# Chapter 13
# Equations of Motion

*Simulation of a physical system means to calculate the time evolution of a model system in many cases. We consider a large class of models which can be described by a first order initial value problem*

$$\frac{dY}{dt} = f(Y(t), t) \quad Y(t = 0) = Y_0 \tag{13.1}$$

*where $Y$ is the state vector (possibly of very high dimension) which contains all information about the system. Our goal is to calculate the time evolution of the state vector $Y(t)$ numerically. For obvious reasons this can be done only for a finite number of values of $t$ and we have to introduce a grid of discrete times $t_n$ which for simplicity are assumed to be equally spaced*[1]*:*

$$t_{n+1} = t_n + \Delta t. \tag{13.2}$$

*Advancing time by one step involves the calculation of the integral*

$$Y(t_{n+1}) - Y(t_n) = \int_{t_n}^{t_{n+1}} f(Y(t'), t')dt' \tag{13.3}$$

*which can be a formidable task since $f(Y(t), t)$ depends on time via the time dependence of all the elements of $Y(t)$. In this chapter we discuss several strategies for the time integration. The explicit Euler forward difference has low error order but is useful as a predictor step for implicit methods. A symmetric difference quotient is much more accurate. It can be used as the corrector step in combination with an explicit Euler predictor step and is often used for the time integration of partial differential equations. Methods with higher error order can be obtained from a Taylor series expansion, like the Nordsieck and Gear predictor-corrector methods which have been often applied in molecular dynamics calculations. Runge–Kutta methods are very important for ordinary differential equations. They are robust and allow an adaptive*

[1] *Control of the step width will be discussed later.*



P.O.J. Scherer, *Computational Physics*, Graduate Texts in Physics,
DOI 10.1007/978-3-319-61088-7_13

*control of the step size. Very accurate results can be obtained for ordinary differential equations with extrapolation methods like the famous Gragg-Bulirsch-Stoer method. If the solution is smooth enough, multistep methods are applicable, which use information from several points. Most known are Adams-Bashforth–Moulton methods and Gear methods (also known as backward differentiation methods), which are especially useful for stiff problems. The class of Verlet methods has been developed for molecular dynamics calculations. They are symplectic and time reversible and conserve energy over long trajectories.*

## 13.1 The State Vector

The state of a classical N-particle system is given by the position in phase space, or equivalently by specifying position and velocity for all the N particles

$$Y = (\mathbf{r}_1, \mathbf{v}_1, \ldots, \mathbf{r}_N, \mathbf{v}_N)\,. \tag{13.4}$$

The concept of a state vector is not restricted to a finite number of degrees of freedom. For instance a diffusive system can be described by the particle concentrations as a function of the coordinate, i.e. the elements of the state vector are now indexed by the continuous variable $\mathbf{x}$

$$Y = (c_1(\mathbf{x}), \ldots c_M(\mathbf{x}))\,. \tag{13.5}$$

Similarly, a quantum particle moving in an external potential can be described by the amplitude of the wave function

$$Y = (\Psi(\mathbf{x})). \tag{13.6}$$

Numerical treatment of continuous systems is not feasible since even the ultimate high end computer can only handle a finite number of data in finite time. Therefore discretization is necessary (Chap. 12), by introducing a spatial mesh (Sects. 12.2, 12.3, 12.6), which in the simplest case means a grid of equally spaced points

$$\mathbf{x}_{ijk} = (ih, jh, kh) \quad i = 1..i_{\max}, j = 1..j_{\max}, k = 1..k_{\max} \tag{13.7}$$

$$Y = \left(c_1(\mathbf{x}_{ijk}) \ldots c_M(\mathbf{x}_{ijk})\right) \tag{13.8}$$

$$Y = \left(\Psi(\mathbf{x}_{ijk})\right) \tag{13.9}$$

or by expanding the continuous function with respect to a finite set of basis functions (Sect. 12.5). The elements of the state vector then are the expansion coefficients

$$|\Psi> = \sum_{s=1}^{N} C_s |\Psi_s> \tag{13.10}$$

$$Y = (C_1, \ldots, C_N)\,. \tag{13.11}$$

If the density matrix formalism is used to take the average over a thermodynamic ensemble or to trace out the degrees of freedom of a heat bath, the state vector instead is composed of the elements of the density matrix

$$\rho = \sum_{s=1}^{N}\sum_{s'=1}^{N} \rho_{ss'} |\Psi_s><\Psi_{s'}| = \sum_{s=1}^{N}\sum_{s'=1}^{N} \overline{C^*_{s'} C_s} |\Psi_s><\Psi_{s'}| \tag{13.12}$$

$$Y = (\rho_{11} \cdots \rho_{1N}, \rho_{21} \cdots \rho_{2N}, \ldots, \rho_{N1} \cdots \rho_{NN})\,. \tag{13.13}$$

## 13.2 Time Evolution of the State Vector

We assume that all information about the system is included in the state vector. Then the simplest equation to describe the time evolution of the system gives the change of the state vector

$$\frac{\mathrm{d}Y}{\mathrm{d}t} = f(Y, t) \tag{13.14}$$

as a function of the state vector (or more generally a functional in the case of a continuous system). Explicit time dependence has to be considered for instance to describe the influence of an external time dependent field.

Some examples will show the universality of this equation of motion:

- N-particle system

The motion of N interacting particles is described by

$$\frac{\mathrm{d}Y}{\mathrm{d}t} = (\dot{\mathbf{r}}_1, \dot{\mathbf{v}}_1 \cdots) = (\mathbf{v}_1, \mathbf{a}_1 \cdots) \tag{13.15}$$

where the acceleration of a particle is given by the total force acting upon this particle and thus depends on all the coordinates and eventually time (velocity dependent forces could be also considered but are outside the scope of this book)

$$\mathbf{a}_i = \frac{\mathbf{F}_i(\mathbf{r}_1 \cdots \mathbf{r}_N, t)}{m_i}. \tag{13.16}$$

- Diffusion

Heat transport and other diffusive processes are described by the diffusion equation

$$\frac{\partial f}{\partial t} = D\Delta f + S(\mathbf{x}, t) \tag{13.17}$$

which in its simplest spatially discretized version for 1-dimensional diffusion reads

$$\frac{\partial f(\mathbf{x}_i)}{\partial t} = \frac{D}{\Delta x^2}\left(f(\mathbf{x}_{i+1}) + f(\mathbf{x}_{i-1}) - 2f(\mathbf{x}_i)\right) + S(\mathbf{x}_i, t). \tag{13.18}$$

- Waves

Consider the simple 1-dimensional wave equation

$$\frac{\partial^2 f}{\partial t^2} = c^2 \frac{\partial^2 f}{\partial x^2} \tag{13.19}$$

which by introducing the velocity $g(\mathbf{x}) = \frac{\partial}{\partial t} f(\mathbf{x})$ as an independent variable can be rewritten as

$$\frac{\partial}{\partial t}\left(f(\mathbf{x}), g(\mathbf{x})\right) = \left(g(\mathbf{x}), c^2 \frac{\partial^2}{\partial x^2} f(\mathbf{x})\right). \tag{13.20}$$

Discretization of space gives

$$\frac{\partial}{\partial t}\left(f(\mathbf{x}_i), g(\mathbf{x}_i)\right) = \left(g(\mathbf{x}_i), \frac{c^2}{\Delta x^2}\left(f(\mathbf{x}_{i+1}) + f(\mathbf{x}_{i-1}) - 2f(\mathbf{x}_i)\right)\right). \tag{13.21}$$

- two-state quantum system

The Schroedinger equation for a two level system (for instance a spin-1/2 particle in a magnetic field) reads

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} C_1 \\ C_2 \end{pmatrix} = \begin{pmatrix} H_{11}(t)\ H_{12}(t) \\ H_{21}(t)\ H_{22}(t) \end{pmatrix}\begin{pmatrix} C_1 \\ C_2 \end{pmatrix}. \tag{13.22}$$

## 13.3 Explicit Forward Euler Method

The simplest method which is often discussed in elementary physics textbooks approximates the integrand by its value at the lower bound (Fig. 13.1):

$$Y(t_{n+1}) - Y(t_n) \approx f(Y(t_n), t_n)\Delta t. \tag{13.23}$$

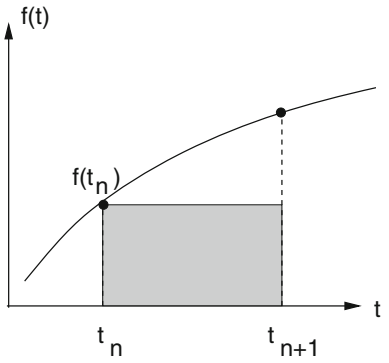


**Fig. 13.1** Explicit Euler method

The truncation error can be estimated from a Taylor series expansion

$$Y(t_{n+1}) - Y(t_n) = \Delta t \frac{\mathrm{d}Y}{\mathrm{d}t}(t_n) + \frac{\Delta t^2}{2}\frac{\mathrm{d}^2 Y}{\mathrm{d}t^2}(t_n) + \cdots$$
$$= \Delta t f(Y(t_n), t_n) + O(\Delta t^2). \tag{13.24}$$

The explicit Euler method has several serious drawbacks

- low error order

Suppose you want to integrate from the initial time $t_0$ to the final time $t_0 + T$. For a time step of $\Delta t$ you have to perform $N = T/\Delta t$ steps. Assuming comparable error contributions from all steps the global error scales as $N\Delta t^2 = O(\Delta t)$. The error gets smaller as the time step is reduced but it may be necessary to use very small $\Delta t$ to obtain meaningful results.

- loss of orthogonality and normalization

The simple Euler method can produce systematic errors which are very inconvenient if you want, for instance, to calculate the orbits of a planetary system. This can be most easily seen from a very simple example. Try to integrate the following equation of motion (see Example 1.5 on p. 13):

$$\frac{\mathrm{d}z}{\mathrm{d}t} = \mathrm{i}\omega z. \tag{13.25}$$

The exact solution is obviously given by a circular orbit in the complex plane:

$$z = z_0 e^{\mathrm{i}\omega t} \tag{13.26}$$

$$|z| = |z_0| = \text{ const.} \tag{13.27}$$

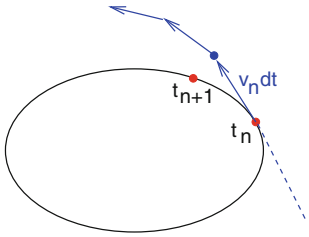


**Fig. 13.2** Systematic errors of the Euler method

Application of the Euler method gives

$$z(t_{n+1}) = z(t_n) + \mathrm{i}\omega\Delta t\, z(t_n) = (1 + \mathrm{i}\omega\Delta t) z(t_n) \tag{13.28}$$

and you find immediately

$$|z(t_n)| = \sqrt{1 + \omega^2\Delta t^2}\, |z(t_{n-1})| = \left(1 + \omega^2\Delta t^2\right)^{n/2} |z(t_0)| \tag{13.29}$$

which shows that the radius increases continually even for the smallest time step possible (Fig. 13.2).

The same kind of error appears if you solve the Schroedinger equation for a particle in an external potential or if you calculate the rotational motion of a rigid body. For the N-body system it leads to a violation of the conservation of phase space volume. This can introduce an additional sensitivity of the calculated results to the initial conditions. Consider a harmonic oscillator with the equation of motion

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} x(t) \\ v(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ -\omega^2 x(t) \end{pmatrix}. \tag{13.30}$$

Application of the explicit Euler method gives

$$\begin{pmatrix} x(t+\Delta t) \\ v(t+\Delta t) \end{pmatrix} = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix} + \begin{pmatrix} v(t) \\ -\omega^2 x(t) \end{pmatrix} \Delta t. \tag{13.31}$$

The change of the phase space volume (Fig. 13.3) is given by the Jacobi determinant

$$J = \left| \frac{\partial(x(t+\Delta t), v(t+\Delta t))}{\partial(x(t), v(t))} \right| = \begin{vmatrix} 1 & \Delta t \\ -\omega^2\Delta t & 1 \end{vmatrix} = 1 + (\omega\Delta t)^2. \tag{13.32}$$

In this case the phase space volume increases continuously.

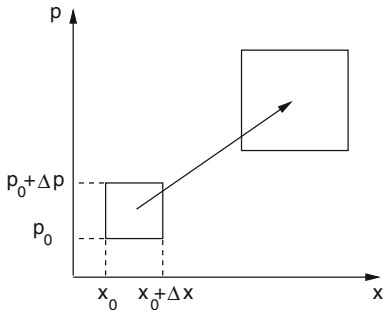


**Fig. 13.3** Time evolution of the phase space volume

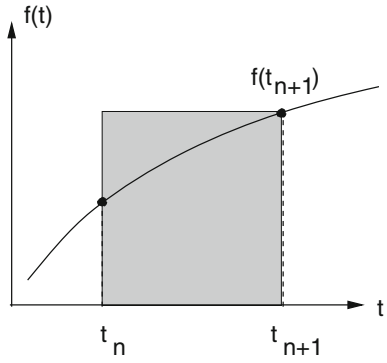


**Fig. 13.4** Implicit backward Euler method

## 13.4 Implicit Backward Euler Method

Alternatively let us make a step backwards in time

$$Y(t_n) - Y(t_{n+1}) \approx -f(Y(t_{n+1}), t_{n+1})\Delta t \tag{13.33}$$

which can be written as (Fig. 13.4)

$$Y(t_{n+1}) \approx Y(t_n) + f(Y(t_{n+1}), t_{n+1})\Delta t. \tag{13.34}$$

Taylor series expansion gives

$$Y(t_n) = Y(t_{n+1}) - \frac{\mathrm{d}}{\mathrm{d}t}Y(t_{n+1})\Delta t + \frac{\mathrm{d}^2}{\mathrm{d}t^2}Y(t_{n+1})\frac{\Delta t^2}{2} + \cdots \tag{13.35}$$

which shows that the error order again is $O(\Delta t^2)$. The implicit method is sometimes used to avoid the inherent instability of the explicit method. For the examples in

Sect. 13.3 it shows the opposite behavior. The radius of the circular orbit as well as the phase space volume decrease in time. The gradient at future time has to be estimated before an implicit step can be performed.

## 13.5 Improved Euler Methods

The quality of the approximation can be improved significantly by employing the midpoint rule (Fig. 13.5)

$$Y(t_{n+1}) - Y(t_n) \approx f\left(Y\left(t + \frac{\Delta t}{2}\right), t_n + \frac{\Delta t}{2}\right)\Delta t. \tag{13.36}$$

The error is smaller by one order of $\Delta t$:

$$\begin{aligned} & Y(t_n) + f\left(Y\left(t + \frac{\Delta t}{2}\right), t_n + \frac{\Delta t}{2}\right)\Delta t \\ & = Y(t_n) + \left(\frac{\mathrm{d}Y}{\mathrm{d}t}(t_n) + \frac{\Delta t}{2}\frac{\mathrm{d}^2 Y}{\mathrm{d}t^2}(t_n) + \cdots\right)\Delta t \\ & = Y(t_n + \Delta t) + O(\Delta t^3). \end{aligned} \tag{13.37}$$

The future value $Y(t + \frac{\Delta t}{2})$ can be obtained by two different approaches:

- predictor-corrector method

Since $f(Y(t + \frac{\Delta t}{2}), t_n + \frac{\Delta t}{2})$ is multiplied with $\Delta t$, it is sufficient to use an approximation with lower error order. Even the explicit Euler step is sufficient. Together the following algorithm results:

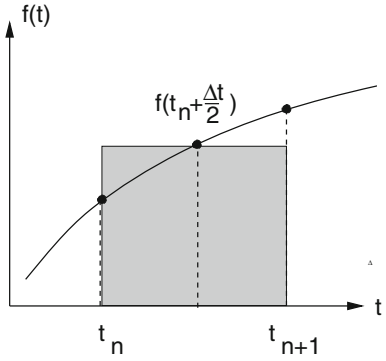


**Fig. 13.5** Improved Euler method

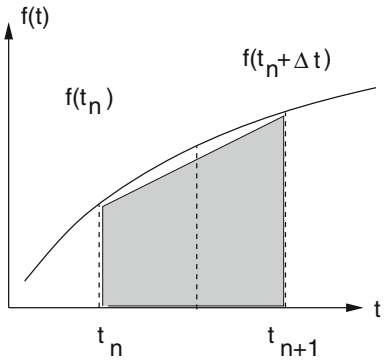


**Fig. 13.6** Improved polygon (or Heun) method

$$\begin{aligned} &\text{predictor step:} \quad Y^{(p)} = Y(t_n) + \tfrac{\Delta t}{2} f(Y(t_n), t_n) \\ &\text{corrector step:} \quad Y(t_n + \Delta t) = Y(t_n) + \Delta t f(Y^{(p)}, t_n + \tfrac{\Delta t}{2}). \end{aligned} \tag{13.38}$$

- averaging (Heun method)

The average of $f(Y(t_n), t_n)$ and $f(Y(t_n + \Delta t), t + \Delta t)$ is another approximation to the midpoint value of comparable quality (Fig. 13.6).

Expansion around $t_n + \Delta t/2$ gives

$$\begin{aligned} &\frac{1}{2}\left(f(Y(t_n), t_n) + f(Y(t_n + \Delta t), t + \Delta t)\right) \\ &= f\left(Y\left(t_n + \frac{\Delta t}{2}\right), t_n + \frac{\Delta t}{2}\right) + O(\Delta t^2). \end{aligned} \tag{13.39}$$

Inserting the average in (13.36) gives the following algorithm, which is also known as improved polygon method and corresponds to the trapezoidal rule for the integral (4.13) or to a combination of explicit and implicit Euler step:

$$Y(t_n + \Delta t) = Y(t_n) + \frac{\Delta t}{2}\left(f(Y(t_n), t_n) + f(Y(t_n + \Delta t), t + \Delta t)\right). \tag{13.40}$$

In the special case of a linear function $f(Y(t), t) = F\,Y(t)$ (for instance rotational motion or diffusion) this can be solved formally by

$$Y(t_n + \Delta t) = \left(1 - \frac{\Delta t}{2}F\right)^{-1}\left(1 + \frac{\Delta t}{2}F\right) Y(t_n). \tag{13.41}$$

Numerically it is not necessary to perform the matrix inversion. Instead a linear system of equations is solved:

$$\left(1 - \frac{\Delta t}{2}F\right) Y(t_n + \Delta t) = \left(1 + \frac{\Delta t}{2}F\right) Y(t_n). \tag{13.42}$$

In certain cases the Heun method conserves the norm of the state vector, for instance if $F$ has only imaginary eigenvalues (as for the 1-dimensional Schroedinger equation, see p. 526).

In the general case a predictor step has to be made to estimate the state vector at $t_n + \Delta t$ before the Heun expression (13.40) can be evaluated:

$$Y^{(p)} = Y(t_n) + \Delta t f(Y(t_n), t_n). \tag{13.43}$$

## 13.6 Taylor Series Methods

Higher order methods can be obtained from a Taylor series expansion

$$Y(t_n + \Delta t) = Y(t_n) + \Delta t f(Y(t_n), t_n) + \frac{\Delta t^2}{2} \frac{\mathrm{d}f(Y(t_n), t_n)}{\mathrm{d}t} + \cdots . \tag{13.44}$$

The total time derivative can be expressed as

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial Y}\frac{\mathrm{d}Y}{\mathrm{d}t} + \frac{\partial f}{\partial t} = f'f + \dot{f} \tag{13.45}$$

where the partial derivatives have been abbreviated in the usual way by $\frac{\partial f}{\partial t} = \dot{f}$ and $\frac{\partial f}{\partial Y} = f'$. Higher derivatives are given by

$$\frac{\mathrm{d}^2 f}{\mathrm{d}t^2} = f''f^2 + f'^2 f + 2\dot{f}'f + \ddot{f} \tag{13.46}$$

$$\frac{\mathrm{d}^3 f}{\mathrm{d}t^3} = \frac{\partial^3 f}{\partial t^3} + f'''f^3 + 3\dot{f}''f^2 + \ddot{f}f' + 3f''\dot{f}f + 3\dot{f}' + 4f''f'f^2 + 5\dot{f}'f'f + f'^3 f + f'^2\dot{f}. \tag{13.47}$$

### *13.6.1 Nordsieck Predictor-Corrector Method*

Nordsieck [151] determines an interpolating polynomial of degree $m$. As variables he uses the 0th to $m$th derivatives[2] evaluated at the current time $t$, for instance for $m = 5$ he uses the variables

[2] In fact the derivatives of the interpolating polynomial which exist even if higher derivatives of $f$ do not exist.

$$Y(t) \tag{13.48}$$

$$g(t) = \frac{\mathrm{d}}{\mathrm{d}t} Y(t) \tag{13.49}$$

$$a(t) = \frac{\Delta t}{2} \frac{\mathrm{d}^2}{\mathrm{d}t^2} Y(t) \tag{13.50}$$

$$b(t) = \frac{\Delta t^2}{6} \frac{\mathrm{d}^3}{\mathrm{d}t^3} Y(t) \tag{13.51}$$

$$c(t) = \frac{\Delta t^3}{24} \frac{\mathrm{d}^4}{\mathrm{d}t^4} Y(t) \tag{13.52}$$

$$d(t) = \frac{\Delta t^4}{120} \frac{\mathrm{d}^5}{\mathrm{d}t^5} Y(t). \tag{13.53}$$

Taylor expansion gives approximate values at $t + \Delta t$

$$\begin{aligned} Y(t + \Delta t) &= Y(t) + \Delta t \left[ g(t) + a(t) + b(t) + c(t) + d(t) + e(t) \right] \\ &= Y^p(t + \Delta t) + e(t) \Delta t \end{aligned} \tag{13.54}$$

$$g(t+\Delta t) = g(t)+2a(t)+3b(t)+4c(t)+5d(t)+6e(t) = g^p(t+\Delta T)+6e(t) \tag{13.55}$$

$$a(t + \Delta t) = a(t) + 3b(t) + 6c(t) + 10d(t) + 15e(t) = a^p(t + \Delta t) + 15e(t) \tag{13.56}$$

$$b(t + \Delta t) = b(t) + 4c(t) + 10d(t) + 20e(t) = b^p(t + \Delta t) + 20e(t) \tag{13.57}$$

$$c(t + \Delta t) = c(t) + 5d(t) + 15e(t) = c^p(t + \Delta t) + 15e(t) \tag{13.58}$$

$$d(t + \Delta t) = d(t) + 6e(t) = d^p(t + \Delta t) + 6e(t) \tag{13.59}$$

where the next term of the Taylor series $e(t) = \frac{\Delta t^5}{6!} \frac{\mathrm{d}^6}{\mathrm{d}t^6} Y(t)$ has been introduced as an approximation to the truncation error of the predicted values $Y^p$, $g^p$, etc. It can be estimated from the second equation

$$e = \frac{1}{6} \left[ f(Y^p(t + \Delta t), t + \Delta t) - g^p(t + \Delta t) \right] = \frac{1}{6} \delta f. \tag{13.60}$$

This predictor-corrector method turns out to be rather unstable. However, stability can be achieved by slightly modifying the coefficients of the corrector step. Nordsieck suggested to use

$$Y(t+\Delta t) = Y^p(t+\Delta t) + \frac{95}{288}\delta f \tag{13.61}$$

$$a(t+\Delta t) = a^p(t+\Delta t) + \frac{25}{24}\delta f \tag{13.62}$$

$$b(t+\Delta t) = b^p(t+\Delta t) + \frac{35}{72}\delta f \tag{13.63}$$

$$c(t+\Delta t) = c^p(t+\Delta t) + \frac{5}{48}\delta f \tag{13.64}$$

$$d(t+\Delta t) = d^p(t+\Delta t) + \frac{1}{120}\delta f\,. \tag{13.65}$$

### *13.6.2 Gear Predictor-Corrector Methods*

Gear [152] designed special methods for molecular dynamics simulations (Chap. 15) where Newton's law (13.15) has to be solved numerically. He uses again a truncated Taylor expansion for the predictor step

$$\mathbf{r}(t+\Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \mathbf{a}(t)\frac{\Delta t^2}{2} + \dot{\mathbf{a}}(t)\frac{\Delta t^3}{6} + \ddot{\mathbf{a}}(t)\frac{\Delta t^4}{24} + \cdots \tag{13.66}$$

$$\mathbf{v}(t+\Delta t) = \mathbf{v}(t) + \mathbf{a}(t)\Delta t + \dot{\mathbf{a}}(t)\frac{\Delta t^2}{2} + \ddot{\mathbf{a}}(t)\frac{\Delta t^3}{6} + \cdots \tag{13.67}$$

$$\mathbf{a}(t+\Delta t) = \mathbf{a}(t) + \dot{\mathbf{a}}(t)\Delta t + \ddot{\mathbf{a}}(t)\frac{\Delta t^2}{2} + \cdots \tag{13.68}$$

$$\dot{\mathbf{a}}(t+\Delta t) = \dot{\mathbf{a}}(t) + \ddot{\mathbf{a}}(t)\Delta t + \cdots \tag{13.69}$$

$$\vdots$$

to calculate new coordinates etc. $\mathbf{r}^p_{n+1}, \mathbf{v}^p_{n+1}, \mathbf{a}^p_{n+1} \ldots$ (Fig. 13.7). The difference between the predicted acceleration and that calculated using the predicted coordinates

$$\delta\mathbf{a}_{n+1} = \mathbf{a}(\mathbf{r}^P_{n+1}, t+\Delta t) - \mathbf{a}^p_{n+1} \tag{13.70}$$

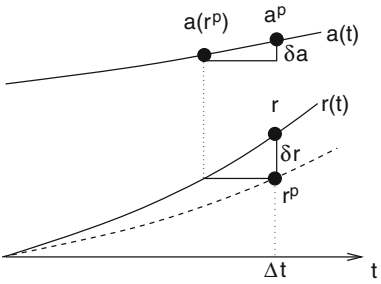


**Fig. 13.7** (Gear Predictor Corrector Method) The difference between predicted acceleration $\mathbf{a}^p$ and acceleration calculated for the predicted coordinates $\mathbf{a}(\mathbf{r}^p)$ is used as a measure of the error to estimate the correction $\delta\mathbf{r}$

is then used as a measure of the error to correct the predicted values according to

$$\mathbf{r}_{n+1} = \mathbf{r}^p_{n+1} + c_1\delta\mathbf{a}_{n+1} \tag{13.71}$$

$$\mathbf{v}_{n+1} = \mathbf{v}^p_{n+1} + c_2\delta\mathbf{a}_{n+1} \tag{13.72}$$

$$\vdots$$

The coefficients $c_i$ were determined to optimize stability and accuracy. For instance the fourth order Gear corrector reads

$$\mathbf{r}_{n+1} = \mathbf{r}^p_{n+1} + \frac{\Delta t^2}{12}\delta\mathbf{a}_{n+1} \tag{13.73}$$

$$\mathbf{v}_{n+1} = \mathbf{v}^p_{n+1} + \frac{5\Delta t}{12}\delta\mathbf{a}_{n+1} \tag{13.74}$$

$$\dot{\mathbf{a}}_{n+1} = \dot{\mathbf{a}}_n + \frac{1}{\Delta t}\delta\mathbf{a}_{n+1}. \tag{13.75}$$

Gear methods are generally not time reversible and show systematic energy drifts. A reversible symplectic predictor-corrector method has been presented recently by Martyna and Tuckerman [153].

## 13.7 Runge–Kutta Methods

If higher derivatives are not so easily available, they can be approximated by numerical differences. $f$ is evaluated at several trial points and the results are combined to reproduce the Taylor series as close as possible [154].

### 13.7.1 Second Order Runge–Kutta Method

Let us begin with two function values. As common in the literature we will denote the function values as $K_1, K_2, \ldots$. From the gradient at time $t_n$

$$K_1 = f_n = f(Y(t_n), t_n) \tag{13.76}$$

we estimate the state vector at time $t_n + \Delta t$ as

$$Y(t_n + \Delta t) \approx \Delta t\, K_1. \tag{13.77}$$

The gradient at time $t_n + \Delta t$ is approximately

$$K_2 = f(Y(t_n) + \Delta t\, K_1, t_n + \Delta t) \tag{13.78}$$

which has the Taylor series expansion

$$K_2 = f_n + (\dot{f}_n + f_n' f_n)\Delta t + \cdots \tag{13.79}$$

and application of the trapezoidal rule (4.13) gives the 2nd order Runge–Kutta method

$$Y_{n+1} = Y_n + \frac{\Delta t}{2}(K_1 + K_2) \tag{13.80}$$

which in fact coincides with the improved Euler or Heun method. Taylor series expansion shows how the combination of $K_1$ and $K_2$ leads to an expression of higher error order:

$$\begin{aligned} Y_{n+1} &= Y_n + \frac{\Delta t}{2}(f_n + f_n + (\dot{f}_n + f_n' f_n)\Delta t + \cdots) \\ &= Y_n + f_n \Delta t + \frac{\mathrm{d}f_n}{\mathrm{d}t}\frac{\Delta t^2}{2} + \cdots . \end{aligned} \tag{13.81}$$

### 13.7.2 Third Order Runge–Kutta Method

The accuracy can be further improved by calculating one additional function value at mid-time. From (13.76) we estimate the gradient at mid-time by

$$\begin{aligned} K_2 &= f\left(Y(t) + \frac{\Delta t}{2}K_1, t + \frac{\Delta t}{2}\right) \\ &= f_n + (\dot{f}_n + f_n' f_n)\frac{\Delta t}{2} + (\ddot{f}_n + f_n'' f_n^2 + 2\dot{f}_n' f_n)\frac{\Delta t^2}{8} + \cdots . \end{aligned} \tag{13.82}$$

The gradient at time $t_n + \Delta t$ is then estimated as

$$\begin{aligned} K_3 &= f(Y(t_n) + \Delta t(2K_2 - K_1), t_n + \Delta t) \\ &= f_n + \dot{f}_n \Delta t + f_n'(2K_2 - K_1)\Delta t + \ddot{f}_n \frac{\Delta t^2}{2} \\ &+ f_n'' \frac{(2K_2 - K_1)^2 \Delta t^2}{2} + 2\dot{f}_n' \frac{(2K_2 - K_1)\Delta t^2}{2} + \cdots . \end{aligned} \tag{13.83}$$

Inserting the expansion (13.82) gives the leading terms

$$K_3 = f_n + (\dot{f}_n + f_n' f_n)\Delta t + (2f_n'^2 f_n + f_n'' f_n^2 + \ddot{f}_n + 2f_n' \dot{f}_n + 2\dot{f}_n^2) \frac{\Delta t^2}{2} + \cdots . \tag{13.84}$$

Applying Simpson's rule (4.14) we combine the three gradients to get the 3rd order Runge–Kutta method

$$Y_{n+1} = Y(t_n) + \frac{\Delta t}{6}(K_1 + 4K_2 + K_3) \tag{13.85}$$

where the Taylor series

$$\begin{aligned} Y_{n+1} &= Y(t_n) + \frac{\Delta t}{6}\left(6f_n + 3(\dot{f}_n + f_n f_n')\Delta t \right. \\ &\left. + (f_n'^2 f_n + f_n'' f_n^2 + 2\dot{f}_n' f_n + f_n + \dot{}_n f_n')\ddot{}\Delta t^2 + \cdot\cdot f\cdot\right) \\ &= Y(t_n + \Delta t) + O(\Delta t^4) \end{aligned} \tag{13.86}$$

recovers the exact Taylor series (13.44) including terms of order $O(\Delta t^3)$.

### 13.7.3 Fourth Order Runge–Kutta Method

The 4th order Runge–Kutta method (RK4) is often used because of its robustness and accuracy. It uses two different approximations for the midpoint

$$\begin{aligned} K_1 &= f(Y(t_n), t_n) \\ K_2 &= f\left(Y(t_n) + \frac{K_1}{2}\Delta t, t_n + \frac{\Delta t}{2}\right) \\ K_3 &= f\left(Y(t_n) + \frac{K_2}{2}\Delta t, t_n + \frac{\Delta t}{2}\right) \\ K_4 &= f(Y(t_n) + K_3 \Delta t, t_n + \Delta t) \end{aligned}$$

and Simpson's rule (4.14) to obtain

$$Y_{n+1} = Y(t_n) + \frac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4) = Y(t_n + \Delta t) + O(\Delta t^5).$$

Expansion of the Taylor series is cumbersome but with the help of an algebra program one can easily check that the error is of order $\Delta t^5$.

## 13.8 Quality Control and Adaptive Step Size Control

For practical applications it is necessary to have an estimate for the local error and to adjust the step size properly. With the Runge Kutta method this can be achieved by a step doubling procedure. We calculate $y_{n+2}$ first by two steps $\Delta t$ and then by one step $2\Delta t$. This needs 11 function evaluations as compared to 8 for the smaller step size only (Fig. 13.8). For the 4th order method we estimate the following errors:

$$\Delta\left(Y_{n+2}^{(\Delta t)}\right) = 2a\Delta t^5 \tag{13.87}$$

$$\Delta\left(Y_{n+2}^{(2\Delta t)}\right) = a(2\Delta t)^5. \tag{13.88}$$

The local error can be estimated from

$$|Y_{n+2}^{(\Delta t)} - Y_{n+2}^{(2\Delta t)}| = 30|a|\Delta t^5$$

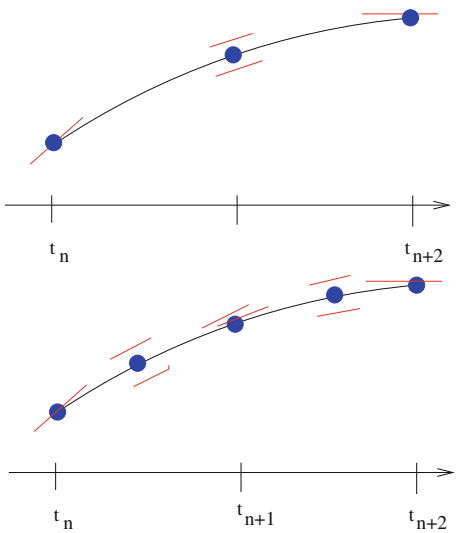


**Fig. 13.8** Step doubling with the fourth order Runge–Kutta method

$$\Delta\left(Y_{n+1}^{(\Delta t)}\right) = a\Delta t^5 = \frac{|Y_{n+2}^{(\Delta t)} - Y_{n+2}^{(2\Delta t)}|}{30}.$$

The step size $\Delta t$ can now be adjusted to keep the local error within the desired limits.

## 13.9 Extrapolation Methods

Application of the extrapolation method to calculate the integral $\int_{t_n}^{t_{n+1}} f(t)dt$ produces very accurate results but can also be time consuming. The famous Gragg-Bulirsch-Stoer method [2] starts from an explicit midpoint rule with a special starting procedure. The interval $\Delta t$ is divided into a sequence of N sub-steps

$$h = \frac{\Delta t}{N}. \tag{13.89}$$

First a simple Euler step is performed

$$\begin{aligned} u_0 &= Y(t_n) \\ u_1 &= u_0 + hf(u_0, t_n) \end{aligned} \tag{13.90}$$

and then the midpoint rule is applied repeatedly to obtain

$$u_{j+1} = u_{j-1} + 2hf(u_j, t_n + jh) \qquad j = 1, 2 \ldots N - 1. \tag{13.91}$$

Gragg [155] introduced a smoothing procedure to remove oscillations of the leading error term by defining

$$v_j = \frac{1}{4}u_{j-1} + \frac{1}{2}u_j + \frac{1}{4}u_{j+1}. \tag{13.92}$$

He showed that both approximations (13.91, 13.92) have an asymptotic expansion in powers of $h^2$ and are therefore well suited for an extrapolation method. The modified midpoint method can be summarized as follows:

$$\begin{aligned} u_0 &= Y(t_n) \\ u_1 &= u_0 + hf(u_0, t_n) \\ u_{j+1} &= u_{j-1} + 2hf(u_j, t_n + jh) \quad j = 1, 2, \ldots N - 1 \\ Y(t_n + \Delta t) &\approx \frac{1}{2}\left(u_N + u_{N-1} + hf(u_N, t_n + \Delta t)\right). \end{aligned} \tag{13.93}$$

The number of sub-steps $N$ is increased according to a sequence like

$$N = 2, 4, 6, 8, 12, 16, 24, 32, 48, 64 \ldots \quad N_j = 2N_{j-2} \quad \text{Bulirsch-Stoer sequence} \tag{13.94}$$

or

$$N = 2, 4, 6, 8, 10, 12 \ldots \quad N_j = 2j \quad \text{Deuflhard sequence.}$$

After each successive N is tried, a polynomial extrapolation is attempted. This extrapolation returns both the extrapolated values and an error estimate. If the error is still too large then N has to be increased further. A more detailed discussion can be found in [156, 157].

## 13.10 Linear Multistep Methods

All methods discussed so far evaluated one or more values of the gradient $f(Y(t), t)$ only within the interval $t_n \cdots t_n + \Delta t$. If the state vector changes sufficiently smooth then multistep methods can be applied. Linear multistep methods use a combination of function values $Y_n$ and gradients $f_n$ from several steps

$$Y_{n+1} = \sum_{j=1}^{k} \left(\alpha_j Y_{n-j+1} + \beta_j f_{n-j+1} \Delta t\right) + \beta_0 f_{n+1} \Delta t \tag{13.95}$$

where the coefficients $\alpha, \beta$ are determined such, that a polynomial of certain order $r$ is integrated exactly. The method is explicit if $\beta_0 = 0$ and implicit otherwise. Multistep methods have a small local error and need fewer function evaluations. On the other hand, they have to be combined with other methods (like Runge–Kutta) to start and end properly and it can be rather complicated to change the step size during the calculation. Three families of linear multistep methods are commonly used: explicit Adams-Bashforth methods, implicit Adams-Moulton methods and backward differentiation formulas (also known as Gear formulas [158]).

### *13.10.1 Adams-Bashforth Methods*

The explicit Adams-Bashforth method of order $r$ uses the gradients from the last $r - 1$ steps (Fig. 13.9) to obtain the polynomial

$$p(t_n) = f(Y_n, t_n), \ldots p(t_{n-r+1}) = f(Y_{n-r+1}, t_{n-r+1}) \tag{13.96}$$

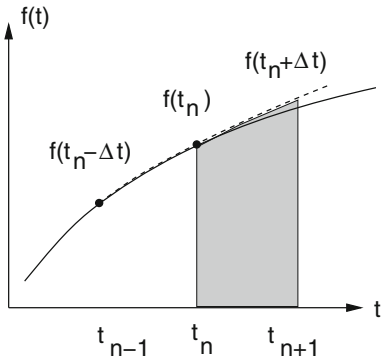


**Fig. 13.9** Adams-Bashforth method

and to calculate the approximation

$$Y_{n+1} - Y_n \approx \int_{t_n}^{t_{n+1}} p(t)dt$$

which is generally a linear combination of $f_n \cdots f_{n-r+1}$. For example, the Adams-Bashforth formulas of order 2, 3, 4 are:

$$\begin{aligned}
Y_{n+1} - Y_n &= \frac{\Delta t}{2}(3f_n - f_{n-1}) + O(\Delta t^3) \\
Y_{n+1} - Y_n &= \frac{\Delta t}{12}(23f_n - 16f_{n-1} + 5f_{m-2}) + O(\Delta t^4) \\
Y_{n+1} - Y_n &= \frac{\Delta t}{24}(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}) + O(\Delta t^5).
\end{aligned} \tag{13.97}$$

### *13.10.2 Adams-Moulton Methods*

The implicit Adams-Moulton method also uses the yet not known value $Y_{n+1}$ (Fig. 13.10) to obtain the polynomial

$$p(t_{n+1}) = f_{n+1}, \ldots p(t_{n-r+2}) = f_{n-r+2}. \tag{13.98}$$

The corresponding Adams-Moulton formulas of order 2 to 4 are:

$$\begin{aligned}
Y_{n+1} - Y_n &= \frac{\Delta t}{2}(f_{n+1} + f_n) + O(\Delta t^3) \\
Y_{n+1} - Y_n &= \frac{\Delta t}{12}(5f_{n+1} + 8f_n - f_{n-1}) + O(\Delta t^4) \qquad (13.99) \\
Y_{n+1} - Y_n &= \frac{\Delta t}{24}(9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2}) + O(\Delta t^5). \qquad (13.100)
\end{aligned}$$

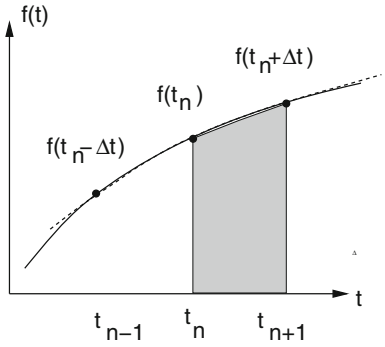


**Fig. 13.10** Adams-Moulton method

### *13.10.3 Backward Differentiation (Gear) Methods*

Gear methods [158] are implicit and usually combined with a modified Newton method. They make use of previous function values $Y_n, Y_{n-1} \ldots$ and the gradient $f_{n+1}$ at time $t + \Delta t$. Only methods of order $r \leq 6$ are stable and useful. The general formula (13.95) is

$$Y_{n+1} = \sum_{j=1}^{r} \alpha_j Y_{n-j+1} + \beta_0 f_{n+1} \Delta t. \tag{13.101}$$

For $r = 1$ this becomes

$$Y_{n+1} = \alpha_1 Y_n + \beta_0 f_1 \Delta t \tag{13.102}$$

and all linear polynomials

$$p = p_0 + p_1(t - t_n), \quad \frac{\mathrm{d}p}{\mathrm{d}t} = p_1 \tag{13.103}$$

are integrated exactly if

$$p_0 + p_1 \Delta t = \alpha_1 p_0 + \beta_0 p_1 \tag{13.104}$$

which is the case for

$$\alpha_1 = 1, \quad \beta_0 = \Delta t. \tag{13.105}$$

Hence the first order Gear method is

$$Y_{n+1} = Y_n + f_{n+1} \Delta t + O(\Delta t^2) \tag{13.106}$$

which coincides with the implicit Euler method. The higher order stable Gear methods are given by

$$r = 2: \quad Y_{n+1} = \frac{4}{3}Y_n - \frac{1}{3}Y_{n-1} + \frac{2}{3}f_{n+1}\Delta t + O(\Delta t^3) \tag{13.107}$$

$$r = 3: \quad Y_{n+1} = \frac{18}{11}Y_n - \frac{9}{11}Y_{n-1} + \frac{2}{11}Y_{n-2} + \frac{6}{11}f_{n+1}\Delta t + O(\Delta t^4) \tag{13.108}$$

$$r = 4: \quad Y_{n+1} = \frac{48}{25}Y_n - \frac{36}{25}Y_{n-1} + \frac{16}{25}Y_{n-2} - \frac{3}{25}Y_{n-3} + \frac{12}{25}f_{n+1}\Delta t + O(\Delta t^5) \tag{13.109}$$

$$r = 5: \quad Y_{n+1} = \frac{300}{137}Y_n - \frac{300}{137}Y_{n-1} + \frac{200}{137}Y_{n-2} - \frac{75}{137}Y_{n-3} + \frac{12}{137}Y_{n-4} + \frac{60}{137}f_{n+1}\Delta t + O(\Delta t^6) \tag{13.110}$$

$$r = 6: \quad Y_{n+1} = \frac{120}{49}Y_n - \frac{150}{49}Y_{n-1} + \frac{400}{147}Y_{n-2} - \frac{75}{49}Y_{n-3} + \frac{24}{49}Y_{n-4} - \frac{10}{147}Y_{n-5} + \frac{20}{49}f_{n+1}\Delta t + O(\Delta t^7). \tag{13.111}$$

This class of algorithms is useful also for stiff problems (differential equations with strongly varying eigenvalues).

### *13.10.4 Predictor-Corrector Methods*

The Adams-Bashforth–Moulton method combines the explicit method as a predictor step to calculate an estimate $y^p_{n+1}$ with a corrector step using the implicit method of same order. The general class of linear multistep predictor corrector methods [159] uses a predictor step

$$Y^{(0)}_{n+1} = \sum_{j=1}^{k} \left( \alpha_j^{(p)} Y_{n-j+1} + \beta_j^{(p)} f_{n-j+1}\Delta t \right) \tag{13.112}$$

which is corrected using the formula

$$Y^{(1)}_{n+1} = \sum_{j=1}^{k} \left( \alpha_j^{(c)} Y_{n-j+1} + \beta_j^{(c)} f_{n-j+1}\Delta t \right) + \beta_0 f(Y^{(0)}_{n+1}, t_{n+1})\Delta t \tag{13.113}$$

and further iterations

$$Y_{n+1}^{(m+1)} = Y_{n+1}^{(m)} - \beta_0 \left[ f(Y_{n+1}^{(m-1)}, t_{n+1}) - f(Y_{n+1}^{(m)}, t_{n+1}) \right] \Delta t \quad m = 1 \ldots M-1 \tag{13.114}$$

$$Y_{n+1} = Y_{n+1}^{(M)}, \quad \dot{Y}_{n+1} = f(Y_{n+1}^{(M-1)}, t_{n+1}). \tag{13.115}$$

The coefficients $\alpha, \beta$ have to be determined to optimize accuracy and stability.

## 13.11 Verlet Methods

For classical molecular dynamics simulations it is necessary to calculate very long trajectories. Here a family of symplectic methods often is used which conserve the phase space volume [160–165]. The equations of motion of a classical interacting N-body system are

$$m_i \ddot{\mathbf{x}}_i = F_i \tag{13.116}$$

where the force acting on atom i can be calculated once a specific force field is chosen. Let us write these equations as a system of first order differential equations

$$\begin{pmatrix} \dot{\mathbf{x}}_i \\ \dot{\mathbf{v}}_i \end{pmatrix} = \begin{pmatrix} \mathbf{v}_i \\ \mathbf{a}_i \end{pmatrix} \tag{13.117}$$

where $\mathbf{x}(t)$ and $\mathbf{v}(t)$ are functions of time and the forces $m\mathbf{a}(\mathbf{x}(t))$ are functions of the time dependent coordinates.

### *13.11.1 Liouville Equation*

We rewrite (13.117) as

$$\begin{pmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{v}} \end{pmatrix} = \mathcal{L} \begin{pmatrix} \mathbf{x} \\ \mathbf{v} \end{pmatrix} \tag{13.118}$$

where the Liouville operator $\mathcal{L}$ acts on the vector containing all coordinates and velocities:

$$\mathcal{L} \begin{pmatrix} \mathbf{x} \\ \mathbf{v} \end{pmatrix} = \left( \mathbf{v} \frac{\partial}{\partial \mathbf{x}} + \mathbf{a} \frac{\partial}{\partial \mathbf{v}} \right) \begin{pmatrix} \mathbf{x} \\ \mathbf{v} \end{pmatrix}. \tag{13.119}$$

The Liouville equation (13.118) can be formally solved by

$$\begin{pmatrix}\mathbf{x}(t)\\ \mathbf{v}(t)\end{pmatrix} = e^{\mathcal{L}t}\begin{pmatrix}\mathbf{x}(0)\\ \mathbf{v}(0)\end{pmatrix}. \tag{13.120}$$

For a better understanding let us evaluate the first members of the Taylor series of the exponential:

$$\mathcal{L}\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix} = \left(\mathbf{v}\frac{\partial}{\partial \mathbf{x}} + \mathbf{a}\frac{\partial}{\partial \mathbf{v}}\right)\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix} = \begin{pmatrix}\mathbf{v}\\ \mathbf{a}\end{pmatrix} \tag{13.121}$$

$$\mathcal{L}^2\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix} = \left(\mathbf{v}\frac{\partial}{\partial \mathbf{x}} + \mathbf{a}\frac{\partial}{\partial \mathbf{v}}\right)\begin{pmatrix}\mathbf{v}\\ \mathbf{a}(\mathbf{x})\end{pmatrix} = \begin{pmatrix}\mathbf{a}\\ \mathbf{v}\frac{\partial}{\partial \mathbf{x}}\mathbf{a}\end{pmatrix} \tag{13.122}$$

$$\mathcal{L}^3\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix} = \left(\mathbf{v}\frac{\partial}{\partial \mathbf{x}} + \mathbf{a}\frac{\partial}{\partial \mathbf{v}}\right)\begin{pmatrix}\mathbf{a}\\ \mathbf{v}\frac{\partial}{\partial \mathbf{x}}\mathbf{a}\end{pmatrix} = \begin{pmatrix}\mathbf{v}\frac{\partial}{\partial \mathbf{x}}\mathbf{a}\\ \mathbf{a}\frac{\partial}{\partial \mathbf{x}}\mathbf{a} + \mathbf{v}\mathbf{v}\frac{\partial}{\partial \mathbf{x}}\frac{\partial}{\partial \mathbf{x}}\mathbf{a}\end{pmatrix}. \tag{13.123}$$

But since

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{a}(\mathbf{x}(t)) = \mathbf{v}\frac{\partial}{\partial \mathbf{x}}\mathbf{a} \tag{13.124}$$

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{a}(\mathbf{x}(t)) = \frac{\mathrm{d}}{\mathrm{d}t}\left(\mathbf{v}\frac{\partial}{\partial \mathbf{x}}\mathbf{a}\right) = \mathbf{a}\frac{\partial}{\partial \mathbf{x}}\mathbf{a} + \mathbf{v}\mathbf{v}\frac{\partial}{\partial \mathbf{x}}\frac{\partial}{\partial \mathbf{x}}\mathbf{a} \tag{13.125}$$

we recover

$$\left(1 + t\mathcal{L} + \frac{1}{2}t^2\mathcal{L}^2 + \frac{1}{6}t^3\mathcal{L}^3 + \cdots\right)\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix} = \begin{pmatrix}\mathbf{x} + \mathbf{v}t + \frac{1}{2}t^2\mathbf{a} + \frac{1}{6}t^3\dot{\mathbf{a}} + \cdots\\ \mathbf{v} + \mathbf{a}t + \frac{1}{2}t^2\dot{\mathbf{a}} + \frac{1}{6}t^3\ddot{\mathbf{a}} + \cdots\end{pmatrix}. \tag{13.126}$$

### *13.11.2 Split Operator Approximation*

We introduce a small time step $\Delta t = t/N$ and write

$$e^{\mathcal{L}t} = \left(e^{\mathcal{L}\Delta t}\right)^N. \tag{13.127}$$

For the small time step $\Delta t$ the split-operator approximation can be used which approximately factorizes the exponential operator. For example, write the Liouville operator as the sum of two terms

$$\mathcal{L}_A = \mathbf{v}\frac{\partial}{\partial \mathbf{x}} \qquad \mathcal{L}_B = \mathbf{a}\frac{\partial}{\partial \mathbf{v}}$$

and make the approximation

$$e^{\mathcal{L}\Delta t} = e^{\mathcal{L}_A \Delta t} e^{\mathcal{L}_B \Delta t} + \cdots . \tag{13.128}$$

Each of the two factors simply shifts positions or velocities

$$e^{\mathcal{L}_A \Delta t}\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix} = \begin{pmatrix}\mathbf{x}+\mathbf{v}\Delta t\\ \mathbf{v}\end{pmatrix} \qquad e^{\mathcal{L}_B \Delta t}\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix} = \begin{pmatrix}\mathbf{x}\\ \mathbf{v}+\mathbf{a}\Delta t\end{pmatrix} \tag{13.129}$$

since these two steps correspond to either motion with constant velocities or constant coordinates and forces.

### 13.11.3 Position Verlet Method

Often the following approximation is used which is symmetrical in time

$$e^{\mathcal{L}\Delta t} = e^{\mathcal{L}_A \Delta t/2} e^{\mathcal{L}_B \Delta t} e^{\mathcal{L}_A \Delta t/2} + \cdots . \tag{13.130}$$

The corresponding algorithm is the so called position Verlet method (Fig. 13.11):

$$\mathbf{x}_{n+1/2} = \mathbf{x}_n + \mathbf{v}_n \frac{\Delta t}{2} \tag{13.131}$$

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathbf{a}_{n+1/2}\Delta t = \mathbf{v}(t_n + \Delta t) + O(\Delta t^3) \tag{13.132}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_{n+1/2} + \mathbf{v}_{n+1}\frac{\Delta t}{2} = \mathbf{x}_n + \frac{\mathbf{v}_n + \mathbf{v}_{n+1}}{2}\Delta t = \mathbf{x}(t_n + \Delta t) + O(\Delta t^3). \tag{13.133}$$

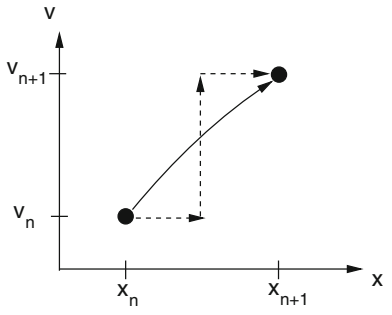


**Fig. 13.11** (Position Verlet method) The exact integration path is approximated by two half-steps with constant velocities and one step with constant coordinates

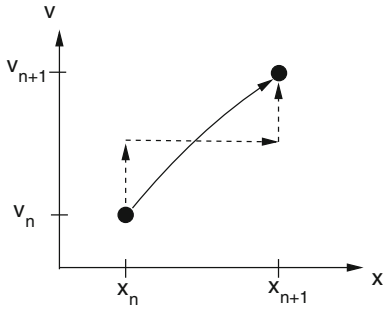


**Fig. 13.12** (Velocity Verlet method) The exact integration path is approximated by two half-steps with constant coordinates and one step with constant velocities

### *13.11.4 Velocity Verlet Method*

If we exchange operators $A$ and $B$ we have

$$e^{\mathcal{L}\Delta t} = e^{\mathcal{L}_B \Delta t/2} e^{\mathcal{L}_A \Delta t} e^{\mathcal{L}_B \Delta t/2} + \cdots \tag{13.134}$$

which produces the velocity Verlet algorithm (Fig. 13.12):

$$\mathbf{v}_{n+1/2} = \mathbf{v}_n + \mathbf{a}_n \frac{\Delta t}{2} \tag{13.135}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{v}_{n+1/2}\Delta t = \mathbf{x}_n + \mathbf{v}_n \Delta t + \mathbf{a}_n \frac{\Delta t^2}{2} = \mathbf{x}(t_n + \Delta t) + O(\Delta t^3) \tag{13.136}$$

$$\mathbf{v}_{n+1} = \mathbf{v}_{n+1/2} + \mathbf{a}_{n+1}\frac{\Delta t}{2} = \mathbf{v}_n + \frac{\mathbf{a}_n + \mathbf{a}_{n+1}}{2}\Delta t = \mathbf{v}(t_n + \Delta t) + O(\Delta t^3). \tag{13.137}$$

### *13.11.5 Stoermer-Verlet Method*

The velocity Verlet method is equivalent to Stoermer's version [166] of the Verlet method which is a two step method given by

$$\mathbf{x}_{n+1} = 2\mathbf{x}_n - \mathbf{x}_{n-1} + \mathbf{a}_n \Delta t^2 \tag{13.138}$$

$$\mathbf{v}_n = \frac{\mathbf{x}_{n+1} - \mathbf{x}_{n-1}}{2\Delta t}. \tag{13.139}$$

To show the equivalence we add two consecutive position vectors

$$\mathbf{x}_{n+2} + \mathbf{x}_{n+1} = 2\mathbf{x}_{n+1} + 2\mathbf{x}_n - \mathbf{x}_n - \mathbf{x}_{n-1} + (\mathbf{a}_{n+1} + \mathbf{a}_n)\Delta t^2 \tag{13.140}$$

which simplifies to

$$\mathbf{x}_{n+2} - \mathbf{x}_n - (\mathbf{x}_{n+1} - \mathbf{x}_n) = (\mathbf{a}_{n+1} + \mathbf{a}_n)\Delta t^2. \tag{13.141}$$

This can be expressed as the difference of two consecutive velocities:

$$2(\mathbf{v}_{n+1} - \mathbf{v}_n) = (\mathbf{a}_{n+1} + \mathbf{a}_n)\Delta t. \tag{13.142}$$

Now we substitute

$$\mathbf{x}_{n-1} = \mathbf{x}_{n+1} - 2\mathbf{v}_n \Delta t \tag{13.143}$$

to get

$$\mathbf{x}_{n+1} = 2\mathbf{x}_n - \mathbf{x}_{n+1} + 2\mathbf{v_n}\Delta t + \mathbf{a}_n \Delta t^2 \tag{13.144}$$

which simplifies to

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{v}_n \Delta t + \frac{\mathbf{a}_n}{2}\Delta t^2. \tag{13.145}$$

Thus the equations of the velocity Verlet algorithm have been recovered. However, since the Verlet method is a 2-step method, the choice of initial values is important. The Stoermer-Verlet method starts from two coordinate sets $x_0, x_1$. The first step is

$$\mathbf{x}_2 = 2\mathbf{x_1} - \mathbf{x_0} + a_1 \Delta t^2 \tag{13.146}$$

$$\mathbf{v}_1 = \frac{\mathbf{x}_2 - \mathbf{x}_0}{2\Delta t} = \frac{\mathbf{x}_1 - \mathbf{x}_0}{\Delta t} + \frac{\mathbf{a}_1}{2}\Delta t^2. \tag{13.147}$$

The velocity Verlet method, on the other hand, starts from one set of coordinates and velocities $\mathbf{x}_1, \mathbf{v}_1$. Here the first step is

$$\mathbf{x}_2 = \mathbf{x}_1 + \mathbf{v}_1 \Delta t + \mathbf{a}_1 \frac{\Delta t^2}{2} \tag{13.148}$$

$$\mathbf{v}_2 = \mathbf{v}_1 + \frac{\mathbf{a}_1 + \mathbf{a}_2}{2}\Delta t. \tag{13.149}$$

The two methods give the same resulting trajectory if we choose

$$\mathbf{x}_0 = \mathbf{x}_1 - \mathbf{v}_1 \Delta t + \frac{\mathbf{a}_1}{2}\Delta t^2. \tag{13.150}$$

If, on the other hand, $\mathbf{x}_0$ is known with higher precision, the local error order of Stoermer's algorithm changes as can be seen from addition of the two Taylor series

$$\mathbf{x}(t_n + \Delta t) = \mathbf{x}_n + \mathbf{v}_n \Delta t + \frac{\mathbf{a}_n}{2}\Delta t^2 + \frac{\dot{\mathbf{a}}_n}{6}\Delta t^3 + \cdots \tag{13.151}$$

$$\mathbf{x}(t_n - \Delta t) = \mathbf{x}_n - \mathbf{v}_n \Delta t + \frac{\mathbf{a}_n}{2}\Delta t^2 - \frac{\dot{\mathbf{a}}_n}{6}\Delta t^3 + \cdots \tag{13.152}$$

which gives

$$\mathbf{x}(t_n + \Delta t) = 2\mathbf{x}(t_n) - \mathbf{x}(t_n - \Delta t) + \mathbf{a}_n \Delta t^2 + O(\Delta t^4) \tag{13.153}$$

$$\frac{\mathbf{x}(t_n + \Delta t) - \mathbf{x}(t_n - \Delta t)}{2\Delta t} = \mathbf{v}_n + O(\Delta t^2). \tag{13.154}$$

### 13.11.6 Error Accumulation for the Stoermer-Verlet Method

Equation (13.153) gives only the local error of one single step. Assume the start values $\mathbf{x}_0$ and $\mathbf{x}_1$ are exact. The next value $\mathbf{x}_2$ has an error with the leading term $\Delta x_2 = \alpha \Delta t^4$. If the trajectory is sufficiently smooth and the time step not too large the coefficient $\alpha$ will vary only slowly and the error of the next few iterations is given by

$$\begin{aligned}
\Delta x_3 &= 2\Delta x_2 - \Delta x_1 = 2\alpha \Delta t^4 \\
\Delta x_4 &= 2\Delta x_3 - \Delta x_2 = 3\alpha \Delta t^4 \\
&\vdots \\
\Delta x_{n+1} &= n\alpha \Delta t^4.
\end{aligned} \tag{13.155}$$

This shows that the effective error order of the Stoermer-Verlet method is only $O(\Delta t^3)$ similar to the velocity Verlet method.

### 13.11.7 Beeman's Method

Beeman and Schofield [167, 168] introduced a method which is very similar to the Stoermer-Verlet method but calculates the velocities with higher precision. This is important if, for instance, the kinetic energy has to be calculated. Starting from the Taylor series

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{v}_n \Delta t + \mathbf{a}_n \frac{\Delta t^2}{2} + \dot{\mathbf{a}}_n \frac{\Delta t^3}{6} + \ddot{\mathbf{a}}_n \frac{\Delta t^4}{24} + \cdots \tag{13.156}$$

the derivative of the acceleration is approximated by a backward difference

$$\begin{aligned}\mathbf{x}_{n+1} &= \mathbf{x}_n + \mathbf{v}_n \Delta t + \mathbf{a}_n \frac{\Delta t^2}{2} + \frac{\mathbf{a}_n - \mathbf{a}_{n-1}}{\Delta t} \frac{\Delta t^3}{6} + O(\Delta t^4) \\ &= \mathbf{x}_n + \mathbf{v}_n \Delta t + \frac{4\mathbf{a}_n - \mathbf{a}_{n-1}}{6} \Delta t^2 + O(\Delta t^4). \end{aligned} \tag{13.157}$$

This equation can be used as an explicit step to update the coordinates or as a predictor step in combination with the implicit corrector step

$$\begin{aligned}\mathbf{x}_{n+1} &= \mathbf{x}_n + \mathbf{v}_n \Delta t + \mathbf{a}_n \frac{\Delta t^2}{2} + \frac{\mathbf{a}_{n+1} - \mathbf{a}_n}{\Delta t} \frac{\Delta t^3}{6} + O(\Delta t^4) \\ &= \mathbf{x}_n + \mathbf{v}_n \Delta t + \frac{\mathbf{a}_{n+1} + 2\mathbf{a}_n}{6} \Delta t^2 + O(\Delta t^4) \end{aligned} \tag{13.158}$$

which can be applied repeatedly (usually two iterations are sufficient). Similarly, the Taylor series of the velocity is approximated by

$$\begin{aligned}\mathbf{v}_{n+1} &= \mathbf{v}_n + \mathbf{a}_n \Delta t + \dot{\mathbf{a}}_n \frac{\Delta t^2}{2} + \ddot{\mathbf{a}}_n \frac{\Delta t^3}{6} + \cdots \\ &= \mathbf{v}_n + \mathbf{a}_n \Delta t + \left( \frac{\mathbf{a}_{n+1} - \mathbf{a}_n}{\Delta t} + O(\Delta t) \right) \frac{\Delta t^2}{2} + \cdots \\ &= \mathbf{v}_n + \frac{\mathbf{a}_{n+1} + \mathbf{a}_n}{2} \Delta t + O(\Delta t^3). \end{aligned} \tag{13.159}$$

Inserting the velocity from (13.158) we obtain the corrector step for the velocity

$$\begin{aligned}\mathbf{v}_{n+1} &= \frac{\mathbf{x}_{n+1} - \mathbf{x}_n}{\Delta t} - \frac{\mathbf{a}_{n+1} + 2\mathbf{a}_n}{6} \Delta t + \frac{\mathbf{a}_{n+1} + \mathbf{a}_n}{2} \Delta t + O(\Delta t^3) \\ &= \frac{\mathbf{x}_{n+1} - \mathbf{x}_n}{\Delta t} + \frac{2\mathbf{a}_{n+1} + \mathbf{a}_n}{6} \Delta t + O(\Delta t^3). \end{aligned} \tag{13.160}$$

In combination with (13.157) this can be replaced by

$$\begin{aligned}\mathbf{v}_{n+1} &= \mathbf{v}_n + \frac{4\mathbf{a}_n - \mathbf{a}_{n-1}}{6} \Delta t + \frac{2\mathbf{a}_{n+1} + \mathbf{a}_n}{6} \Delta t + O(\Delta t^3) \\ &= \mathbf{v}_n + \frac{2\mathbf{a}_{n+1} + 5\mathbf{a}_n - \mathbf{a}_{n-1}}{6} \Delta t + O(\Delta t^3). \end{aligned} \tag{13.161}$$

Together, (13.157) and (13.161) provide an explicit method which is usually understood as Beeman's method. Inserting the velocity (13.160) from the previous step

$$\mathbf{v}_n = \frac{\mathbf{x}_n - \mathbf{x}_{n-1}}{\Delta t} + \frac{2\mathbf{a}_n + \mathbf{a}_{n-1}}{6} \Delta t + O(\Delta t^3) \tag{13.162}$$

into (13.157) gives

$$\mathbf{x}_{n+1} = 2\mathbf{x}_n - \mathbf{x}_{n-1} + \mathbf{a}_n \Delta t^2 + O(\Delta t^4) \tag{13.163}$$

which coincides with the Stoermer-Verlet method (13.138). We conclude that Beeman's method should produce the same trajectory as the Stoermer-Verlet method if numerical errors can be neglected and comparable initial values are used. In fact, the Stoermer-Verlet method may suffer from numerical extinction and Beeman's method provides a numerically more favorable alternative.

### *13.11.8 The Leapfrog Method*

Closely related to the Verlet methods is the so called leapfrog method [165]. It uses the simple decomposition

$$e^{\mathcal{L}\Delta t} \approx e^{\mathcal{L}_A \Delta t} e^{\mathcal{L}_B \Delta t} \tag{13.164}$$

but introduces two different time grids for coordinates and velocities which are shifted by $\Delta t/2$ (Fig. 13.13).

The leapfrog algorithm is given by

$$\mathbf{v}_{n+1/2} = \mathbf{v}_{n-1/2} + \mathbf{a}_n \Delta t \tag{13.165}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{v}_{n+1/2} \Delta t. \tag{13.166}$$

Due to the shifted arguments the order of the method is increased as can be seen from the Taylor series:

$$\mathbf{x}(t_n) + \left( \mathbf{v}(t_n) + \frac{\Delta t}{2} \mathbf{a}(t_n) + \cdots \right) \Delta t = \mathbf{x}(t_n + \Delta t) + O(\Delta t^3) \tag{13.167}$$

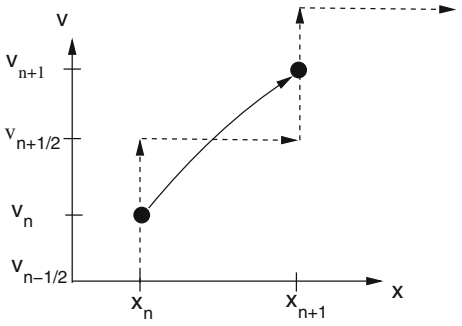


**Fig. 13.13** (Leapfrog method) The exact integration path is approximated by one step with constant coordinates and one step with constant velocities. Two different grids are used for coordinates and velocities which are shifted by $\Delta t/2$

$$\mathbf{v}\left(t_n + \frac{\Delta t}{2}\right) - \mathbf{v}\left(t_n - \frac{\Delta t}{2}\right) = \mathbf{a}(t_n)\Delta t + O(\Delta t^3). \tag{13.168}$$

One disadvantage of the leapfrog method is that some additional effort is necessary if the velocities are needed. The simple expression

$$\mathbf{v}(t_n) = \frac{1}{2}\left(\mathbf{v}\left(t_n - \frac{\Delta t}{2}\right) + \mathbf{v}\left(t_n + \frac{\Delta t}{2}\right)\right) + O(\Delta t^2) \tag{13.169}$$

is of lower error order than (13.168).

## Problems

### Problem 13.1 Circular Orbits

In this computer experiment we consider a mass point moving in a central field. The equation of motion can be written as the following system of first order equations:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{1}{(x^2+y^2)^{3/2}} & 0 & 0 & 0 \\ 0 & -\frac{1}{(x^2+y^2)^{3/2}} & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ v_x \\ v_y \end{pmatrix}. \tag{13.170}$$

For initial values

$$\begin{pmatrix} x \\ y \\ v_x \\ v_y \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{13.171}$$

the exact solution is given by

$$x = \cos t \quad y = \sin t. \tag{13.172}$$

The following methods are used to calculate the position $x(t)$, $y(t)$ and the energy

$$E_{tot} = E_{kin} + E_{pot} = \frac{1}{2}(v_x^2 + v_y^2) - \frac{1}{\sqrt{x^2+y^2}}. \tag{13.173}$$

- The explicit Euler method (13.3)

$$\begin{aligned} x(t_{n+1}) &= x(t_n) + v_x(t_n)\Delta t \\ y(t_{n+1}) &= y(t_n) + v_y(t_n)\Delta t \\ v_x(t_{n+1}) &= v_x(t_n) - \tfrac{x(t_n)}{R(t_n)^3}\Delta t \\ v_y(t_{n+1}) &= v_y(t_n) - \tfrac{y(t_n)}{R(t_n)^3}\Delta t. \end{aligned} \tag{13.174}$$

- The 2nd order Runge–Kutta method (13.7.1)

which consists of the predictor step

$$x(t_n + \Delta t/2) = x(t_n) + \frac{\Delta t}{2} v_x(t_n) \tag{13.175}$$

$$y(t_n + \Delta t/2) = y(t_n) + \frac{\Delta t}{2} v_y(t_n) \tag{13.176}$$

$$v_x(t_n + \Delta t/2) = v_x(t_n) - \frac{\Delta t}{2} \frac{x(t_n)}{R(t_n)^3} \tag{13.177}$$

$$v_y(t_n + \Delta t/2) = v_y(t_n) - \frac{\Delta t}{2} \frac{y(t_n)}{R(t_n)^3} \tag{13.178}$$

and the corrector step

$$x(t_{n+1}) = x(t_n) + \Delta t\ v_x(t_n + \Delta t/2) \tag{13.179}$$

$$y(t_{n+1}) = y(t_n) + \Delta t\ v_y(t_n + \Delta t/2) \tag{13.180}$$

$$v_x(t_{n+1}) = v_x(t_n) - \Delta t \frac{x(t_n + \Delta t/2)}{R^3(t_n + \Delta t/2)} \tag{13.181}$$

$$v_y(t_{n+1}) = v_y(t_n) - \Delta t \frac{y(t_n + \Delta t/2)}{R^3(t_n + \Delta t/2)}. \tag{13.182}$$

- The fourth order Runge–Kutta method (13.7.3)
- The Verlet method (13.11.5)

$$x(t_{n+1}) = x(t_n) + (x(t_n) - x(t_{n-1})) - \Delta t \frac{x(t_n)}{R^3(t_n)} \tag{13.183}$$

$$y(t_{n+1}) = y(t_n) + (y(t_n) - y(t_{n-1})) - \Delta t \frac{y(t_n)}{R^3(t_n)} \tag{13.184}$$

$$v_x(t_n) = \frac{x(t_{n+1}) - x(t_{n-1})}{2\Delta t} = \frac{x(t_n) - x(t_{n-1})}{\Delta t} - \frac{\Delta t}{2} \frac{x(t_n)}{R^3(t_n)} \tag{13.185}$$

$$v_y(t_n) = \frac{y(t_{n+1}) - y(t_{n-1})}{2\Delta t} = \frac{y(t_n) - y(t_{n-1})}{\Delta t} - \frac{\Delta t}{2} \frac{y(t_n)}{R^3(t_n)}. \tag{13.186}$$

To start the Verlet method we need additional coordinates at time $-\Delta t$ which can be chosen from the exact solution or from the approximation

$$x(t_{-1}) = x(t_0) - \Delta t\, v_x(t_0) - \frac{\Delta t^2}{2}\frac{x(t_0)}{R^3(t_0)} \tag{13.187}$$

$$y(t_{-1}) = y(t_0) - \Delta t\, v_y(t_0) - \frac{\Delta t^2}{2}\frac{y(t_0)}{R^3(t_0)}. \tag{13.188}$$

- The leapfrog method (13.11.8)

$$x(t_{n+1}) = x(t_n) + v_x(t_{n+1/2})\Delta t \tag{13.189}$$

$$y(t_{n+1}) = y(t_n) + v_y(t_{n+1/2})\Delta t \tag{13.190}$$

$$v_x(t_{n+1/2}) = v_x(t_{n-1/2}) - \frac{x(t_n)}{R(t_n)^3}\Delta t \tag{13.191}$$

$$v_y(t_{n+1/2}) = v_y(t_{n-1/2}) - \frac{y(t_n)}{R(t_n)^3}\Delta t \tag{13.192}$$

where the velocity at time $t_n$ is calculated from

$$v_x(t_n) = v_x(t_{n+1/2}) - \frac{\Delta t}{2}\frac{x(t_{n+1})}{R^3(t_{n+1})} \tag{13.193}$$

$$v_y(t_n) = v_y(t_{n+1/2}) - \frac{\Delta t}{2}\frac{y(t_{n+1})}{R^3(t_{n+1})}. \tag{13.194}$$

To start the leapfrog method we need the velocity at time $t_{-1/2}$ which can be taken from the exact solution or from

$$v_x(t_{-1/2}) = v_x(t_0) - \frac{\Delta t}{2}\frac{x(t_0)}{R^3(t_0)} \tag{13.195}$$

$$v_y(t_{-1/2}) = v_y(t_0) - \frac{\Delta t}{2}\frac{y(t_0)}{R^3(t_0)}. \tag{13.196}$$

Compare the conservation of energy for the different methods as a function of the time step $\Delta t$. Study the influence of the initial values for leapfrog and Verlet methods.

**Problem 13.2 N-body System**

In this computer experiment we simulate the motion of three mass points under the influence of gravity. Initial coordinates and velocities as well as the masses can be varied. The equations of motion are solved with the 4th order Runge–Kutta method with quality control for different step sizes. The local integration error is estimated

using the step doubling method. Try to simulate a planet with a moon moving round a sun!

**Problem 13.3 Adams-Bashforth Method**

In this computer experiment we simulate a circular orbit with the Adams-Bashforth method of order 2 . . . 7. The absolute error at time T

$$\Delta(T) = |x(T) - \cos(T)| + |y(t) - \sin(T)| + |v_x(T) + \sin(T)| + |v_y(T) - \cos(T)| \tag{13.197}$$

is shown as a function of the time step $\Delta t$ in a log-log plot. From the slope

$$s = \frac{d(\log_{10}(\Delta))}{d(\log_{10}(\Delta t))} \tag{13.198}$$

the leading error order $s$ can be determined. For very small step sizes rounding errors become dominating which leads to an increase $\Delta \sim (\Delta t)^{-1}$.

Determine maximum precision and optimal step size for different orders of the method. Compare with the explicit Euler method.