Chapter II. Runge-Kutta and Extrapolation Methods

Numerical methods for ordinary differential equations fall naturally into two classes: those which use *one* starting value at each step ("one-step methods") and those which are based on *several* values of the solution ("multistep methods" or "multi-value methods"). The present chapter is devoted to the study of one-step methods, while multistep methods are the subject of Chapter III. Both chapters can, to a large extent, be read independently of each other.

We start with the theory of Runge-Kutta methods: the derivation of order conditions with the help of labelled trees, error estimates, convergence proofs, implementation, methods of higher order, dense output. Section II.7 introduces implicit Runge-Kutta methods. More attention will be drawn to these methods in Volume II on stiff differential equations. Two sections then discuss the elegant idea of *extrapolation* (Richardson, Romberg, etc) and its use in obtaining high order codes. The methods presented are then tested and compared on a series of problems. The potential of parallelism is discussed in a separate section. We then turn our attention to an algebraic theory of the composition of methods. This will be the basis for the study of order properties for many general classes of methods in the following chapter. The chapter ends with special methods for second order differential equations $y'' = f(x,y)$, for Hamiltonian systems (symplectic methods) and for problems with delay.

We illustrate the methods of this chapter with an example from Astronomy, the restricted three body problem. One considers two bodies of masses $1-\mu$ and μ in circular rotation in a plane and a third body of negligible mass moving around in the same plane. The equations are (see e.g., the classical textbook Szebehely 1967)

$$\begin{aligned}
y_1'' &= y_1 + 2y_2' - \mu' \frac{y_1+\mu}{D_1} - \mu \frac{y_1-\mu'}{D_2},\\
y_2'' &= y_2 - 2y_1' - \mu' \frac{y_2}{D_1} - \mu \frac{y_2}{D_2},\\
D_1 &= ((y_1+\mu)^2 + y_2^2)^{3/2}, \qquad D_2 = ((y_1-\mu')^2 + y_2^2)^{3/2},\\
\mu &= 0.012277471, \qquad \mu' = 1-\mu\,.
\end{aligned} \tag{0.1}$$

There exist initial values, for example

$$\begin{aligned} y_1(0) &= 0.994\,, \qquad y_1'(0) = 0\,, \qquad y_2(0) = 0\,, \\ y_2'(0) &= -2.00158510637908252240537862224\,, \qquad (0.2) \\ x_{\text{end}} &= 17.0652165601579625588917206249\,, \end{aligned}$$

such that the solution is periodic with period x_{end}. Such periodic solutions have fascinated astronomers and mathematicians for many decades (Poincaré; extensive numerical calculations are due to Sir George Darwin (1898)) and are now often called "Arenstorf orbits" (see Arenstorf (1963) who did numerical computations "on high speed electronic computers"). The problem is $\mathcal{C}^\infty$ with the exception of the two singular points $y_1 = -\mu$ and $y_1 = 1-\mu$, $y_2 = 0$, therefore the Euler polygons of Section I.7 are known to converge to the solution. But are they really numerically useful here? We have chosen 24000 steps of step length $h = x_{\text{end}}/24000$ and plotted the result in Figure 0.1. The result is not very striking.

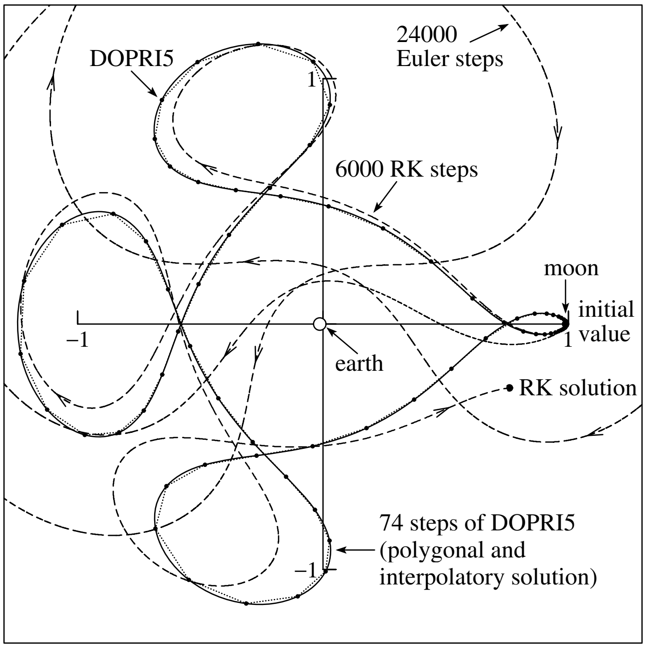

Fig. 0.1. An Arenstorf orbit computed by equidistant Euler, equidistant Runge-Kutta and variable step size Dormand & Prince

The performance of the Runge-Kutta method (left tableau of Table 1.2) is already much better and converges faster to the solution. We have used 6000 steps of step size $x_{\text{end}}/6000$, so that the numerical work becomes equivalent. Clearly, most accuracy is lost in those parts of the orbit which are close to a singularity. Therefore, codes with automatic step size selection, described in Section II.4, perform much better and the code DOPRI5 (Table 5.2) computes the orbit with a precision of 10^{-3} in 98 steps (74 accepted and 24 rejected). The step size becomes very large in some regions and the graphical representation as polygons connecting the solution points becomes unsatisfactory. The solid line is the interpolatory solution (Section II.6), which is also precise for all intermediate values and useful for many other questions such as delay differential equations, event location or discontinuities in the differential equation.

For still higher precision one needs methods of higher order. For example, the code DOP853 (Section II.5) computes the orbit faster than DOPRI5 for more stringent tolerances, say smaller than about 10^{-6}. The highest possible order is obtained by extrapolation methods (Section II.9) and the code ODEX (with $K_{\text{max}} = 15$) obtains the orbit with a precision of 10^{-30} with about 25000 function evaluations, precisely the same amount of work as for the above Euler solution.