae can also be written in the form
Modified Euler

$$F_1 = f(t_i, y_i)$$
$$F_2 = f(t_i + \frac{h_i}{2}, y_i + \frac{h_i}{2} F_1)$$
$$y_{i+1} = y_i + h_i F_2$$

Runge-Kutta methods -- Runge-Kutta tableau

Improved Euler -- Heun

$$F_1 = f(t_i, y_i)$$
$$F_2 = f(t_i + h_i, y_i + h_i F_1)$$
$$y_{i+1} = y_i + \frac{h_i}{2}(F_1 + F_2)$$

Other Heun

$$F_1 = f(t_i, y_i)$$
$$F_2 = f(t_i + \frac{2}{3}h_i, y_i + \frac{2}{3}h_i F_1)$$
$$y_{i+1} = y_i + \frac{h_i}{4}(F_1 + 3F_2)$$

It turns out that the above forms are very convenient for computation, in the sense that they build the values of f required at each step in an incremental fashion.

All explicit RK formulae can be written in such incremental form. More specifically, an s-stage explicit RK formula (using s function evaluations) takes the form

Runge-Kutta methods -- Runge-Kutta tableau

$$y_{i+1} = y_i + h_i \sum_{j=1}^{s} c_j F_j$$

where

$$F_1 = f(t_i, y_i)$$
$$F_2 = f(t_i + a_2 h_i, y_i + h_i b_{21} F_1)$$
$$F_3 = f(t_i + a_3 h_i, y_i + h_i b_{31} F_1 + h_i b_{32} F_2)$$
$$\cdots$$
$$F_s = f(t_i + a_s h_i, y_i + h_i \sum_{r=1}^{s-1} b_{sr} F_r)$$

for some parameters a_j, $j = 2, \cdots, s$, b_{jr}, $j = 2, \cdots, s$, $r = 1, \cdots, j$, c_j, $j = 1, \cdots, s$.
Instead of writing the above formulae, RK methods are usually found in the literature as represented by a tableau (called **Butcher tableau**) of the form

–	–				
a_2	b_{21}	–			
a_3	b_{31}	b_{32}	–		
.	.	.			
a_s	b_{s1}	b_{s2}	$\cdots$	$b_{s,s-1}$	–
	c_1	c_2	$\cdots$	c_{s-1}	c_s

Runge-Kutta methods -- Runge-Kutta tableau

For example, the methods developed so far are represented by:
Modified Euler

–	–	
1/2	1/2	–
	0	1

Improved Euler -- Heun

–	–	
1	1	–
	1/2	1/2

Other Heun

–	–	
2/3	2/3	–
	1/4	3/4

Here is a well-known fourth order RK formula developed in 1895:

–	–			
1/2	1/2	–		
1/2	0	1/2	–	
1	0	0	1	–
	1/6	1/3	1/3	1/6

Implicit RK formulae can also be represented by tableaux. (Consider filling some of the dashes in the tableau.)

Numerical methods textbooks display the tableaux of a variety of RK formulae.

Runge-Kutta methods -- error and stepsize control

- Software for ODE solvers often includes an error control strategy. That is, the stepsize h_i is adaptively adjusted in order to keep the error below a certain user-chosen tolerance.
- The ultimate target is to control the final GTE below the tolerance. However, the formulae that give bounds for the GTE are usually pessimistic (the bounds are not sharp enough) and inefficient to implement.
- It is preferable to control the LTE at each step. It can be shown that, when the LTE at each step is below hE, for some tolerance E, then
$$|y(t_n) - y_n| \le \frac{e^{(t_n - t_0)L} - 1}{L} \mathrm{E} \approx (t_n - t_0)\mathrm{E}.$$
- The implementation of the error control strategy can be done, as in the case of quadrature rules, by stepsize halving, assuming the order of the LTE of the method is known. If the LTE is $O(h^{\alpha+1})$, and, for a given step, y_{j+1} and $\hat{y}_{j+1}$ are the computed approximations with stepsize h and $\frac{h}{2}$, respectively, then an estimate for the LTE for the half-stepsize approximation is given by $est_j = \frac{1}{2^{\alpha+1} - 1}(\hat{y}_j - y_j)$.

Runge-Kutta methods -- error and stepsize control

- For RK methods, the implementation of the error control strategy is often done by adopting a pair of RK methods, one of order p and another of higher order q, where often $q = p + 1$. For each step i, two approximations y_i and $\hat{y}_i$ are obtained by the pth and qth order methods, respectively. Then an estimate for the LTE of the pth order method is given by $est_j = \hat{y}_j - y_j$.
When two RK methods are paired to form an error estimator, we try to choose the two methods so that they share as many function evaluations as possible. The qth order method is often referred to as *companion* formula for the pth order method.
- Note: If e_j is the local error at t_j, then
$$\frac{e_{j+1}}{e_j} \approx (\frac{h_j}{h_{j-1}})^{p+1}$$
and thus to target tolerance ε in the next timestep, assuming we have an estimate est_j at the current timestep, it makes sense to ask that
$$\frac{\varepsilon}{est_j} \approx (\frac{h_j}{h_{j-1}})^{p+1} \quad \text{or} \quad h_j \approx h_{j-1}(\frac{\varepsilon}{est_j})^{1/(p+1)}.$$
To play it safer, we ask that
$$h_j = h_{j-1}(\frac{\varepsilon}{2est_j})^{1/(p+1)}.$$

Runge-Kutta methods -- error and stepsize control

- One way of choosing the stepsize h_j to step from t_j to t_{j+1} (for $j > 0$) to target the user-chosen tolerance ε is the following:

Let $h_j^{(0)} = h_{j-1}$ and $est_{j+1}^{(0)} = est_j$ $(= \hat{y}_j - y_j)$;
For $i = 1, \cdots$

$$h_j^{(i)} = \min(2h_j^{(i-1)}, \max(\frac{h_j^{(i-1)}}{2}, h_j^{(i-1)}(\frac{\varepsilon}{2est_{j+1}^{(i-1)}})^{1/(p+1)})) \tag{7}$$

Compute y_{j+1} and $\hat{y}_{j+1}$ with this $h_j^{(i)}$. Let $est_{j+1}^{(i)} = \hat{y}_{j+1} - y_{j+1}$.
Check whether the criterion

$$\|\hat{y}_{j+1} - y_{j+1}\| \le \varepsilon \frac{h_j^{(i)}}{t_n - t_0} \quad \text{or} \quad \|\hat{y}_{j+1} - y_{j+1}\| \le \varepsilon \frac{h_j^{(i)}}{t_n - t_0} \|\hat{y}_{j+1}\|, \tag{8}$$

is satisfied.
If the criterion is satisfied, the stepsize $h_j^{(i)}$ is accepted, we let $h_j = h_j^{(i)}$, break the loop, and the method proceeds to the next timestep.
endfor
With this procedure, the stepsize is kept close to as large as the tolerance allows, and $\frac{h_j^{(i-1)}}{2} \le h_j^{(i)} \le 2h_j^{(i-1)}$, that is, we don't have huge changes in the stepsizes.
Usually only a few (often 1 or 2) iterations are needed in the above loop.

Runge-Kutta-Fehlberg method

A fairly popular pair of RK methods referred to as *Runge-Kutta-Fehlberg* method, consists of the following two methods: The fourth order RK formula

–	–				
1/4	1/4	–			
3/8	3/32	9/32	–		
12/13	1932/2197	-7200/2197	7296/2197	–	
1	439/216	-8	3680/513	-845/4104	–
	25/216	0	1408/2565	2197/4104	-1/5

and the fifth order RK formula

–	–					
1/4	1/4	–				
3/8	3/32	9/32	–			
12/13	1932/2197	-7200/2197	7296/2197	–		
1	439/216	-8	3680/513	-845/4104	–	
1/2	-8/27	2	-3544/2565	1859/4104	-11/40	–
	16/135	0	6656/12825	28561/56430	-9/50	2/55

Note that the two methods share the first 5 rows of the tableau.

ODE solvers -- MATLAB

MATLAB's `ode45` uses a pair of RK methods of order 4 and 5 with 6 function evaluations (shared among the two methods). The method is known as the Dormand-Prince pair. This function is commonly used for non-stiff problems, and is the first try for most problems.

Other MATLAB ODE functions:
`ode23`: low order RK (Bogacki and Shampine), non-stiff
`ode23s`: low order RK (modified Rosenbrock), stiff
`ode23t`: low order (trapezoid), moderately stiff
`ode113`: variable order (low to high, Adams-Bashforth-Moulton PC), non-stiff
`ode15s`: variable order (low to medium), stiff

Other ODE methods: Backward differentiation formulae (BDFs)

BDFs are implicit LMMs of the form

$$y_{i+1} = \sum_{k=1}^{m} a_k y_{i-k+1} + h_i b_0 f_{i+1}$$

i.e. LMMs with $b_k = 0,\ k = 1, \cdots, m$. These are very efficient (only one function evaluation) methods, appropriate for stiff problems, and can be developed to exhibit mth order convergence, but there is no BDF (or LMM) method of order greater than 2 that is A-stable.

More on stability of numerical methods for ODEs

We have seen the definition of **A-stable** methods: methods whose region of absolute stability includes the entire left half plane. For those methods the amplification factor is less than 1, for any (positive) step-size h, and the errors tend to zero as $h \to 0$. Equivalently, any perturbations in the initial conditions tend to zero as the number of steps goes to infinity.
A-stability is a strict requirement, satisfied by relatively few methods. However, there are numerical methods for ODEs that are practically useful, but still not A-stable. Those methods usually satisfy some other stability definition, weaker than A-stability. There are many different (and not necessarily equivalent) types of stability, one of which is mentioned here.
A numerical ODE method is called **zero-stable**, if perturbations of size ε in the initial conditions remain bounded by $\kappa\varepsilon$ over a fixed time interval, where κ does not depend on the stepsize h.
Note that zero-stability is weaker than A-stability, but it is enough to result in a convergent numerical method, as long as the local truncation errors tend to zero at a rate faster than the stepsize h.