Time splitting methods

Martin Schreiber <schreiberx@gmail.com>

October 21, 2017

Abstract

This document discusses how time methods can be applied to ordinary and partial differential equations which are split into two different parts. One part is assumed to be linear and the other one non-linear.

The first goal is to gain understanding in the the formulation of higher-order time integrations and how they can be realized in software. Based on this step, the second goal is to investigate the accuracy of the time stepping methods.

Changelog

2018-12-28: (Martin) Added series section to avoid singularities

1 Introduction

Let a physical system be described *explicitly* by its time derivative

$$\frac{\partial U}{\partial t} = LU + N(U)$$

with U denoting the state variables and L and N a stiff linear and non-stiff non-linear operator, respectively. Here, stiff part L is assumed to require by far smaller time step sizes than for the non-stiff part

2 Related work

- Dale Duran's book "Numerical Methods for Fluid Dynamics" for basic time stepping methods (explicit, implicit, Strang splitting, ...)
- Hilary Weller's paper "Runge–Kutta IMEX schemes for the Horizontally Explicit/Vertically Implicit (HEVI) solution of wave equations"

3 Time splitting methods

3.1 Explicit time stepping

Explicit time stepping methods treat each tendency term with an explicit time stepping method.

3.2 Implicit Explicit time stepping (IMEX)

3.2.1 1st order

The most trivial 1st order accurate implicit time stepping method is given by

$$\frac{U^{n+1} - U^n}{\Delta t} = LU^{n+1} + N(U^n)$$

hence requiring to solve for

$$\begin{aligned} U^{n+1} - U^n &= \Delta t L U^{n+1} + \Delta t N(U^n) \\ (I - \Delta t L)\, U^{n+1} &= U^n + \Delta t N(U^n) \end{aligned}$$

3.2.2 2nd order

We can get a 2nd order accurate time stepping by combining 2nd order accurate time steppers in a particular way. Let

$$L^{imp}(\Delta t, U) = (I - \Delta t L)^{-1} U$$

compute one implicit time step for the linear parts and let L^{imp2} be the 2nd order pedant , then

$$U^{n+1} = L_2^{imp2}\left(\frac{1}{2}\Delta t,\, N\left(\Delta t,\, L_2^{imp2}\left(\frac{1}{2}\Delta t,\, U^n\right)\right)\right).$$

Hence, we first run a half time step for the linear part, then a full one for the nonlinear part and again a half time step for the linear one. An alternative interpretation would be a full time step with the linear and non-linear operator, but shifted by 1/2.

For sake of applying some analysis we assume that the non-linear operator is linear as use denote N^2_Δ a second order accurate time stepping method (e.g. using Runge-Kutta 2)

$$L^2_{\Delta t} \circ U^n = L^{2,imp}(\Delta t, U^n)$$

We can then write

$$U^{n+1} = L^2_{\frac{1}{2}\Delta t} N^2_{\Delta t} L^2_{\frac{1}{2}\Delta t} U^n$$

where L^2 and N^2 represent a 2nd order accurate time stepping methods.

3.3 Explicit and implicit time stepping, 2nd order with multistepping

A common leap-frog-like choice is a 2nd order multistep method

$$\frac{U^{t+1} - U^{t-1}}{2\Delta t} = L\left(\frac{U^{t-1} + U^{t+1}}{2\Delta t}\right) + N(U^t)$$

Here the implicitness comes into play with a centered averaging. The centered difference and averaging is known to be 2nd order accurate. Solving this for U^{t+1} yields

$$\begin{aligned} U^{t+1} &= U^{t-1} + LU^{t-1} + LU^{t+1} + 2\Delta t N(U^t) \\ U^{t+1} - LU^{t+1} &= U^{t-1} + LU^{t-1} + 2\Delta t N(U^t) \\ (I - W_f)\, U^{t+1} &= U^{t-1} + LU^{t-1} + 2\Delta t N(U^t) \end{aligned}$$

3.4 Fractional-Step methods

Using the formulation

$$U(t) = exp\left((L_1 + L_2)\,\Delta t\right) U(0)$$

with $L_{1/2}$ two linear operators and assuming a linearity of both terms, we can treat both terms independently

$$U(t) = exp\left((L_1 + L_1)\,\Delta t\right) U(0) = exp\left(L_1 \Delta t\right) exp\left(L_2 \Delta t\right) U(0)$$

We can in particular evaluate both terms in arbitrary order:

$$U(t) = exp\left(L_1 \Delta t\right) exp\left(L_2 \Delta t\right) U(0) = exp\left(L_1 \Delta t\right) exp\left(L_2 \Delta t\right) U(0)$$

Note, that each $exp()$ term can be replaced with an arbitrary time stepping method, however each time stepping method executable with different time step sizes.

In case of non-linear terms, Strang splitting can be used in non-linear cases where a commutative property of L_1 and L_2 is not given:

$$U(t) = exp\left(L_1 \frac{\Delta t}{2}\right) exp\left(L_2 \Delta t\right) exp\left(L_1 \frac{\Delta t}{2}\right) U(0)$$

with L_2 typically the non-linear term N, yielding for a 1st order accurate scheme including non-linearities N

$$U(t) = exp\left(L_1 \frac{\Delta t}{2}\right) \Delta t N\left(exp\left(L_1 \frac{\Delta t}{2}\right) U(0)\right)$$

or

$$U(t) = \frac{\Delta t}{2} N \left(exp\left(\Delta t L_1\right) \frac{\Delta t}{2} N\left(U(0)\right)\right)$$

We can also in general swap the order of operators if

$$\begin{aligned} L_1 L_2 &= \Sigma_1 \Lambda_1 \Sigma_1^{-1} \Sigma_2 \Lambda_2 \Sigma_2^{-1} \\ &= \Sigma_1 \Lambda_1 \Lambda_2 \Sigma_2^{-1} \\ &= \Sigma_1 \Lambda_2 \Lambda_1 \Sigma_2^{-1} \\ &= \Sigma_1 \Lambda_2 \Sigma_2^{-1} \Sigma_1 \Lambda_1 \Sigma_2^{-1} \\ &= L_2 L_1 \end{aligned}$$

Hence, this requires that the eigenvectors are identical: $\Sigma_1 = \Sigma_2$.

3.5 Split-explicit methods

(See e.g. Knoth and Wensch, Generalized Split-Explicit ...)

These methods are similar to the previous ones, assume the non-linear part to be constant over one time step size and to subcycling for the linear stiff part.

3.6 Exponential integrator methods

See also Will Wright, History of Exponential Integrators

3.6.1 Integrating factor method (IF)

The IF method was introduced in Lawson (1967). First, we solve the linear part exactly:

$$V = e^{-tL} U$$

where we emphasize a time step backward in time which we can also write with

$$U = e^{tL} V$$

This is followed by a change of variables with $U = e^{tL}V$ leading to

$$\begin{aligned} \frac{\partial U}{\partial t} - LU &= N(U) \\ \frac{\partial e^{tL} V}{\partial t} - Le^{tL}V &= N(e^{tL}V) \\ Le^{tL}V + e^{tL}\frac{\partial V}{\partial t} - Le^{tL}V &= N(e^{tL}V) \\ \frac{\partial V}{\partial t} &= e^{-tL} N(e^{tL}V) \end{aligned}$$

Here, we first factor out the linear parts and can then apply whatever higher order TS method for the non-linear parts. It it important to note that the backward transformation to U only has to be done if the prognostic fields are required (e.g. for Semi-Lagrangian methods).

IFAB2 Second order Integrating Factor method mixed with Adams-Bashforth (see [Cox & Matthews]):

$$u_{n+1} = u_n e^{ch} + \frac{3h}{2} F_n e^{ch} - \frac{h}{2} F_{n-1} e^{2ch}$$

For linear operators, we get

$$\begin{aligned} U_{n+1} &= e^{\Delta t L} U_n + \frac{3\Delta t}{2} F(U_n) e^{\Delta t L} - \frac{h}{2} F(U_{n-1}) e^{2\Delta t L} \\ &= \varphi_0(\Delta t L) U_n + \frac{3\Delta t}{2} \varphi_0(\Delta t L) F(U_n) - \frac{h}{2} \varphi_0(2\Delta t L) F(U_{n-1}) \end{aligned}$$

IFRK2 Written with RK2, we get (see [Cox & Matthews])

$$u_{n+1} = u_n e^{ch} + \frac{h}{2}(F(u_n)e^{ch} + F((u_n + hF(u_n)e^{ch}, t_n + h))$$

which is in terms of linear operators:

$$\begin{aligned}
u_{n+1} &= e^{\Delta t L} U_n + \frac{\Delta t}{2}(e^{\Delta t L} F(U_n) + F((U_n + \Delta t e^{\Delta t L} F(U_n), t_n + \Delta t)) \\
&= \varphi_0(\Delta t L) U_n \\
&\quad + \frac{\Delta t}{2}(\\
&\qquad \varphi_0(\Delta t L) F(U_n) \\
&\qquad + F((U_n + \Delta t \varphi_0(\Delta t L) F(U_n), t_n + \Delta t) \\
&\quad)
\end{aligned}$$

3.6.2 Exponential time differencing (ETD)

See [Cox and Matthews]. We first multiply both sides with e^{-tL}, yielding

$$e^{-tL}\frac{\partial U}{\partial t} = e^{-tL} L U + e^{-tL} N(U)$$

and computes the integral over the time $[0;t]$

$$\begin{aligned}
\int_0^t e^{-\tau L}\frac{\partial U}{\partial \tau} d\tau &= \int_0^t e^{-\tau L} L U d\tau + \int_0^t e^{-\tau L} N(U) d\tau \\
\int_0^t e^{-\tau L}\frac{\partial U}{\partial \tau} d\tau - \int_0^t e^{-\tau L} L U d\tau &= \int_0^t e^{-\tau L} N(U) d\tau
\end{aligned}$$

We can write

$$\begin{aligned}
\int_0^t e^{-\tau L}\frac{\partial U}{\partial \tau} d\tau - \int e^{-\tau L} L U d\tau &= \int_0^t \left(e^{-\tau L}\frac{\partial U}{\partial \tau} - e^{-\tau L} L U \right) d\tau \\
&= \int_0^t \frac{\partial}{\partial \tau}\left(e^{-\tau L} U \right) d\tau \\
&= \left[e^{-tL} U \right]_0^t
\end{aligned}$$

leading to

$$\begin{aligned}
\left[e^{-tL} U \right]_0^t &= \int_0^t e^{-\tau L} N(U) d\tau \\
e^{-tL} U(t) - U(0) &= \int_0^t e^{-\tau L} N(U) d\tau \\
U(t) &= e^{tL} U(0) + e^{tL} \int_0^t e^{-\tau L} N(U) d\tau
\end{aligned}$$

So far, there's no time integration error introduced and the integral on the RHS then requires to be approximated. The following formulations can be efficiently written with the φ_i function (see Hockbruck and Ostermann):

$$\begin{aligned}
\varphi_0(K) &= e^K \\
\varphi_1(K) &= K^{-1}\left(e^K - I\right) \\
\varphi_2(K) &= K^{-2}\left(e^K - I - K\right) \\
\varphi_3(K) &= K^{-3}\left(2e^K - 2 - 2K - K^2\right)/2 \\
\varphi_4(K) &= K^{-4}\left(6e^K - 6 - 6K - 3K^2 - K^3\right)/6
\end{aligned}$$

(using equation below) or in general given by[1]

[1] [numerically validated in REXI test script test_function_identities.py]

$$\varphi_N(K) = K^{-N}\left((N-1)!e^K - \sum_{i=0}^{N-1}\frac{(N-1)!}{i!}K^i\right)/(N-1)!.$$

A generic analytical formulation is given by given by

$$\varphi_i(K) = \int_0^1 e^{(1-\tau)K}\frac{\tau^{i-1}}{(i-1)!}d\tau \,\text{for}\, i \geq 1$$

or

$$\varphi_{i+1}(K) = \frac{\varphi_i(K) - \varphi_i(0)}{K}\,\text{for}\, i \geq 0$$

ETD1: A first order approximation is given by the assumption of $N(U)$ not significantly changing over the time integration interval. Using $U(0) = U_0$ and $U(\tau) = U_1$, this leads to the approximation

$$\begin{aligned}\int_0^t e^{-\tau L}N(U(\tau))d\tau &\approx \int_0^t e^{-\tau L}N(U_0)d\tau \\ &= \int_0^t e^{-\tau L}d\tau N(U_0) \\ &= \left[-L^{-1}e^{-tL}\right]_0^t N(U_0) \\ &= \left(L^{-1}e^{-tL} - L^{-1}\right)N(U_0) \\ &= \left(L^{-1}\left(e^{-tL} - I\right)\right)N(U_0)\end{aligned}$$

Finally, we get the ETD1 time stepping method

$$U_1 = e^{\Delta tL}U_0 + \left(L^{-1}\left(e^{-\Delta tL} - I\right)\right)N(U_0).$$

For ETD1, we can then write

$$U_1 = \varphi_0(\Delta tL)U_0 + \Delta t\varphi_1(\Delta tL)N(U_0).$$

This is also known as Rosenbrock method which has order 2 (TODO: Check this order 2, should be only 1st order in the nonlinearities!)

ETD2 (multi-step): We can approximate the quadrature of the nonlinearities with a 2nd order accurate method and using a multi-step method for the non-linearities over time interval

$$N(\tau) \approx N_0 + \frac{\tau}{\Delta t}(N_{-1} - N_0)$$

With the exponential integrator formulation above this leads to

$$\int_0^t e^{-\tau L}N(U)d\tau \approx \int_0^t e^{-\tau L}\left(N_0 + \frac{\tau}{\Delta t}(N_{-1} - N_0)\right)d\tau.$$

For an ODE, this leads to (see paper)

$$u_{n+1} = u_n e^{ch} + F(u_n)\left((1+hc)e^{ch} - 1 - 2hc\right)/hc^2 + F(u_{n-1})\left(-e^{ch} + 1 + hc\right)/hc^2$$

which we can rewrite with matrix notation to [TODO: This is not checked!]

$$u_{n+1} = u_n e^{\Delta tL} + \left(\Delta tL^2\right)^{-1}\left((1+\Delta tL)e^{\Delta tL} - I - 2\Delta tL\right)F(u_n) + \left(\Delta tL^2\right)^{-1}\left(-e^{\Delta tL} + I + \Delta tL\right)F(u_{n-1})$$

Using the φ notations, this can be compactly written as

$$\begin{aligned}u_{n+1} &= e^{\Delta tL}u_n + \Delta t\left(\Delta tL\right)^{-2}\left((1+\Delta tL)e^{\Delta tL} - I - 2\Delta tL\right)F(u_n) - \Delta t\left(\Delta tL\right)^{-2}\left(e^{\Delta tL} - I - \Delta tL\right)F(u_{n-1}) \\ &= e^{\Delta tL}u_n + \Delta t\left(\Delta tL\right)^{-2}\left(\Delta tLe^{\Delta tL} - \Delta tL + e^{\Delta tL} - I - \Delta tL\right)F(u_n) - \Delta t\varphi_2\left(\Delta tL\right)F(u_{n-1}) \\ &= e^{\Delta tL}u_n + \Delta t\left(\Delta tL\right)^{-2}\left(\Delta tLe^{\Delta tL} - \Delta tL\right)F(u_n) + \Delta t\left(\Delta tL\right)^{-2}\left(e^{\Delta tL} - I - \Delta tL\right)F(u_n) - \Delta t\varphi_2\left(\Delta tL\right)F(u_{n-1}) \\ &= e^{\Delta tL}u_n + \Delta t\left(\Delta tL\right)^{-2}\left(\Delta tLe^{\Delta tL} - \Delta tL\right)F(u_n) + \Delta t\varphi_2\left(\Delta tL\right)F(u_n) - \Delta t\varphi_2\left(\Delta tL\right)F(u_{n-1}) \\ &= \varphi_0\left(\Delta tL\right)u_n + \Delta t\varphi_1\left(\Delta tL\right)F(u_n) + \Delta t\varphi_2\left(\Delta tL\right)\left(F(u_n) - F(u_{n-1})\right)\end{aligned}$$

That's fracking pretty! We can even parallelize over the different sum terms in this equation!

$$\boldsymbol{TODO: IMPLEMENT}\, u_{n+1} = \varphi_0\left(\Delta tL\right)u_n + \Delta t\varphi_1\left(\Delta tL\right)F(u_n) + \Delta t\varphi_2\left(\Delta tL\right)\left(F(u_n, t_n) - F(u_{n-1})\right)$$

Exponential time differencing Runge-Kutta (ETDRK): The main issue with ETD is that it requires storing the previous stages. E.g. This gets a problem for the first few time steps. In this section we summarize (see [Cox and Matthews, Exp. time diff.]). We use a “change” of variables:

$$\begin{aligned} h &= \Delta t \\ c &= L \end{aligned}$$

ETD1RK Same as ETD1

ETD2RK The ODE version (See [C&M]) is given by

$$a_n = u_n e^{ch} + F(u_n, t_n) \left(e^{ch} - 1\right) / c,$$

$$F(t) = F(u_n, t_n) + \Delta t \left(F(a_n, t_n + \Delta t) - F(u_n, t_n)\right) / h + O(h^2)$$

and finally

$$u_{n+1} = a_n + \left(F(a_n, t_n + \Delta t) - F(u_n, t_n)\right) \left(e^{ch} - 1 - hc\right) / hc^2$$

ETD2RK for linear operators can then be formulated by

$$A_n = \varphi_0(\Delta t L) U_n + \Delta t \varphi_1(\Delta t L) F(U_n, t_n)$$

$$U_{n+1} = A_n + \Delta t \varphi_2(\Delta t L) \left(F(A_n, t_n + \Delta t) - F(U_n, t_n)\right)$$

ETD4RK This version for ODE (See [C&M])is given by

$$\begin{aligned} a_n &= u_n e^{ch/2} + \left(e^{ch/2} - 1\right) F(u_n, t_n) / c \\ b_n &= u_n e^{ch/2} + \left(e^{ch/2} - 1\right) F(a_n, t_n + h/2) / c \\ c_n &= a_n e^{ch/2} + \left(e^{ch/2} - 1\right) \left(2F(b_n, t_n + h/2) - F(u_n, t_n)\right) / c \end{aligned}$$

Then the solution for an ODE is given by

$$\begin{aligned} u_{n+1} = \; & u_n e^{ch} + \\ & \{ \\ & \; F(u_n, t_n) \left[-4 - hc + e^{ch} \left(4 - 3hc + h^2 c^2\right)\right] \\ & +2 \left(F(a_n, t_n + h/2) + F(b_n, t_n + h/2)\right) \left[2 + hc + e^{ch} \left(-2 + hc\right)\right] \\ & +F(c_n, t_n + h) \left[-4 - 3hc - h^2 c^2 + e^{ch} (4 - hc)\right] \\ & \}/h^2 c^3 \end{aligned}$$

For a linear operator matrix L, this yields

$$\begin{aligned} A_n &= e^{\frac{1}{2}\Delta t L} U_n + L^{-1} \left(e^{\frac{1}{2}\Delta t L} - I\right) F(U_n, t_n) \\ B_n &= e^{\frac{1}{2}\Delta t L} U_n + L^{-1} \left(e^{\frac{1}{2}\Delta t L} - I\right) F(A_n, t_n + \frac{1}{2}\Delta t) \\ C_n &= e^{\frac{1}{2}\Delta t L} A_n + L^{-1} \left(e^{\frac{1}{2}\Delta t L} - I\right) \left(2F(B_n, t_n + \frac{1}{2}\Delta t) - F(U_n, t_n)\right) \end{aligned}$$

and with φ functions

$$\begin{aligned} A_n &= \varphi_0 \left(\frac{1}{2}\Delta t L\right) U_n + \frac{1}{2}\Delta t \varphi_1 \left(\frac{1}{2}\Delta t L\right) F(U_n, t_n) \\ B_n &= \varphi_0 \left(\frac{1}{2}\Delta t L\right) U_n + \frac{1}{2}\Delta t \varphi_1 \left(\frac{1}{2}\Delta t L\right) F(A_n, t_n + \frac{1}{2}\Delta t) \\ C_n &= \varphi_0 \left(\frac{1}{2}\Delta t L\right) A_n + \frac{1}{2}\Delta t \varphi_1 \left(\frac{1}{2}\Delta t L\right) \left(2F(B_n, t_n + \frac{1}{2}\Delta t) - F(U_n, t_n)\right) \end{aligned}$$

Note, that the φ_0 function is applied to A_n and not U_n!

Furthermore, we set

$$\begin{aligned}
R_0 &= U_n \\
R_1 &= F(U_n, t_n) \\
R_2 &= F(A_n, t_n + \frac{1}{2}\Delta t) + F(B_n, t_n + \frac{1}{2}\Delta t) \\
R_3 &= F(C_n, t_n + \Delta t)
\end{aligned}$$

and further

$$\begin{aligned}
U_{n+1} = \; & \varphi_0\left(\Delta t L\right) R_0 \\
& + \Delta t \left(\Delta t L\right)^{-3} \left[-4I - \Delta t L + e^{\Delta t L}\left(4I - 3\Delta t L + \left(\Delta t L\right)^2\right)\right] R_1 \\
& + 2\Delta t \left(\Delta t L\right)^{-3} \left[2I + \Delta t L + e^{\Delta t L}\left(-2I + \Delta t L\right)\right] R_2 \\
& + \Delta t \left(\Delta t L\right)^{-3} \left[-4I - 3\Delta t L - \left(\Delta t L\right)^2 + e^{\Delta t L}(4 - \Delta t L)\right] R_3.
\end{aligned}$$

We introduce the υ_n functions which are given by

$$\begin{aligned}
\upsilon_1(K) &= \frac{-4 - K + e^K \cdot \left(4 - 3K + K^2\right)}{K^3} \\
\upsilon_2(K) &= \frac{2 + K + e^K \cdot (-2 + K)}{K^3} \\
\upsilon_3(K) &= \frac{-4 - 3K - K^2 + e^K \cdot (4 - K)}{K^3}
\end{aligned}$$

and we can write

$$U_{n+1} = \varphi_0\left(\Delta t L\right) R_0 + \Delta t \left(\upsilon_1\left(\Delta t L\right) R_1 + 2\upsilon_2\left(\Delta t L\right) R_2 + \upsilon_3\left(\Delta t L\right) R_3\right)$$

3.7 Coping with φ and υ functions and their singularity

Next we discuss how to cope with the singularity for $K \to 0$ for e.g. φ_n functions with $n > 0$.

3.7.1 Limit

We can write the recurrence equation as

$$\lim_{K\to 0} \varphi_{i+1}(K) = \lim_{K\to 0} \frac{\varphi_k(K) - \varphi_k(0)}{K} \approx \frac{\partial}{\partial K}\varphi_k(K).$$

Hence, we get

$$\begin{aligned}
\lim_{K\to 0} \varphi_1(K) &= e^K = 1 \\
\lim_{K\to 0} \varphi_2(K) &= \frac{\partial}{\partial K}\left(K^{-1}\left(e^K - I\right)\right) \\
&= K^{-2}\left(e^K K^{-1} + \left(e^K - I\right) K^{-2}\right) \\
&= K^{-1}\left(e^K + \left(e^K - I\right) K^{-1}\right)
\end{aligned}$$

Similarly, we can handle the υ_n functions for RK4 which all have 1.0/6.0 as their limit for $K \to 0$. For the K^{-3}, K^{-2} and K^{-1} terms this leads to

$$\begin{aligned}
\lim_{K\to 0} \upsilon_1(K) &= \lim_{K\to 0} \frac{-4 - K + e^K\left(4 - 3K + K^2\right)}{K^3} \\
&= \lim_{K\to 0} \frac{e^K(K^2 - K + 1) - 1}{3K^2} \\
&= \lim_{K\to 0} \frac{e^K K(K+1)}{6K} \\
&= \lim_{K\to 0} \frac{e^K(K^2 + 3K + 1)}{6} \\
&= \frac{1}{6}
\end{aligned}$$

$$\begin{aligned}\lim_{K\to 0} \upsilon_2(K) &= \lim_{K\to 0} \frac{2+K+e^K(-2+K)}{K^3}\\ &= \lim_{K\to 0} \frac{e^K(K-1)+1}{3K^2}\\ &= \lim_{K\to 0} \frac{e^K K}{6K}\\ &= \lim_{K\to 0} \frac{e^K(K+1)}{6}\\ &= \frac{1}{6}\end{aligned}$$

$$\begin{aligned}\lim_{K\to 0} \upsilon_3(K) &= \lim_{K\to 0} \frac{-4-3K-K^2+e^K(4-K)}{K^3}\\ &= \lim_{K\to 0} \frac{-e^K(K-3)-2K-3}{3K^2}\\ &= \lim_{K\to 0} \frac{-e^K(K-2)-2}{6K}\\ &= \lim_{K\to 0} \frac{-e^K(K-1)}{6}\\ &= \frac{1}{6}\end{aligned}$$

3.7.2 Series

An alternative way is given by a series form of the exponential

$$\exp(K) = \sum_{l=0}^{\infty} \frac{K^l}{l!},$$

helping to canceling out the singularity.

φ_i **functions** We can find a generalization by

$$\begin{aligned}\varphi_N(K) &= K^{-N}\left((N-1)!e^K - \sum_{i=0}^{N-1}\frac{(N-1)!}{i!}K^i\right)/(N-1)!\\ &= K^{-N}\left((N-1)!\sum_{i=0}^{\infty}\frac{K^i}{i!} - \sum_{i=0}^{N-1}\frac{(N-1)!}{i!}K^i\right)/(N-1)!\\ &= K^{-N}\left(\sum_{i=0}^{\infty}\frac{(N-1)!}{i!}K^i - \sum_{i=0}^{N-1}\frac{(N-1)!}{i!}K^i\right)/(N-1)!\\ &= K^{-N}\left(\sum_{i=N}^{\infty}\frac{(N-1)!}{i!}K^i\right)/(N-1)!\\ &= \sum_{i=0}^{\infty}\frac{K^i}{(i+N)!}\end{aligned}$$

υ_i **functions:** Given the υ_i equations, we can try to find a similar formulation

$$
\begin{aligned}
\upsilon_1(K) &= \frac{-4-K+e^K\cdot\left(4-3K+K^2\right)}{K^3} \\
\upsilon_1(K)K^3 &= -4-K+\sum_{l=0}^{\infty}\frac{K^l}{l!}\left(4-3K+K^2\right) \\
&= -4-K+\sum_{l=0}^{\infty}\frac{K^l}{l!}4-\sum_{l=0}^{\infty}\frac{K^l}{l!}3K+\sum_{l=0}^{\infty}\frac{K^l}{l!}K^2 \\
&= -4-K+4\sum_{l=0}^{\infty}\frac{K^l}{l!}-3\sum_{l=0}^{\infty}\frac{K^{l+1}}{l!}+\sum_{l=0}^{\infty}\frac{K^{l+2}}{l!} \\
&= -K+4\sum_{l=1}^{\infty}\frac{K^l}{l!}-3\sum_{l=0}^{\infty}\frac{K^{l+1}}{l!}+\sum_{l=0}^{\infty}\frac{K^{l+2}}{l!} \\
&= 4\sum_{l=2}^{\infty}\frac{K^l}{l!}-3\sum_{l=1}^{\infty}\frac{K^{l+1}}{l!}+\sum_{l=0}^{\infty}\frac{K^{l+2}}{l!} \\
&= 2K^2+4\sum_{l=3}^{\infty}\frac{K^l}{l!}-3K^2-3\sum_{l=2}^{\infty}\frac{K^{l+1}}{l!}+K^2+\sum_{l=1}^{\infty}\frac{K^{l+2}}{l!} \\
&= 4\sum_{l=3}^{\infty}\frac{K^l}{l!}-3\sum_{l=2}^{\infty}\frac{K^{l+1}}{l!}+\sum_{l=1}^{\infty}\frac{K^{l+2}}{l!} \\
&= 4\sum_{l=3}^{\infty}\frac{K^l}{l!}-3\sum_{l=3}^{\infty}\frac{K^{l+1}l}{l!}+\sum_{l=3}^{\infty}\frac{K^{l+2}l(l-1)}{l!} \\
&= \sum_{l=3}^{\infty}\frac{K^l(4-3l+l(l-1))}{l!} \\
\upsilon_1(K) &= \sum_{l=0}^{\infty}\frac{K^l(4-3(l+3)+(l+3)((l+3)-1))}{(l+3)!} \\
&= \sum_{l=0}^{\infty}\frac{K^l(l+1)^2}{(l+3)!}
\end{aligned}
$$

Next, we work on deriving a similar formulation for υ_2:

$$
\begin{aligned}
\upsilon_2(K)K^3 &= 2+K+e^K\cdot(-2+K) \\
&= 2+K+\sum_{l=0}^{\infty}\frac{K^l}{l!}(-2+K) \\
&= 2+K-2\sum_{l=0}^{\infty}\frac{K^l}{l!}+K\sum_{l=0}^{\infty}\frac{K^l}{l!} \\
&= 2+K-2-2\sum_{l=1}^{\infty}\frac{K^l}{l!}+K+K\sum_{l=1}^{\infty}\frac{K^l}{l!} \\
&= 2K-2\sum_{l=1}^{\infty}\frac{K^l}{l!}+K\sum_{l=1}^{\infty}\frac{K^l}{l!} \\
&= 2K-2K-2\sum_{l=2}^{\infty}\frac{K^l}{l!}+K^2+K\sum_{l=2}^{\infty}\frac{K^l}{l!} \\
&= -K^2-2\sum_{l=3}^{\infty}\frac{K^l}{l!}+K^2+\frac{1}{2}K^3+K\sum_{l=3}^{\infty}\frac{K^l}{l!} \\
&= \frac{1}{2}K^3+(K-2)\sum_{l=3}^{\infty}\frac{K^l}{l!} \\
\upsilon_2(K) &= \frac{1}{2}+(K-2)\sum_{l=0}^{\infty}\frac{K^l}{(l+3)!}
\end{aligned}
$$

And the same for υ_3:

$$\begin{aligned}
\upsilon_3(K)K^3 &= -4 - 3K - K^2 + e^K \cdot (4 - K) \\
&= -4 - 3K - K^2 + \sum_{l=0}^{\infty} \frac{K^l}{l!} \cdot (4 - K) \\
&= -4 - 3K - K^2 + 4\sum_{l=0}^{\infty} \frac{K^l}{l!} - K\sum_{l=0}^{\infty} \frac{K^l}{l!} \\
&= -4K - K^2 + 4\sum_{l=1}^{\infty} \frac{K^l}{l!} - K\sum_{l=1}^{\infty} \frac{K^l}{l!} \\
&= -2K^2 + 4\sum_{l=2}^{\infty} \frac{K^l}{l!} - K\sum_{l=2}^{\infty} \frac{K^l}{l!} \\
&= 4\sum_{l=3}^{\infty} \frac{K^l}{l!} - \frac{1}{2}K^3 - K\sum_{l=3}^{\infty} \frac{K^l}{l!} \\
&= -\frac{1}{2}K^3 + 4\sum_{l=3}^{\infty} \frac{K^l}{l!} - K\sum_{l=3}^{\infty} \frac{K^l}{l!} \\
&= -\frac{1}{2}K^3 + (4 - K)\sum_{l=3}^{\infty} \frac{K^l}{l!} \\
\upsilon_3(K) &= -\frac{1}{2} + (4 - K)\sum_{l=0}^{\infty} \frac{K^l}{(l+3)!}
\end{aligned}$$

SL-related functions Next, we study a particular SL-REXI formulation (see Pedro et al. “SEMI-LAGRANGIAN EXPONENTIAL INTEGRATION WITH APPLICATION TO THE ROTATING SHALLOW WATER EQUATIONS”):

$$\varphi_1^{SL}(K) = \varphi_1(-K)$$

$$\begin{aligned}
\varphi_2^{SL}(K) &= -\varphi_2(-K) + \varphi_1(-K) \\
&= -\sum_{i=0}^{\infty} \frac{(-K)^i}{(i+2)!} + \sum_{i=0}^{\infty} \frac{(-K)^i}{(i+1)!} \\
&= -\sum_{i=0}^{\infty} \frac{(-K)^i}{(i+2)!} + \sum_{i=0}^{\infty} \frac{(i+2)!(-K)^i}{(i+2)!} \\
&= \sum_{i=0}^{\infty} \frac{((i+2)! - 1)\,(-K)^i}{(i+2)!}
\end{aligned}$$

$$\varphi_3^{SL}(K) = ???$$

EPIRK [TODO], see Tokman (2011) - A new class of exponential propagation iterative methods of Runge-Kutta type (EPIRK).pdf