MA3227 Numerical Analysis II

Lecture 16: Explicit Runge-Kutta Methods

Simon Etter

2019/2020

Explicit Runge-Kutta Methods

Introduction

The last lecture introduced the basic theory for ODEs: if $f(y)$ is Lipschitz-continuous, then $\dot{y} = f(y)$ has a unique solution and the solution is a Lipschitz-continuous function of the initial conditions. Our aim in this lecture is to compute these solutions numerically.

We have already noted previously that

$$y(0) = y_0, \qquad \dot{y}(t) = f(y(t))$$

is equivalent to

$$y(t) = y_0 + \int_0^t f(y(\tau))\, d\tau.$$

All ODE solvers in this module will be derived by applying quadrature to the above integral. This lecture will therefore first review some definitions and results regarding quadrature and then explain how to apply them to ODEs.

Explicit Runge-Kutta Methods

Terminology

Quadrature refers to computing an integral numerically.
Solving an ODE is sometimes referred to as *numerical integration*.

These two terms are easily confused, so try your best to remember this.

Quadrature rule

A quadrature rule is a collection of quadrature points $x_k \in [a, b]$, and quadrature weights $w_k \in \mathbb{R}$ such that

$$\sum_{k=1}^{p} f(x_k)\, w_k \approx \int_a^b f(x)\, dx.$$

Explicit Runge-Kutta Methods

Mapping of quadrature rules

A quadrature rule (x_k, w_k) for an interval $[a, b]$ can be mapped to a quadrature rule $(\hat{x}_k, \hat{w}_k)$ for another interval $[\hat{a}, \hat{b}]$ using the formula

$$\hat{x}_k = \phi(x_k), \qquad \hat{w}_k = \tfrac{\hat{b}-\hat{a}}{b-a}\, w_k,$$

where

$$\phi(x) = \hat{a}\,\tfrac{b-x}{b-a} + \hat{b}\,\tfrac{x-a}{b-a}, \qquad \phi'(x) = \tfrac{\hat{b}-\hat{a}}{b-a}.$$

This follows immediately from the integration by substitution formula

$$\int_{\hat{a}}^{\hat{b}} f(\hat{x})\, d\hat{x} = \int_a^b f(\phi(x))\, \phi'(x)\, dx.$$

Explicit Runge-Kutta Methods

Composite quadrature

Instead of applying a quadrature rule (x_k, w_k) to the whole integral of interest, we can split that integral into smaller parts and apply the quadrature rule mapped according to the result on the previous slide to each partial integral. This is called composite quadrature.

For example, if (x_k, w_k) is a quadrature rule for $[0, 1]$, we can write

$$\int_0^1 f(x)\,dx = \sum_{k=1}^{n} \int_{(k-1)/n}^{k/n} f(x)\,dx = \sum_{k=1}^{n} \int_0^1 f\left(\tfrac{k-1+x}{n}\right) \tfrac{1}{n}\,dx$$
$$\approx \sum_{k=1}^{n} \sum_{\ell=1}^{p} f\left(\tfrac{k-1+x_\ell}{n}\right) \tfrac{w_\ell}{n}.$$

Explicit Runge-Kutta Methods

Error estimates for composite quadrature

Error estimates for composite quadrature rules typically take the form

$$\left| \sum_{k=1}^{n} \sum_{\ell=1}^{p} f\left(\phi_k(x_\ell)\right) \phi'(x_k)\, w_\ell - \int_a^b f(x)\, dx \right| \leq \mathcal{O}\left(n^{-d}\right)$$

for some $d \in [p, 2p]$, assuming $f(x)$ has d derivatives.

d is just a parameter of the quadrature rule which you can look up in textbooks / on Wikipedia along with the quadrature points x_k and weights w_k.

Explicit Runge-Kutta Methods

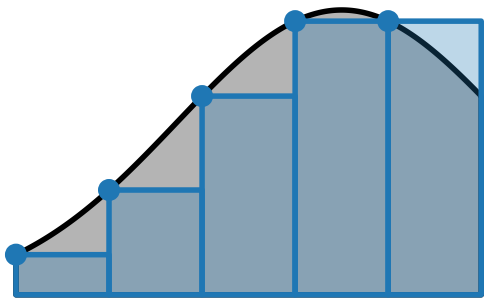
Important quadrature rules

Left-point: $$\int_0^1 f(x)\,dx = \sum_{k=1}^{n} f\left(\tfrac{k-1}{n}\right)\tfrac{1}{n} + \mathcal{O}\left(n^{-1}\right)$$

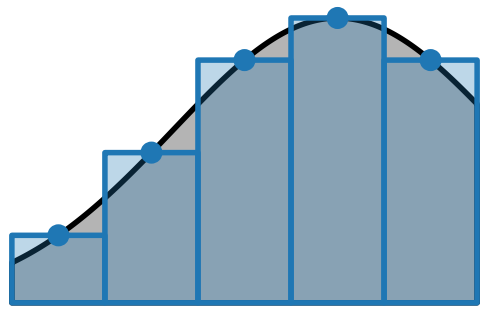
Midpoint: $$\int_0^1 f(x)\,dx = \sum_{k=1}^{n} f\left(\tfrac{k-1/2}{n}\right)\tfrac{1}{n} + \mathcal{O}\left(n^{-2}\right)$$

Trapezoidal: $$\int_0^1 f(x)\,dx = \left(f(0) + 2\sum_{k=1}^{n-1} f(\tfrac{k}{n}) + f(1)\right)\tfrac{1}{n} + \mathcal{O}\left(n^{-2}\right)$$

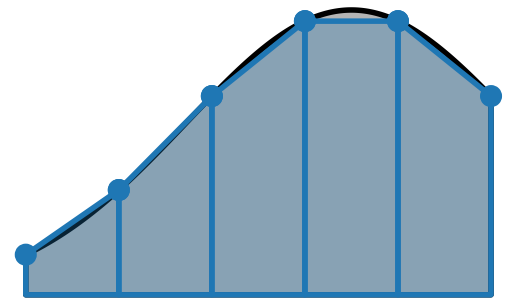
Left-point



Midpoint



Trapezoidal



Explicit Runge-Kutta Methods

Quadrature for ODEs

Our aim is to solve ODEs by applying quadrature to

$$y(t) = y_0 + \int_0^t f(y(\tau))\, d\tau.$$

Given a quadrature formula (τ_k, w_k) for $[0, 1]$, we can easily write down

$$\tilde{y}(t) = y_0 + \sum_{k=1}^{p} f(y(t\,\tau_k))\, t\, w_k,$$

but this formula is not practical because we do not know $y(t)$ for $t > 0$.

An easy solution is to simply avoid evaluating $y(t)$ for $t > 0$ by using the left-point rule:

$$\tilde{y}(t) = y_0 + f\left(y_0\right) t.$$

This formula is known as Euler's method. We will see that it works but does not give good accuracy since it is based on a very low-order quadrature rule.

Explicit Runge-Kutta Methods

Quadrature for ODEs (continued)

Better accuracy can be achieved by using the left-point rule to estimate $y(\frac{t}{2})$ and then using the midpoint rule:

$$\tilde{y}(t) = y_0 + f\left(\tilde{y}(\tfrac{t}{2})\right) t, \qquad \tilde{y}(\tfrac{t}{2}) = y_0 + f\left(y_0\right) \tfrac{t}{2}.$$

The same idea can also be used for the trapezoidal rule and higher-order quadrature rules.

For large t, all of the above methods will deliver poor accuracy since they apply a single quadrature rule to the whole interval. The obvious remedy is to use composite quadrature, which in the context of ODEs works as follows.

- Fix a small interval $[0, T_1]$ and compute $\tilde{y}(T_1)$ using any of the above formulae.
- Fix another small interval $[T_1, T_2]$ and compute $\tilde{y}(T_2)$ using any of the above formulae and initial value $\tilde{y}(T_1)$.
- Repeat until we hit the desired final time T.

ODE solvers of this form are called *single-step* or *Runge-Kutta* methods. See `16_explicit_runge_kutta_methods.jl` for algorithmic details.

Explicit Runge-Kutta Methods

Discussion

As usual, once we have a numerical method which appears to work, we would like to study how the error changes as a function of the parameters of the method.

For Runge-Kutta methods, these parameters are:

- The local stepping function, e.g. Euler vs. midpoint.
- The local step length. In the following, we will assume that we are only interested in $y(T)$ and we compute a numerical approximation $\tilde{y}(T) \approx y(T)$ by using n equispaced steps of length $\frac{T}{n}$.

We are therefore aiming for a result of the form

$$\|\tilde{y}(T) - y(T)\| = \mathcal{O}\big(\varphi(n)\big)$$

where $\varphi(n)$ depends on the stepping function.

The following slides introduce the terminology and notation required to formulate and prove such a result.

Explicit Runge-Kutta Methods

Time propagators

An ODE $\dot{y} = f(y)$ implicitly defines a function $\Phi_T : y_0 \mapsto y(T)$.

Similarly, Runge-Kutta methods define functions $\tilde{\Phi}_T^{(n)} : y_0 \mapsto \tilde{y}(T)$ which satisfy the recurrence formulae

$$\tilde{\Phi}_T^{(n)}(y_0) = \tilde{\Phi}_{T/n}^{(1)}\left(\tilde{\Phi}_{T-1/n}^{(n-1)}(y_0)\right).$$

Note that $\tilde{\Phi}_t^{(1)}(y_0)$ denotes the local stepping function and n indicates the number of steps taken by the Runge-Kutta method.

We will refer to $\Phi_T(y_0)$ and $\tilde{\Phi}_T^{(n)}(y_0)$ as exact and numerical time propagators, respectively.

Def: Consistency of time propagators

We say that $\tilde{\Phi}_t^{(1)}(y_0)$ is $(p+1)$th-order consistent with $\Phi_t(y_0)$ if there exists $C > 0$ such that for all $y_0 \in \mathbb{R}^n$ and small enough $t > 0$ we have

$$\left\|\tilde{\Phi}_t^{(1)}(y_0) - \Phi_t(y_0)\right\| \leq C\, t^{p+1}.$$

Explicit Runge-Kutta Methods

Def: Stability of time propagators

We say that $\tilde{\Phi}_t^{(1)}(y_0)$ has stability constant $\tilde{L} > 0$ if for all small enough t and for all $y_1, y_2 \in \mathbb{R}^n$ we have

$$\left\|\tilde{\Phi}_t^{(1)}(y_2) - \tilde{\Phi}_t^{(1)}(y_1)\right\| \leq (1 + \tilde{L}t) \|y_2 - y_1\|.$$

Remark 1

Stability is in particular guaranteed if we have a bound

$$\left\|\tilde{\Phi}_t^{(1)}(y_2) - \tilde{\Phi}_t^{(1)}(y_1)\right\| \leq \left(1 + \tilde{L}'t + \mathcal{O}(t^2)\right) \|y_2 - y_1\|$$

since we can choose t small enough such that the $\mathcal{O}(t^2)$ term is less than εt and hence we have a stability constant $\tilde{L} = \tilde{L}' + \varepsilon$.

Remark 2

Stability is the same as Lipschitz continuity with Lipschitz constant $1 + \tilde{L}t$. This special form of the Lipschitz constant takes into account that $\tilde{\Phi}_t^{(1)}(y_0) \to y_0$ for $t \to 0$ and hence the Lipschitz constant must approach 1 in this limit.

Explicit Runge-Kutta Methods

Thm: Convergence of Runge-Kutta methods

Assume $\Phi_t^{(1)}(y_0)$ is $(p+1)$th-order consistent with $\Phi_t(y_0)$ and has stability constant $\tilde{L}$. Then,

$$\left\|\Phi_T^{(n)}(y_0) - \Phi_T(y_0)\right\| = \mathcal{O}\left(\exp(\tilde{L}T)\,\frac{T^{p+1}}{n^p}\right).$$

Remark

Note that $(p+1)$th-order consistency is required for pth-order convergence. A simple but wrong explanation for this goes as follows: $(p+1)$th-order consistency means we commit a $\mathcal{O}(n^{-p-1})$ error in each step, and we make n steps so the overall error is $\mathcal{O}(n\,n^{-p-1}) = \mathcal{O}(n^{-p})$.

This argument is wrong because it is not obvious that the total error is simply the sum of the local errors, but it provides an easy way to remember the correct result.

A rigorous argument is presented on the following slides.

Explicit Runge-Kutta Methods

Proof (not examinable). For notational convenience, let us introduce

$$\Phi(y) = \Phi_{T/n}(y), \qquad y_k = \Phi_{Tk/n}(y_0)$$
$$\tilde{\Phi}(y) = \tilde{\Phi}^{(1)}_{T/n}(y), \qquad \tilde{y}_k = \tilde{\Phi}^{(k)}_{Tk/n}(y_0).$$

We then compute

$$\begin{aligned}
\left\|\tilde{\Phi}^{(n)}_T(y_0) - \Phi_T(y_0)\right\| &= \|\tilde{\Phi}(\tilde{y}_{n-1}) - \Phi(y_{n-1})\| \\
&\leq \|\tilde{\Phi}(\tilde{y}_{n-1}) - \tilde{\Phi}(y_{n-1})\| + \|\tilde{\Phi}(y_{n-1}) - \Phi(y_{n-1})\| \\
&\leq (1 + \tfrac{\tilde{L}T}{n}) \|\tilde{y}_{n-1} - y_{n-1}\| + \mathcal{O}\big((\tfrac{T}{n})^{p+1}\big) \\
&\leq (1 + \tfrac{\tilde{L}T}{n})^2 \|\tilde{y}_{n-2} - y_{n-2}\| + \big(1 + (1 + \tfrac{\tilde{L}T}{n})\big) \mathcal{O}\big((\tfrac{T}{n})^{p+1}\big) \\
&\leq \dots \\
&\leq (1 + \tfrac{\tilde{L}T}{n})^n \|\tilde{y}_0 - y_0\| + \left(\sum_{k=0}^{n-1} (1 + \tfrac{\tilde{L}T}{n})^k\right) \mathcal{O}\big(\tfrac{T^{p+1}}{n^p}\big) \\
&\leq 0 + (1 + \tfrac{\tilde{L}T}{n})^{n-1} \mathcal{O}\big(\tfrac{T^{p+1}}{n^p}\big)
\end{aligned}$$

Explicit Runge-Kutta Methods

Proof (not examinable, continued).

Copied from previous slide:

$$\left\|\tilde{\Phi}_T^{(n)}(y_0) - \Phi_T(y_0)\right\| \leq (1 + \tfrac{\tilde{L}T}{n})^{n-1}\, \mathcal{O}\big(\tfrac{T^{p+1}}{n^p}\big)$$

We observe that since $1 + x \leq \exp(x)$, we have

$$(1 + \tfrac{\tilde{L}T}{n})^{n-1} \leq \exp\big(\tilde{L}T\, \tfrac{n-1}{n}\big) \leq \exp\big(\tilde{L}T\big)$$

and thus

$$\left\|\tilde{\Phi}_T^{(n)}(y_0) - \Phi_T(y_0)\right\| = \mathcal{O}\left(\exp(\tilde{L}T)\, \frac{T^{p+1}}{n^p}\right)$$

as claimed.

Explicit Runge-Kutta Methods

Remark

The above result can be abbreviated as

$$\text{stability \& consistency} \implies \text{convergence.}$$

We have already seen a statement of this form when we discussed the convergence of the finite difference discretisation using the bound

$$\|u - u_n\| \leq \underbrace{\|\Delta_n^{-1}\|}_{\text{stability}} \underbrace{\|\Delta_n u + f\|}_{\text{consistency}}.$$

Outlook

In order to apply the above convergence result, we must show that the method is consistent and stable. The following slides illustrate how to do this for the Euler and midpoint methods.

Explicit Runge-Kutta Methods

Consistency of Euler and midpoint methods

Assuming y (or equivalently f) has sufficiently many derivatives, the consistency error $\tilde{y}(t) - y(t)$ can be estimated using Taylor series.

- Euler's method:

$$\tilde{y}(t) = y(0) + f\big(y(0)\big)\, t$$
$$y(t) = y(0) + \dot{y}(0)\, t + \mathcal{O}(t^2)$$

  Since $\dot{y}(0) = f\big(y(0)\big)$, we have $\tilde{y}(t) - y(t) = \mathcal{O}(t^2)$.

- Midpoint method:

$$\begin{aligned}\tilde{y}(t) &= y(0) + f\big(y(0) + f(y(0))\tfrac{t}{2}\big)\, t \\ &= y(0) + f\big(y(0)\big)\, t + f'\big(y(0)\big)\, f\big(y(0)\big)\, \tfrac{t^2}{2} + \mathcal{O}(t^3) \\ y(t) &= y(0) + \dot{y}(0)\, t + \ddot{y}(0)\, \tfrac{t^2}{2} + \mathcal{O}(t^3)\end{aligned}$$

  Since $\dot{y}(0) = f\big(y(0)\big)$ and $\ddot{y}(0) = f'\big(y(0)\big)\, \dot{y}(0) = f'\big(y(0)\big)\, f\big(y(0)\big)$, we have $\tilde{y}(t) - y(t) = \mathcal{O}(t^3)$.

Explicit Runge-Kutta Methods

Stability of Euler and midpoint method

Assuming $f(y)$ is Lipschitz continuous, $\|f(y_2) - f(y_1)\| \leq L \|y_2 - y_1\|$, we can determine the stability constant as follows.

- Euler's method: $\tilde{\Phi}_t(y) = y + f\big(y\big)\, t$.

$$\begin{aligned}\|\tilde{\Phi}_t(y_2) - \tilde{\Phi}_t(y_1)\| &\leq \|y_2 - y_1\| + t\, \|f(y_2) - f(y_1)\| \\ &\leq (1 + tL)\, \|y_2 - y_1\|.\end{aligned}$$

- Midpoint method: $\tilde{\Phi}_t(y) = y + f\big(y + f(y)\, \frac{t}{2}\big)\, t$.

$$\begin{aligned}\|\tilde{\Phi}_t(y_2) - \tilde{\Phi}_t(y_1)\| &\leq \|y_2 - y_1\| + t\, \|f(y_2 + f(y_2)\, \tfrac{t}{2}) - f(y_1 + f(y_1)\, \tfrac{t}{2})\| \\ &\leq (1 + tL)\, \|y_2 - y_1\| + \tfrac{t^2}{2}\, L\, \|f(y_2) - f(y_1)\| \\ &\leq (1 + tL + \tfrac{(tL)^2}{2})\, \|y_2 - y_1\|.\end{aligned}$$

We conclude that both Euler and midpoint methods have a stability constant $\tilde{L}$ which can be chosen equal / arbitrarily close to the Lipschitz constant L of $f(y)$ (cf. Remark 1 on slide 12).

Explicit Runge-Kutta Methods

Convergence of Runge-Kutta methods

Combining the abstract convergence theorem with the above stability and consistency estimates, we conclude that

$$\|\tilde{y}(T) - y(T)\| = \mathcal{O}\left(\exp(LT)\,\frac{T^{p+1}}{n^p}\right)$$

where $p = 1$ for Euler's method and $p = 2$ for the midpoint method.

An ODE solver which achieves the above error is called a pth-order method. Euler is thus a first-order and midpoint a second-order method.

See `convergence()` for a numerical demonstration of these convergence estimates.

Remark

Recall the ODE stability estimate from Lecture 15,

$$\dot{y}_i = f(y_i) \quad \Longrightarrow \quad \|y_1(T) - y_2(T)\| \leq \|y_1(0) - y_2(0)\|\,\exp(LT).$$

The exponential factor in this estimate is the same as in the above convergence result. This is not surprising since already the first Runge-Kutta step introduces an error and this error then appears as an error in the initial value for steps $2, \ldots, n$.

Explicit Runge-Kutta Methods

Remark

The above convergence estimate once again indicates that it may be difficult to solve ODEs on time-scales larger than one over the Lipschitz constant; see `error_bound()`.

However, it is also worth pointing out that the error estimate is a worst-case bound, which is achieved in `error_bound()` because the function itself blows up exponentially. The situation is more benign if the solution converges to a fixed point or is oscillatory. Set λ `= -1.0` or λ `= 1.0im`, respectively, for numerical demonstration.

Explicit Runge-Kutta Methods

Discussion

Recall: the midpoint method achieves a higher order of convergence by using a high-order quadrature rule (the midpoint rule) to compute $\tilde{y}(t)$ and a low-order quadrature rule (the left-point rule) to compute the $y(t\tau_k)$ required by the high-order quadrature rule.

This procedure can be iterated even further to obtain even higher convergence orders; see next slide.

Explicit Runge-Kutta Methods

s-stage Runge-Kutta methods

Assume we have quadrature points θ_i and a sequence of quadrature weights w_{ij} with $i \in \{0, \ldots, s\}$ and $j \in \{0, \ldots, i-1\}$ such that

$$\theta_0 = 0, \qquad \theta_s = 1,$$

and

$$\int_0^{\theta_i} f(\tau)\, d\tau \approx \sum_{j=0}^{i-1} w_{ij}\, f(\theta_j) \qquad \text{for all } i \in \{0, \ldots, s\}.$$

Then, we can compute an approximate solution to $\dot{y} = f(y)$ through

$$\tilde{y}(t) = y(0) + t \sum_{j=0}^{s} w_{sj}\, f_j \approx y(t)$$

where

$$f_i = f\left(y(0) + t \sum_{j=0}^{i-1} w_{ij} f_j\right) \approx f\big(y(\theta_i\, t)\big)$$

Algorithms of this form are called s-stage Runge-Kutta methods.

Explicit Runge-Kutta Methods

Butcher's tableau

Runge-Kutta methods with $s > 2$ involve many parameters. A convenient way to display these parameters is in a table as follows.

$$
\begin{array}{c|cccccc}
0 & & & & & & \\
\theta_1 & w_{10} & & & & & \\
\theta_2 & w_{20} & w_{21} & & & & \\
\vdots & \vdots & \vdots & \ddots & & & \\
\theta_{s-1} & w_{s-1,0} & w_{s-1,1} & \cdots & w_{s-1,s-2} & & \\
\hline
 & w_{s,0} & w_{s,1} & \cdots & w_{s,s-2} & w_{s,s-1}
\end{array}
$$

Such tables are called Butcher tableaus.

Examples

Euler

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

Midpoint

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

Interpretation for midpoint rule:

- $f_0 = f(y(0))$
- $f_1 = f\left(y(0) + \frac{t}{2} f_0\right) \approx f\left(\frac{t}{2}\right)$
- $y(t) \approx y(0) + 0\, t\, f_0 + 1\, t\, f_1$

Explicit Runge-Kutta Methods

Discussion

Given an arbitrary Runge-Kutta method, we can establish its order of convergence by showing that it is consistent and stable and then using the abstract convergence theorem presented above.

Bad news: showing consistency and stability of s-stage Runge-Kutta methods requires long and tedious calculations.

Good news: others have done the work for us, see

`https://en.wikipedia.org/wiki/List_of_Runge-Kutta_methods`

Such lists make it very easy to implement ODE solvers: simply look up a method of the desired order, translate the Butcher tableau into code and make a convergence plot to test that your implementation is correct.

Explicit Runge-Kutta Methods

Example

Runge-Kutta ODE solvers are named after two German mathematicians who among other methods proposed the following Runge-Kutta scheme and showed that it is a fourth-order method.

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

See `rk4_step()` regarding the implementation of this scheme, and `convergence()` for numerical demonstration that it is indeed a fourth-order method.

Explicit Runge-Kutta Methods

Remark

Any reasonable non-composite quadrature rule with s quadrature points leads to a composite quadrature rule with order of convergence in the range $[s, 2s]$. That is, the number of function evaluations per local interval and the order of convergence are proportional.

The same is not true for ODE solvers: it can be shown that the minimal number of stages s to achieve error $\mathcal{O}\left(n^{-p}\right)$ grows faster than p.

p	1	2	3	4	5	6	7	8
$\min s$	1	2	3	4	6	7	9	11

This is one of the reasons why Runge-Kutta methods with more than $s = 4$ stages are rarely used.

Explicit Runge-Kutta Methods

Summary

- Consistency of time propagator: $\left\|\tilde{\Phi}_t^{(1)}(y_0) - \Phi_t(y_0)\right\| = C\, t^{p+1}$.
- Stability of time propagator:

$$\left\|\tilde{\Phi}_t^{(1)}(y_2) - \tilde{\Phi}_t^{(1)}(y_1)\right\| \leq \left(1 + \tilde{L}t + \mathcal{O}(t^2)\right) \|y_2 - y_1\|.$$

- Stability & consistency implies convergence theorem:
  If $\tilde{\Phi}_t^{(1)}(y_0)$ is stable and $(p+1)$th-order consistent, then

$$\left\|\Phi_T^{(n)}(y_0) - \Phi_T(y_0)\right\| = \mathcal{O}\left(\exp(\tilde{L}\,T)\,\frac{T^{p+1}}{n^p}\right).$$

- Butcher-tableau representation of Runge-Kutta schemes:

$$\begin{array}{c|c} \theta_i & w_{ij} \\ \hline & w_{sj} \end{array} \quad \longleftrightarrow \quad \begin{aligned} f_i &= f\left(y_0 + \sum\nolimits_j w_{ij}\, t\, f_j\right) \approx f(\theta_j t) \\ \tilde{y}(t) &= y_0 + \sum\nolimits_j w_{sj}\, t\, f_j \end{aligned}$$