Computanional Physics - Problem Sheet 2

Christian Gorjaew (354259)
Julius Meyer-Ohlendorf (355733)

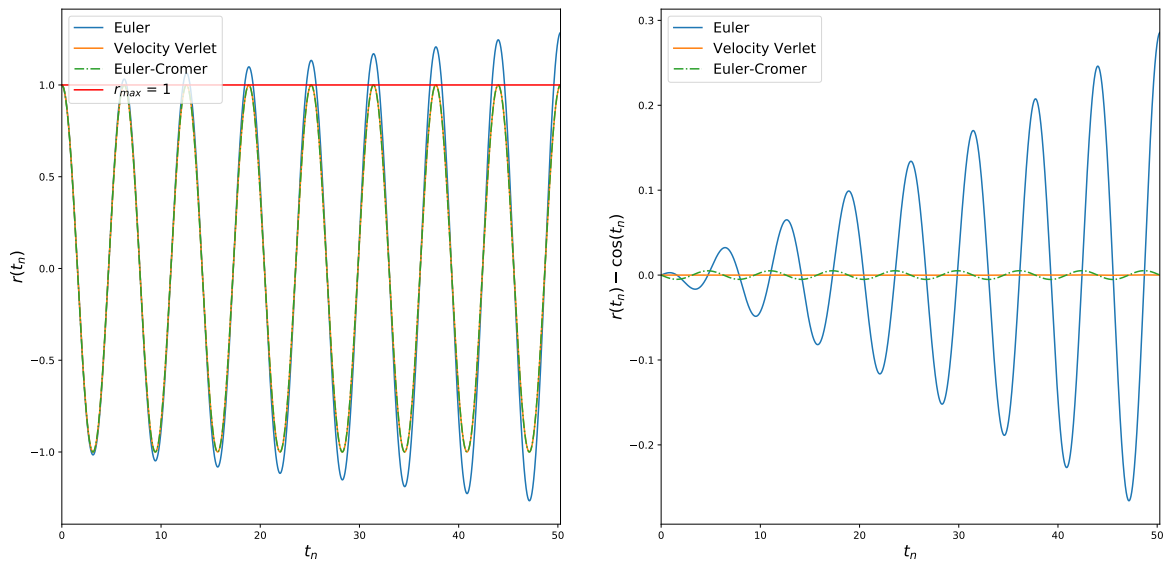
1 Euler-Cromer Method

The code for this exercise can be found in the file "cp_ex2_p1-2.py" ll. 1-175. The methods are implemented as classes ("euler" and "verlet") that inherit a parent skeleton class ("class newton_1D_sket"). The daughter classes define method specific parameters in their initializers and redefine the "step" function from their parent class ("newton_1D_sket._step"). The "step" function executes one step of the applied method. Call of the "solve" function iterates over the specified integration range in steps of the chosen step size for the desired initial conditions and the function that describes $\ddot{r}(t)$ which are all specified when an instance of the class is initialized. The class "euler" implements both the Euler as well as the Euler-Cromer method at once. The method is specified by a boolean parameter in the class' initializer.

The results for a 1D harmonic oscillator

$$\ddot{r}(t) = -\omega^2 r$$

solved by the desired methods can be seen in Fig. 1.



(a) Numerical solutions for $r(t_n)$.

(b) Residuals.

Figure 1: Solution of the 1D harmonic oscillator calculated with different methods (a). Corresponding residuals are shown in (b). Initial conditions $x(t_0 = 0) = x_{max} = 1$ and $v(t_0) = 0$ and step size $\tau = 0.01$ have been used for the calculations.

The parameters ω and $r(t_0) \equiv r_{max}$ were set to 1. The step site was $\tau = 0.01$ in a range of $t_0 = 0$ to $t_f = 32\pi$. One sees in Fig. 1(a) that the amplitude increases for the Euler method with t_n while the solution is bounded (at least over a much longer time period) for the velocity Verlet and the Euler-Cromer methods. In Fig. 1(b) the exact solution $r(t) = \cos(t)$ has been subtracted from the numerical results. One notices that the Euler-Cromer solution shows some periodic deviation from the analytical result. Not visible is the velocity Verlet

solution's oscillatory behavior which occurs on a much smaller length scale with respect to the Euler-Cromer method.

In Fig. 2 the corresponding energy (disregarding mass m and force constant k)

$$E_n = \frac{1}{2} r(t_n)^2 + \frac{1}{2} v(t_n)^2$$

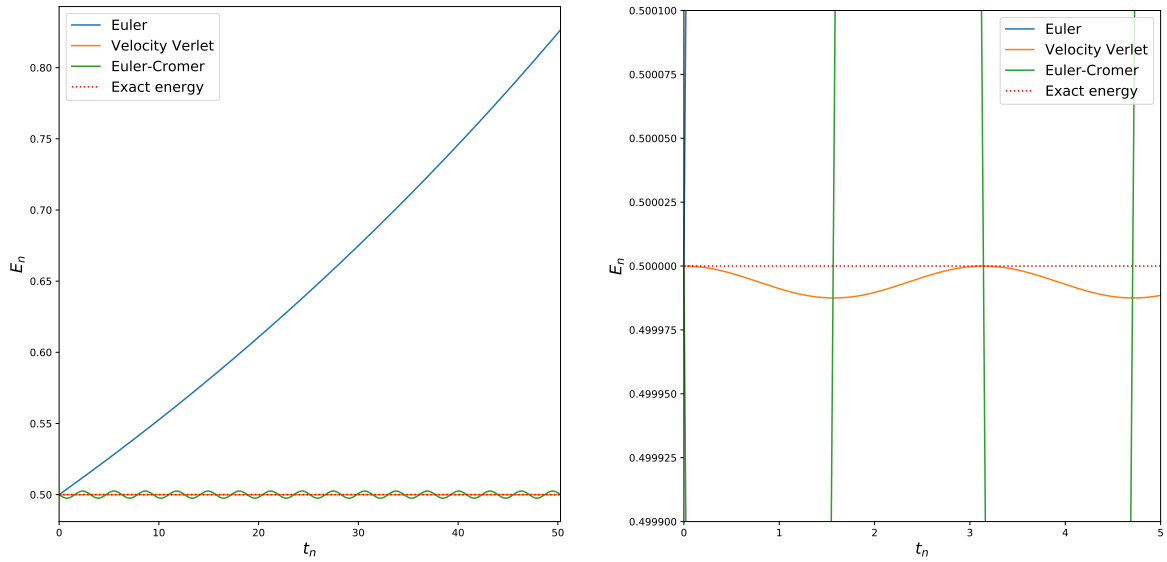
can bee seen.

The energy calculated with the results from the Euler method increases at high rate explaining the increase in amplitude of the solution for $r(t_n)$. The energy corresponding to the Verlet solution shows a subtle oscillatory behavior leaving to deviation of less then 0.001 % from the initial energy but stays bounded as expected for a symplectic algorithm. The Euler-Cromer leads to a bounded, oscillating energy as well but with deviation on a larger scale with respect to the velocity Verlet method. This suggests that the method is symplectic explaining why it performs better than the Euler method.

Furthermore, the deviation of energies between two adjacent steps of the Euler-Cromer method is

$$E(t_{n+1}) - E(t_n) = \tau^2 \Big(E(t_n) - v(t_n)^2 \Big) + \mathcal{O}(\tau^3) \approx \frac{\tau^2}{2} \Big(r(t_n)^2 - v(t_n)^2 \Big)$$

where it can be seen that the deviations are not strictly positive anymore as it was the case for the Euler method and led to an strictly monotonically increasing energy. Thus, the deviations are able to steer back towards the initial value in the Euler-Cromer method.



(a) Energies over the full time intervall.

(b) Zoom into 2(a) in order to resolve oscillations of the Verlet algorithm.

Figure 2: Energy E_n as a function of time t_n for the investigated methods.

2 Ideal Pendulum

The code for this exercise can be found in the file "cp_ex2_p1-2.py" ll. 176-214. The differential equation of the ideal pendulum

$$\ddot{r} = -\frac{l}{g} \cdot \sin(x)$$

was solved using the implementation of the velocity Verlet method in problem 1. The length constants have been chosen such that they yield 1. The initial parameters were kept at the same values as in problem 1 with exceptions for the step size which was set to $\tau = 0.001$ and the initial position x_{max} which was varied.

The period was calculated using the fast Fourier transformation of the calculated positions. Here, simply the bin with the highest peak was chosen as the signal's frequency which was then used to calculate the period. The result can for varying x_{max} can be seen in Fig. 3 where the period for the ideal pendulum was determined for 10 different initial values.

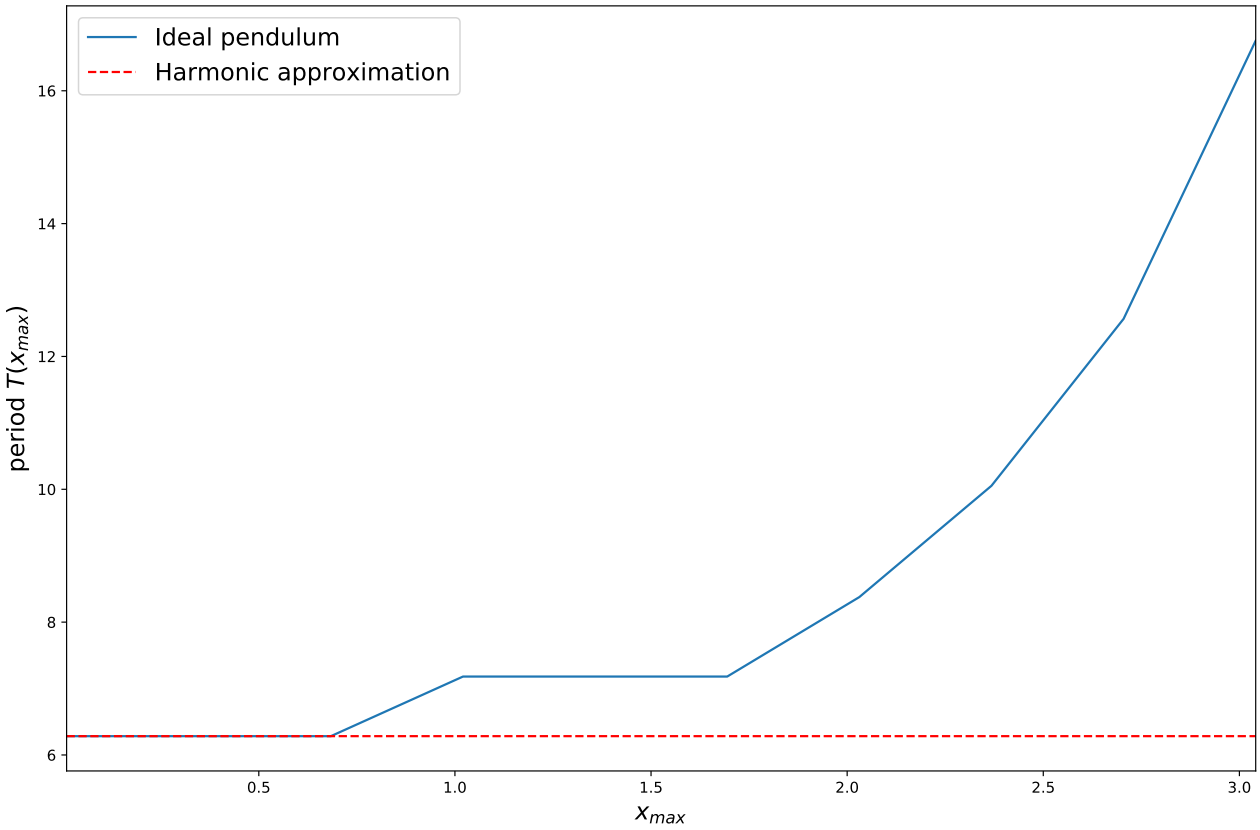


Figure 3: Period T as function of maximum deflection x_max for an ideal pendulum and harmonic oscillator.

One recognizes a quantization of the period for the ideal pendulum which originates from the limited resolution of the fast Fourier transformation and the fact that just the highest bin was used. For the same reason, there were some slight deviations from the actual period. One can still see that the period of the ideal pendulum depends on the initial deflection of the pendulum in a visually quadratic manner while the harmonic oscillator's period remains constant as can be easily seen by solving the problem analytically. By the comparison of the two curves, one could say, with caution though due to quantization in the ideal pendulum's curve, that a harmonic approximation is indeed valid for small deflection x_{max}.

3 Leap-frog Scheme with Friction

The leap-frog method is defined by the following scheme where $\vec{r}(t_n) \equiv \vec{r}(n)$, $\vec{v}(t_n) \equiv \vec{v}(n)$ and $\vec{a}(\vec{r}(t_n), \vec{v}(t_n), t_n) \equiv \vec{a}(n)$:

$$\vec{r}(n+1) = \vec{r}(n) + \tau \vec{v}\left(n + \frac{1}{2}\right) \tag{1}$$

$$\vec{v}\left(n + \frac{1}{2}\right) = \vec{v}\left(n - \frac{1}{2}\right) + \tau \vec{a}(n) \tag{2}$$

Plugging the given expression for the acceleration into Eq. (2) yields

$$\vec{v}\left(n + \frac{1}{2}\right) = \vec{v}\left(n - \frac{1}{2}\right) + \tau \vec{a}^0(n) - \tau \Gamma \vec{v}(n) \tag{3}$$

We then notice, using the three-point formula for the velocity as given in the lecture slides of lecture 2 on page 32, that

$$\vec{v}\left(n + \frac{1}{2}\right) + \vec{v}\left(n - \frac{1}{2}\right) = \frac{\vec{r}(n+1) - \vec{r}(n)}{\tau} + \frac{\vec{r}(n) - \vec{r}(n-1)}{\tau} \tag{4}$$

$$= 2 \cdot \underbrace{\frac{\vec{r}(n+1) - \vec{r}(n-1)}{2\tau}}_{\vec{v}(n)} \tag{5}$$

$$= 2\vec{v}(n) \tag{6}$$

Thus, (3) becomes

$$\vec{v}\left(n + \frac{1}{2}\right) = \vec{v}\left(n - \frac{1}{2}\right) + \tau \vec{a}^0(n) - \frac{\tau}{2}\Gamma\left(\vec{v}\left(n + \frac{1}{2}\right) + \vec{v}\left(n - \frac{1}{2}\right)\right) \tag{7}$$

$$\Leftrightarrow \quad \left(\mathbb{1} + \frac{\tau}{2}\Gamma\right)\vec{v}\left(n + \frac{1}{2}\right) = \left(\mathbb{1} - \frac{\tau}{2}\Gamma\right)\vec{v}\left(n - \frac{1}{2}\right) + \tau \vec{a}^0(n) \tag{8}$$

$$\Leftrightarrow \quad \vec{v}\left(n + \frac{1}{2}\right) = \underline{\underline{\left(\mathbb{1} + \frac{\tau}{2}\Gamma\right)^{-1}\left[\left(\mathbb{1} - \frac{\tau}{2}\Gamma\right)\vec{v}\left(n - \frac{1}{2}\right) + \tau \vec{a}^0(n)\right]}} \tag{9}$$

Since the formula for the position vector (Eq. (1)) does not depend explicitly on the acceleration, it stays unchanged.