Appendix A

Lagrange polynomial interpolation

A.1 Introduction

In this chapter, a short introduction to Lagrange polynomial interpolation is presented. This writeup is based on the NIST Library of Mathematical Functions [?].
The nodes z_k are real or complex valued, the function values are $f_k = f(z_k)$. Given $(n+1)$ distinct points z_k with their corresponding function values f_k, the Lagrange interpolation polynomial is the unique polynomial $P_n(z)$ satisfying $P(z_k) = f_k$ while not exceeding order n, with $k = 0, 1, ..., n$. The Lagrange polynomial is given by

$$P_n(z) = \sum_{k=0}^{n} \mathscr{L}_k(z) f_k$$

with Lagrange coefficients

$$\mathscr{L}_k(z) = \prod_{j=0, j\neq k}^{n} \frac{z - z_j}{z_k - z_j}$$

where the factor for $j = k$ is omitted in the product. The Lagrange coefficients are again polynomials with the property

$$\mathscr{L}_k(z_j) = \delta_{k,j},$$

thus, each $\mathscr{L}_k(z)$ has a weight of 1 if $z = z_k$ or 0 if $z = z_j$ with $j \neq k$. For this property, $P_n(z)$ goes exactly through all data points (z_k, f_k).

A.2 Application for a simple exponential

In practice one applies a low order polynomial interpolation for a small set of points lying close to the target position, this approach guarantees smoothness of the interpolated curve even for non-smooth data, thus effectively reducing high order polynomial oscillations. For demonstrative purposes, some Lagrange polynomials (line style: solid, black) for a basic exponential function (line style: dotted, blue) are depicted below.

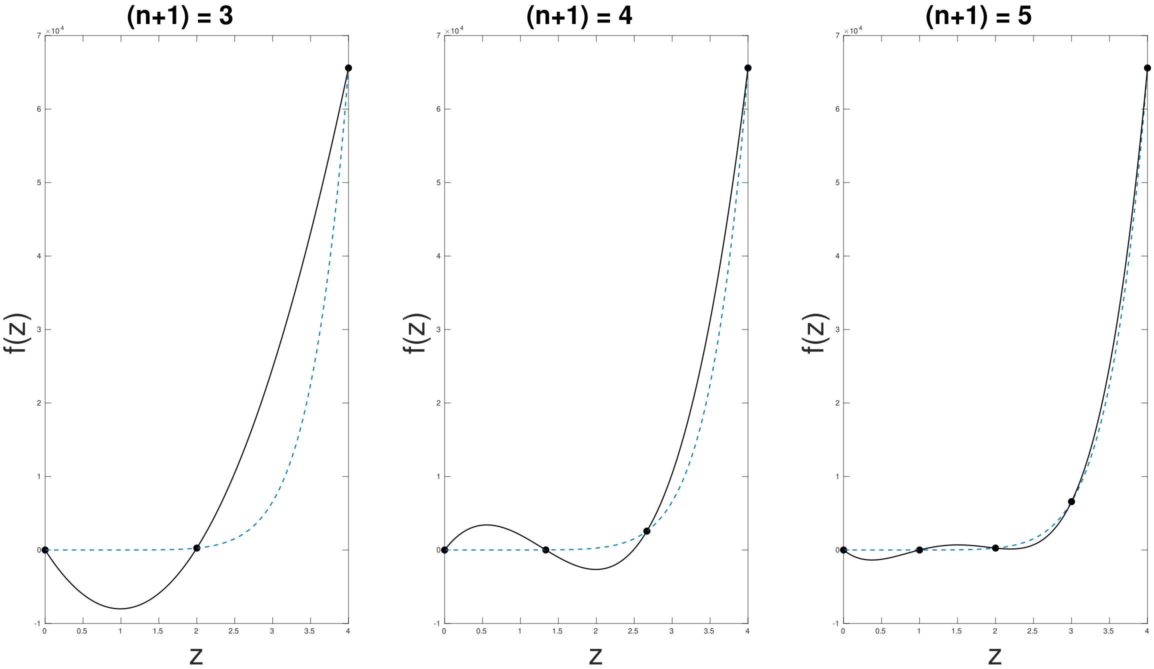

Figure A.1: Lagrange polynomials of order n with equidistant z_k for $f(z) = e^z$

Appendix B

Runge-Kutta integration

B.1 General formulation

In this chapter, a brief summary of Runge-Kutta integration is presented, based on the comprehensive summary by E. Hairer in [?]. In numerical analysis, efficient integration methods for solving initial value problems of the form $y' = f(x, y)$ were initially implemented by Euler (1768), although later further methods based on his work were developed by Runge (1895) and Kutta (1905). The most widely known algorithm is the so-called fourth order Runge-Kutta solver (commonly abbreviated *RK4*), however, an entire generalized class of integrators has since been derived. Such an integrating scheme is fully described by the coefficients of the corresponding *Butcher tableau* given in B.1:

Table B.1: *Butcher tableau* for a general s-stage *Runge-Kutta method*

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_s & a_{s1} & a_{s2} & \dots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \dots & b_{s-1} & b_s
\end{array}$$

Generally, the condition

$$c_i = \sum_{j=1}^{i-1} a_{ij} \tag{B.1}$$

is further imposed, which greatly simplifies the problem of deriving order conditions for higher order methods.

Using the coefficients of B.1 one can explicitly calculate the approximate solution to the initial value problem for a single step $\Delta x = h$ by computing

$$y_1 = y_0 + h(b_1 k_1 + \ldots + b_s k_s) \tag{B.2}$$

with

$$\begin{aligned} k_1 &= f(x_0, y_0) \\ k_2 &= f(x_0 + c_2 h, y_0 + h a_{21} k_1) \\ k_3 &= f(x_0 + c_3 h, y_0 + h(a_{31} k_1 + a_{32} k_2)) \\ &\ldots \\ k_s &= f(x_0 + c_s h, y_0 + h(a_{s1} k_1 + \ldots + a_{s,s-1} k_{s-1})). \end{aligned} \tag{B.3}$$

B.2 RK4 with application

For the specific case of the *RK4*-method, which is also implemented in the *GORILLA* code, the corresponding *Butcher tableau* is given in B.2.

Table B.2: *Butcher tableau* for the *RK4*-method

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

One is especially interested in the application of this scheme to an ODE system of the shape

$$f(\tau, \mathbf{z}(\tau)) = \mathbf{f}(\mathbf{z}(\tau)) = \frac{\mathrm{d}\mathbf{z}(\tau)}{\mathrm{d}\tau} = \hat{\mathbf{a}} \cdot \mathbf{z}(\tau) + \mathbf{b} \tag{B.4}$$

with initial conditions $\mathbf{z}(0) = \mathbf{z}_0$. Note that $\mathbf{f}(\tau, \mathbf{z}(\tau))$ does not explicitly depend on τ, thus, (B.2) and (B.3) yield for a single *RK4* step with $h = \tau$

$$\begin{aligned}
\mathbf{z}_{RK4} &= \mathbf{z}_0 + \tau \left(\frac{1}{3}k_1 + \frac{1}{6}k_2 + \frac{1}{6}k_3 + \frac{1}{3}k_4 \right) \\
k_1 &= \mathbf{f}(\mathbf{z}_0) \\
k_2 &= \mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}(\mathbf{z}_0)\right) \\
k_3 &= \mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}(\mathbf{z}_0)\right)\right) \\
k_4 &= \mathbf{f}\left(\mathbf{z}_0 + \tau\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}(\mathbf{z}_0)\right)\right)\right),
\end{aligned} \tag{B.5}$$

which allows to explicitly write the approximate *RK4*-solution for this ODE system as

$$\begin{aligned}
\mathbf{z}_{RK4}(\tau) &= \mathbf{z}_0 + \frac{\tau}{6}\mathbf{f}(\mathbf{z}_0) + \frac{\tau}{3}\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}(\mathbf{z}_0)\right) + \frac{\tau}{3}\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}(\mathbf{z}_0)\right)\right) \\
&+ \frac{\tau}{6}\mathbf{f}\left(\mathbf{z}_0 + \tau\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}\left(\mathbf{z}_0 + \frac{\tau}{2}\mathbf{f}(\mathbf{z}_0)\right)\right)\right) \ .
\end{aligned} \tag{B.6}$$

It is important to note that the *RK4*-method has the property that for sufficiently smooth functions, the approximate *RK4*-solution $\mathbf{z}_{RK4}(\tau)$ coincides with the fourth order Taylor expansion of the analytical solution for $\mathbf{z}(\tau)$. The associated errors are therefore of order $\mathcal{O}(\tau^5)$.

B.3 Runge-Kutta-Fehlberg - RK45

Although the introduced *RK4*-method is a very useful tool, its accuracy greatly depends on the chosen step size h, while the routine generally does not yield an estimate for the error, if not computed separately. A clever way to circumvent this problem was introduced by E. Fehlberg (1969), namely, to evaluate a given step successively with proposed fourth-order and fifth-order routines and then compute the difference of these two results. If the difference is smaller than a set tolerance, the step is accepted, if not, the step is discarded and the previous step size h is halved for the next attempt. The Butcher tableau for the *RK45* method is given by

Table B.3: *Butcher tableau* for the *RK45*-method

$$
\begin{array}{c|cccccc}
0 & & & & & & \\
\frac{1}{4} & \frac{1}{4} & & & & & \\
\frac{3}{8} & \frac{3}{32} & \frac{9}{32} & & & & \\
\frac{12}{13} & \frac{1932}{2197} & -\frac{7200}{2197} & & & & \\
1 & \frac{439}{216} & -8 & \frac{3680}{513} & -\frac{845}{4104} & & \\
\frac{1}{2} & -\frac{8}{27} & 2 & -\frac{3544}{2565} & \frac{1859}{4104} & -\frac{11}{40} & \\
\hline
 & \frac{16}{135} & 0 & \frac{6656}{12825} & \frac{28561}{56430} & -\frac{9}{50} & \frac{2}{55} \\
 & \frac{25}{216} & 0 & \frac{1408}{2565} & \frac{2197}{4104} & -\frac{1}{5} & 0
\end{array}
$$

Here, the first row at the bottom gives the coefficients for the fifth order method, the second row gives the coefficients for the fourth order method.