HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Assignment Sheet Nr. 3

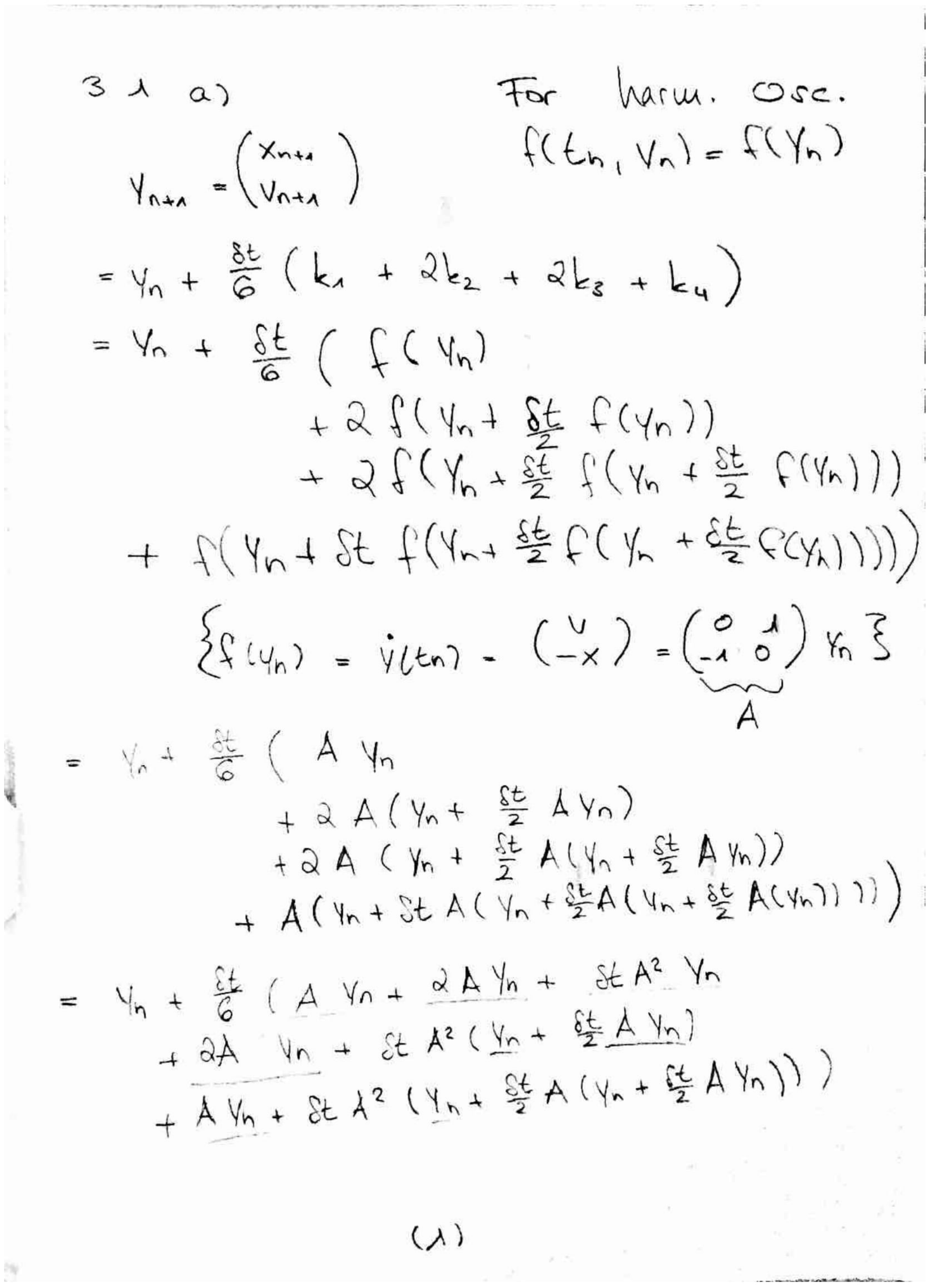
Paul Monderkamp, Matr.Nr. 2321677

monderkamp@thphy.uni-duesseldorf.de

Contents

1 Exercise 1

1.1 Exercise 1 (a)



3 1 a) For harm. osc.

$$Y_{n+1} = \begin{pmatrix} x_{n+1} \\ v_{n+1} \end{pmatrix} \qquad f(t_n, Y_n) = f(Y_n)$$

$$= Y_n + \frac{\delta t}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

$$= Y_n + \frac{\delta t}{6}\Big(f(Y_n) + 2f\big(Y_n + \tfrac{\delta t}{2} f(Y_n)\big) + 2f\big(Y_n + \tfrac{\delta t}{2} f(Y_n + \tfrac{\delta t}{2} f(Y_n))\big) + f\big(Y_n + \delta t\, f(Y_n + \tfrac{\delta t}{2} f(Y_n + \tfrac{\delta t}{2} f(Y_n)))\big)\Big)$$

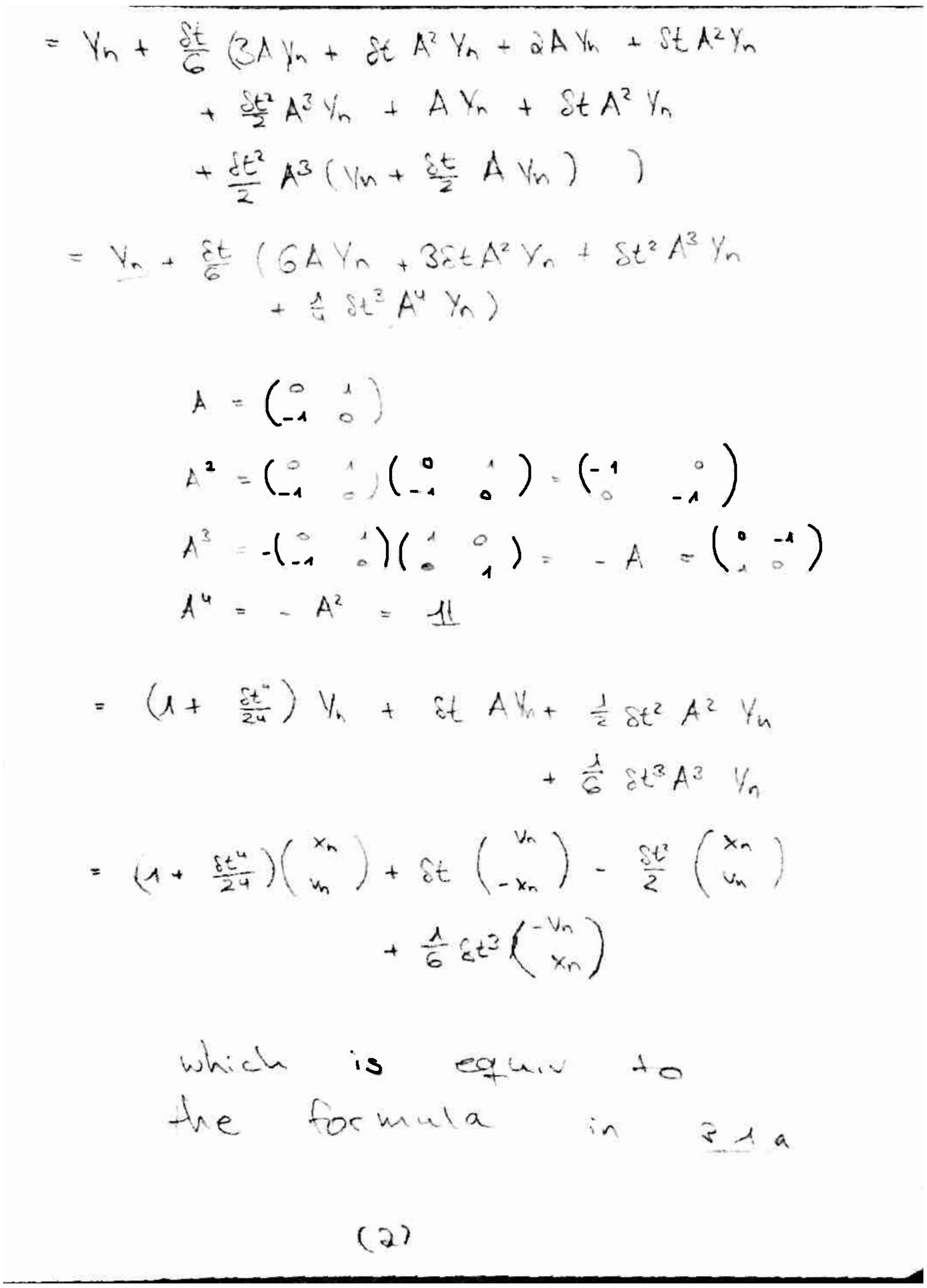
$$\left\{ f(Y_n) = \dot{Y}(t_n) = \begin{pmatrix} v \\ -x \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}}_{A} Y_n \right\}$$

$$= Y_n + \frac{\delta t}{6}\Big(A Y_n + 2A\big(Y_n + \tfrac{\delta t}{2} A Y_n\big) + 2A\big(Y_n + \tfrac{\delta t}{2} A(Y_n + \tfrac{\delta t}{2} A Y_n)\big) + A\big(Y_n + \delta t\, A(Y_n + \tfrac{\delta t}{2} A(Y_n + \tfrac{\delta t}{2} A(Y_n)))\big)\Big)$$

$$= Y_n + \frac{\delta t}{6}\Big(A Y_n + \underline{2 A Y_n} + \delta t A^2 Y_n + \underline{2A Y_n} + \delta t A^2 \big(\underline{Y_n} + \underline{\tfrac{\delta t}{2} A Y_n}\big) + \underline{A Y_n} + \delta t A^2 \big(\underline{Y_n} + \tfrac{\delta t}{2} A (Y_n + \tfrac{\delta t}{2} A Y_n)\big)\Big)$$

(1)

Figure 1.1: first part of the derivation of the formula



$$= Y_n + \frac{\delta t}{6}\Big(3A Y_n + \delta t A^2 Y_n + 2A Y_n + \delta t A^2 Y_n$$
$$+ \frac{\delta t^2}{2} A^3 Y_n + A Y_n + \delta t A^2 Y_n$$
$$+ \frac{\delta t^2}{2} A^3 \big(Y_n + \frac{\delta t}{2} A Y_n\big) \Big)$$

$$= Y_n + \frac{\delta t}{6}\Big(6 A Y_n + 3\delta t A^2 Y_n + \delta t^2 A^3 Y_n$$
$$+ \frac{1}{4} \delta t^3 A^4 Y_n\Big)$$

$$A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$
$$A^2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$$
$$A^3 = -\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = -A = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$
$$A^4 = -A^2 = \mathbb{1}$$

$$= \Big(1 + \frac{\delta t^4}{24}\Big) Y_n + \delta t A Y_n + \frac{1}{2} \delta t^2 A^2 Y_n$$
$$+ \frac{1}{6} \delta t^3 A^3 Y_n$$

$$= \Big(1 + \frac{\delta t^4}{24}\Big)\begin{pmatrix} x_n \\ v_n \end{pmatrix} + \delta t \begin{pmatrix} v_n \\ -x_n \end{pmatrix} - \frac{\delta t^2}{2}\begin{pmatrix} x_n \\ v_n \end{pmatrix}$$
$$+ \frac{1}{6} \delta t^3 \begin{pmatrix} -v_n \\ x_n \end{pmatrix}$$

which is equiv to
the formula in $\underline{3}$ 1 a

(2)

Figure 1.2: second part of the derivation of the formula

1.2 Exercise 1 (b)

1.2.1 Code

```
#include <iostream>
#include <cmath>
#include <fstream>
using namespace std;

int main()
{
double tmin       = 0.0;
double tmax       = 50.0;
const int N       = 500;
double dt         = (double)(tmax - tmin)/(N-1);

double y[N][4];
y[0][0] = 1.0;
y[0][1] = 0.0;
y[0][2] = 0.5*(y[0][0]*y[0][0]+y[0][1]*y[0][1]);
y[0][3] = 0.0;

for (int i=0;i<N-1;i++)
{
  y[i+1][0] = y[i][0] + y[i][1]*dt -0.5*y[i][0]*dt*dt-
    (1.0/6.0)*y[i][1]*pow(dt,3.0)+
    (1.0/24.0)*y[i][0]*pow(dt,4.0);
  y[i+1][1] = y[i][1] - y[i][0]*dt -0.5*y[i][1]*dt*dt+
    (1.0/6.0)*y[i][0]*pow(dt,3.0)+
    (1.0/24.0)*y[i][1]*pow(dt,4.0);
  y[i+1][2] = 0.5*(y[i+1][0]*y[i+1][0]+y[i+1][1]*y[i+1][1]);
  y[i+1][3] = dt*i;

}
ofstream outputfile;
outputfile.open("3_1_b_results.txt");

for (int i=0; i<N; i++)
{
 outputfile << y[i][0] << "␣␣" << y[i][1]
 << "␣␣" << y[i][2] << "␣␣" << y[i][3] << endl;
}
outputfile.close();
return 0;
}
```

1.2.2 Results

The following results are achived via the code in the previous subsection.

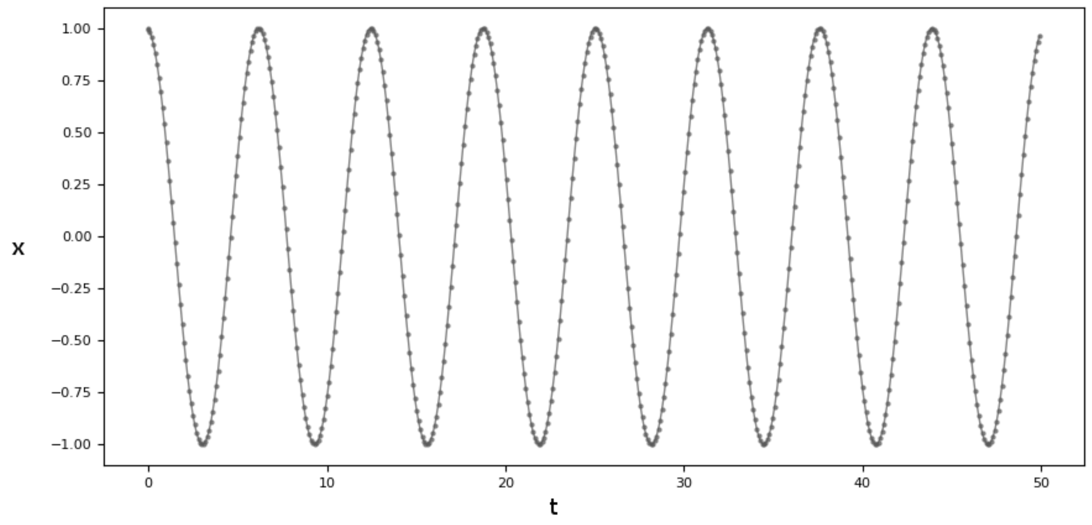


Figure 1.3: time evolution of the x coordinate of the harmonic oscillator

Figure 3.3 shows the time evolution of the x coordinate of a one dimensional harmonic oscillator solved with the Runge-Kutta 4 scheme. There is no apparent divergence from the analytical solution visible in the timeframe of the simulation.

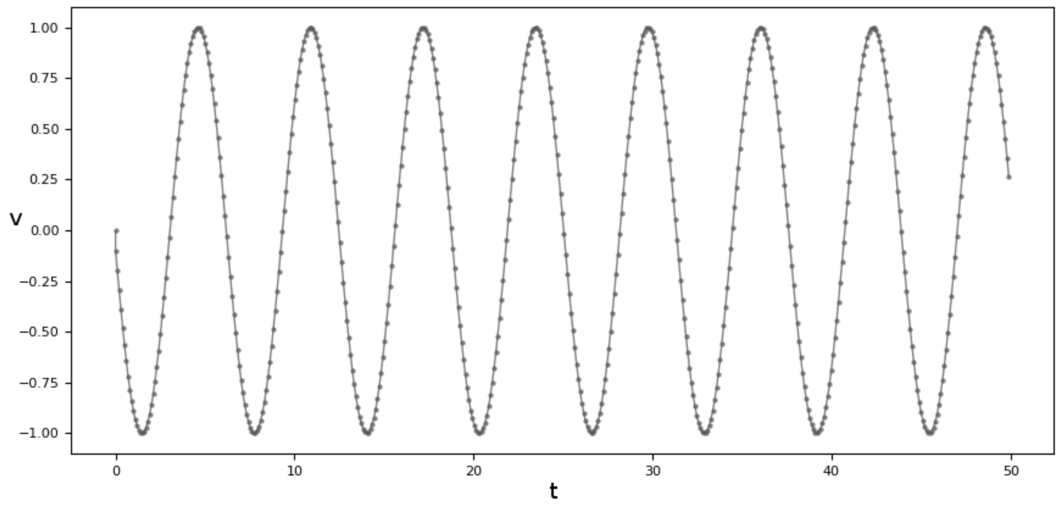


Figure 1.4: time evolution of the velocity of the harmonic oscillator

Figure 3.4 shows the time evolution of the velocity of a one dimensional harmonic oscillator solved with the Runge-Kutta 4 scheme. Similarly to the x coordinate, there is no visible instability in the timeframe of the simulation period.

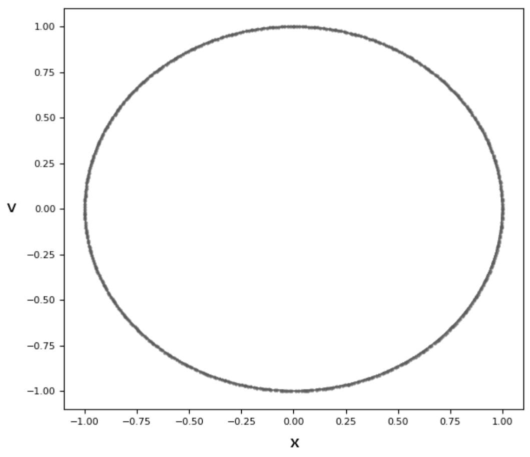
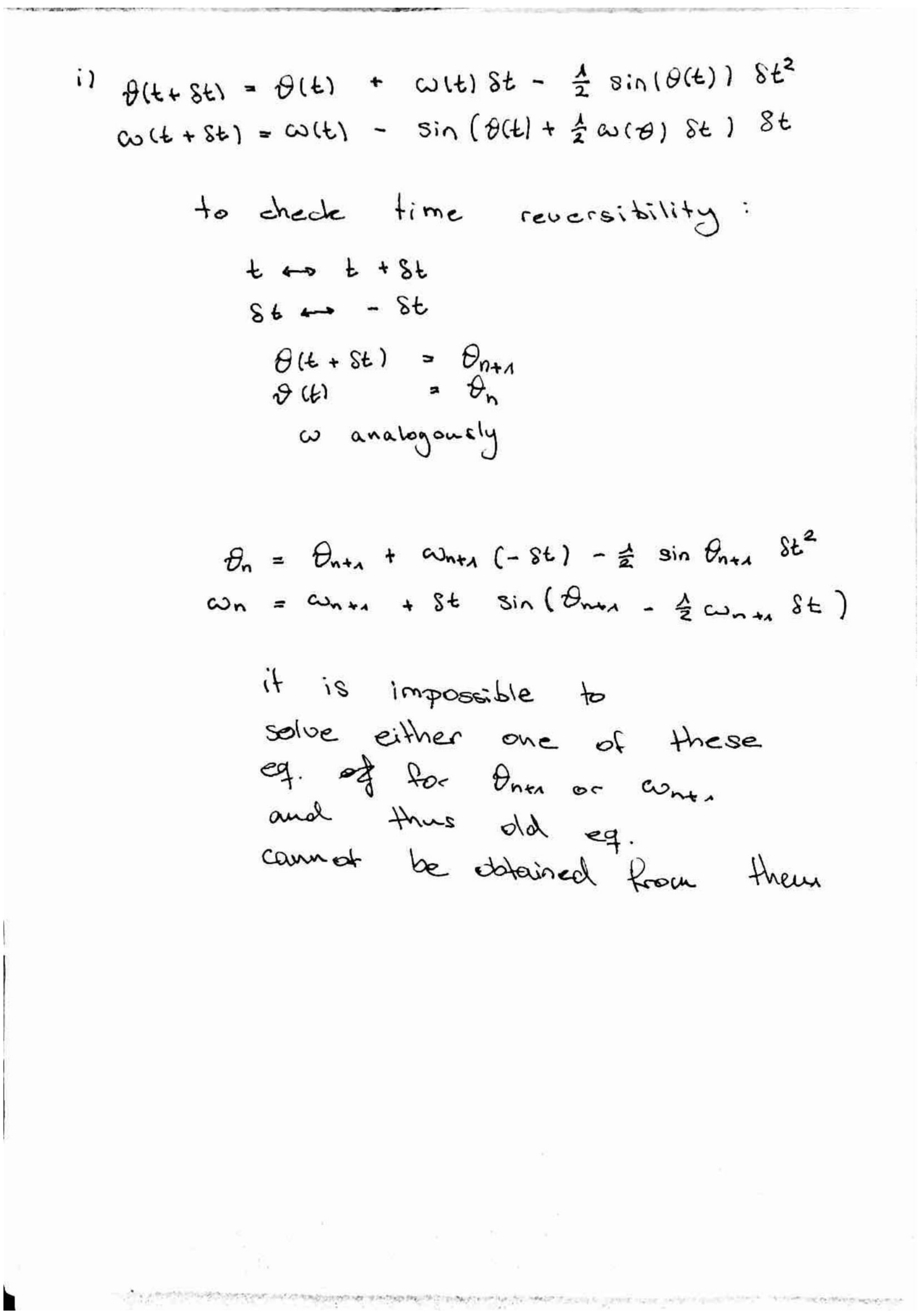


Figure 1.5: phase space trajectory of the harmonic oscillator

Figure 3.5 shows the phase space profile for the simulation results above. The trajectory seems reasonably spherical.

2 Exercise 2

2.1 Exercise 2 (a)



i) $\theta(t+\delta t) = \theta(t) + \omega(t)\,\delta t - \frac{1}{2}\sin(\theta(t))\,\delta t^2$

$\omega(t+\delta t) = \omega(t) - \sin(\theta(t) + \frac{1}{2}\omega(\theta)\,\delta t)\,\delta t$

to check time reversibility:

$t \leftrightarrow t+\delta t$

$\delta t \leftrightarrow -\delta t$

$\theta(t+\delta t) = \theta_{n+1}$

$\theta(t) = \theta_n$

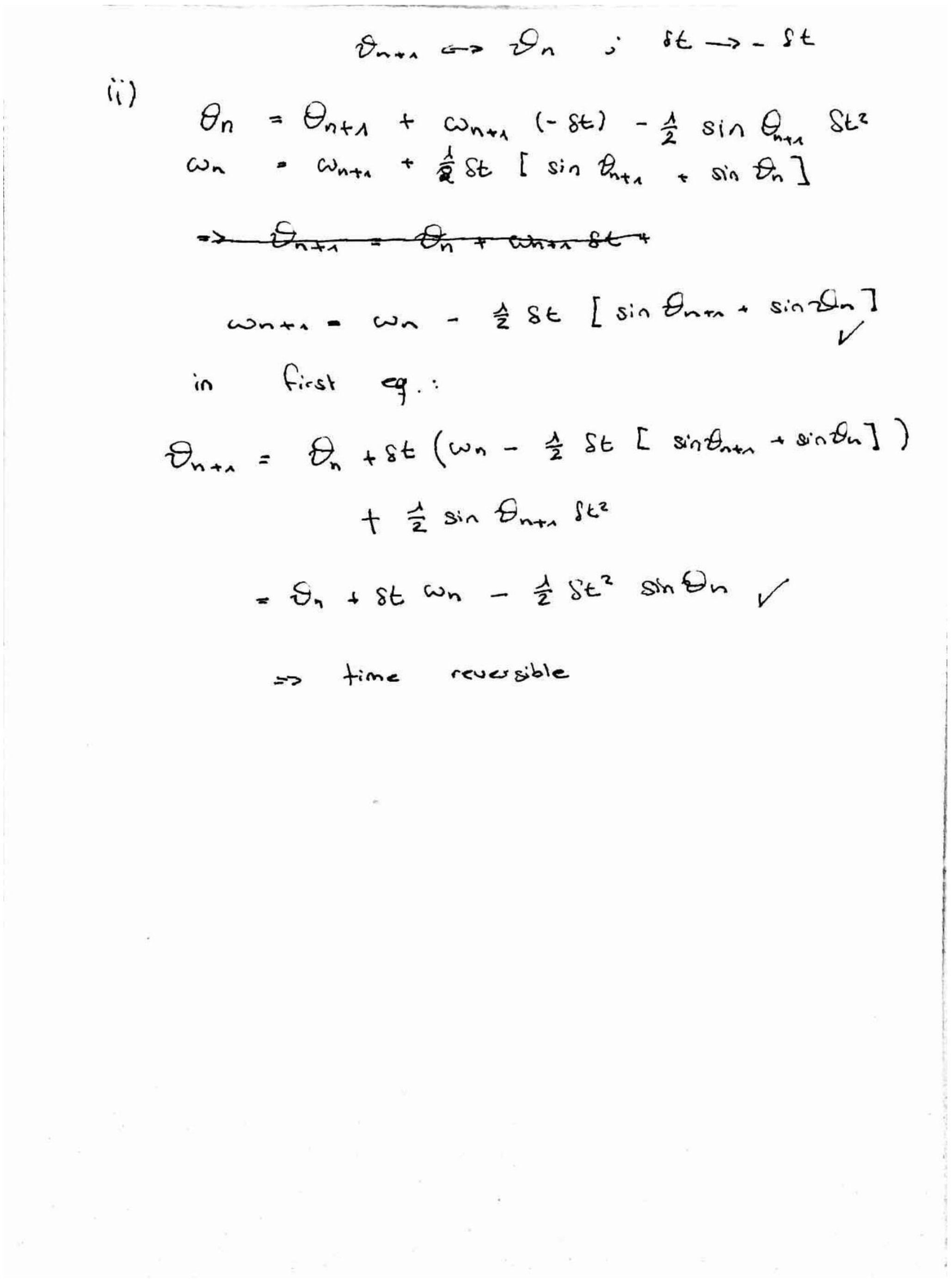
ω analogously

$\theta_n = \theta_{n+1} + \omega_{n+1}(-\delta t) - \frac{1}{2}\sin\theta_{n+1}\,\delta t^2$

$\omega_n = \omega_{n+1} + \delta t\,\sin(\theta_{n+1} - \frac{1}{2}\omega_{n+1}\,\delta t)$

it is impossible to
solve either one of these
eq. ~~of~~ for θ_{n+1} or ω_{n+1}
and thus old eq.
cannot be obtained from them

Figure 2.1: time reversibility for Runge-Kutta 2 scheme



$\vartheta_{n+1} \leftrightarrow \vartheta_n \; ; \; \delta t \to -\delta t$

(i)

$$\theta_n = \theta_{n+1} + \omega_{n+1}(-\delta t) - \frac{1}{2}\sin\theta_{n+1}\,\delta t^2$$

$$\omega_n = \omega_{n+1} + \frac{1}{2}\delta t\,[\sin\theta_{n+1} + \sin\theta_n]$$

~~$\Rightarrow \theta_{n+1} = \theta_n + \omega_{n+1}\delta t +$~~

$$\omega_{n+1} = \omega_n - \frac{1}{2}\delta t\,[\sin\theta_{n+1} + \sin\theta_n] \quad ✓$$

in first eq.:

$$\theta_{n+1} = \theta_n + \delta t\left(\omega_n - \frac{1}{2}\delta t\,[\sin\theta_{n+1} + \sin\theta_n]\right) + \frac{1}{2}\sin\theta_{n+1}\,\delta t^2$$

$$= \theta_n + \delta t\,\omega_n - \frac{1}{2}\delta t^2 \sin\theta_n \quad ✓$$

$\Rightarrow$ time reversible

Figure 2.2: time reversibility for velocity Verlet algorithm

2.2 Exercise 2 (b)

2.2.1 Code

```
#include <iostream>
#include <cmath>
#include <fstream>
#include <string>

using namespace std;

int main()
{
string timeinterval = "0_005";
string outputfilename = "3_2b_euler_dt_" + timeinterval + ".txt";

double tmin = 0.0;
double tmax = 50.0;
double dt = 0.005;

int N = (int)(tmax - tmin)/dt + 1;

double y[N][3];

y[0][0] = (7.0/360.0)*2.0*M_PI;
y[0][1] = 0.0;
y[0][2] = 50.0*y[0][1]*y[0][1]+100.0*(1.0-cos(y[0][0]));

ofstream out;
out.open(outputfilename);
out << y[0][0] << "␣␣" << y[0][1] << "␣␣" << y[0][2] << endl;

for (int i=0;i<N;i++)
  {
    y[i+1][0] = y[i][0] + y[i][1]*dt;
    y[i+1][1] = y[i][1] - dt*sin(y[i][0]);
    y[i+1][2] = 50.0*y[i+1][1]*y[i+1][1]+100.0*(1.0-cos(y[i+1][0]));

    out << y[i+1][0] << "␣␣" << y[i+1][1] << "␣␣" << y[i+1][2] << endl;
  }


out.close();
return 0;
}
```

The code above is the program with which the differential equation

$$\dot{\theta} = \omega$$

$$\dot{\omega} = -\frac{g}{L}\sin(\theta) \tag{2.1}$$

with $g = L = 10$ is solved according to the Euler algorithm. The Euler-Cromer algorithm follows the same scheme except the for loop with the iteration looks as follows:

```
for (int i=0;i<N;i++)
  {
    y[i+1][1] = y[i][1] - dt*sin(y[i][0]);
    y[i+1][0] = y[i][0] + y[i+1][1]*dt;
    y[i+1][2] = 50.0*y[i+1][1]*y[i+1][1]
       +100.0*(1.0-cos(y[i+1][0]));

    out << y[i+1][0] << "   " << y[i+1][1]
         << "   " << y[i+1][2] << endl;
  }
```

2.2.2 Results

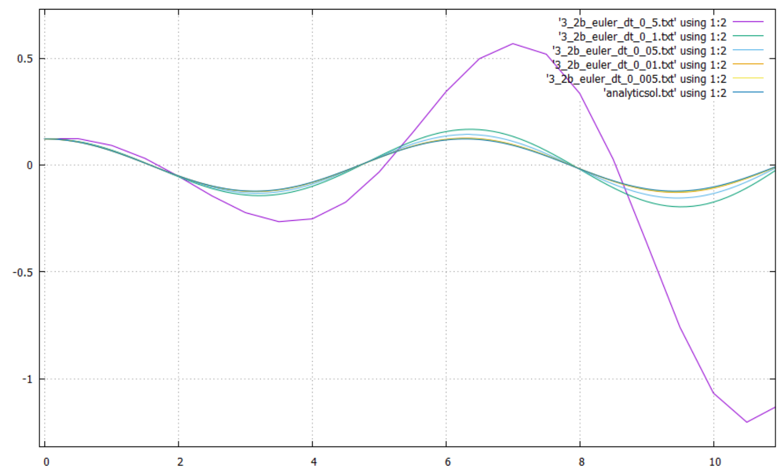


Figure 2.3: θ olutions for eq. (2.1) solved with Euler-alg.

Figure 2.3 shows the plots of the solutions of the angle θ for the set of differential equations (2.1) compared to the analytic solution. It already includes the plots from exercise (d) which corresponds to smaller timesteps.
The x axis in all following plots of θ and ω is in units of time whereas all following phace space profiles have θ on the x axis and ω on the y axis.
The most unstable solution which is the purple line corresponds to the biggest time step. This is to be expected since the instability scales with the size of the timestep.

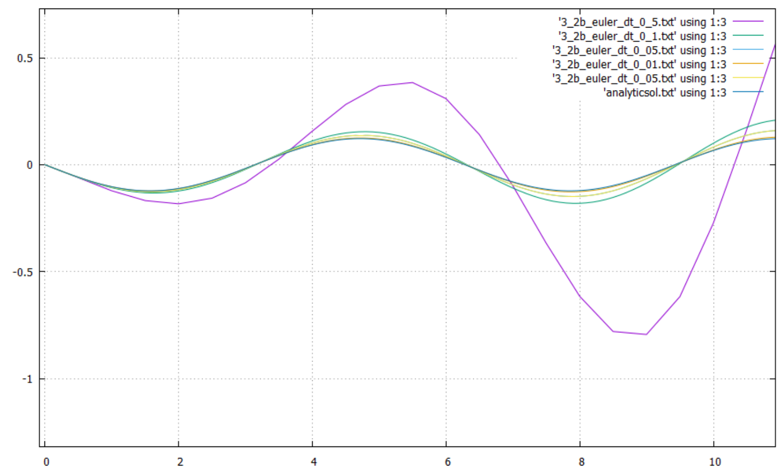


Figure 2.4: ω solutions for eq. (2.1) solved with Euler-alg.

In analogy Figure 2.4 shows the solutions for the angular velocity ω.

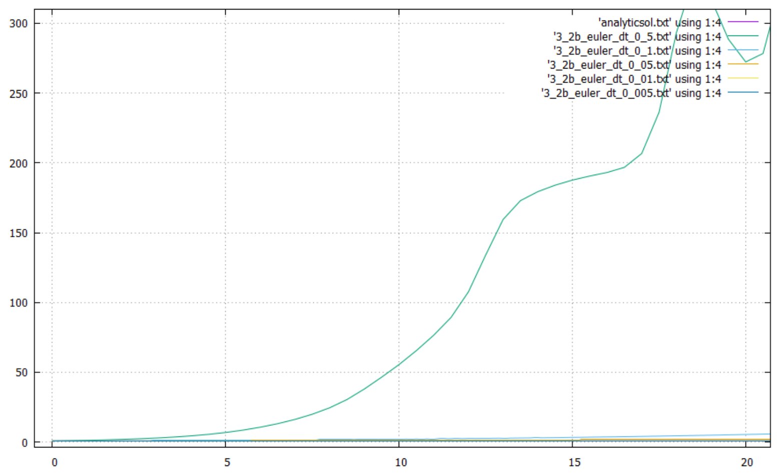


Figure 2.5: Energy solutions for eq. (2.1) solved with Euler-alg.

Figure 2.5 shows the time evolution of the solutions for the energy according to the results above. Is has to be mentioned that the time span of the simulation runs from $t = 0$ to $t = 50$. However the plots above only show partial ranges. This is due to the domination of the diverging solution with the biggest time step. Showing the results on the whole scale makes most of the lines indistinguishable.

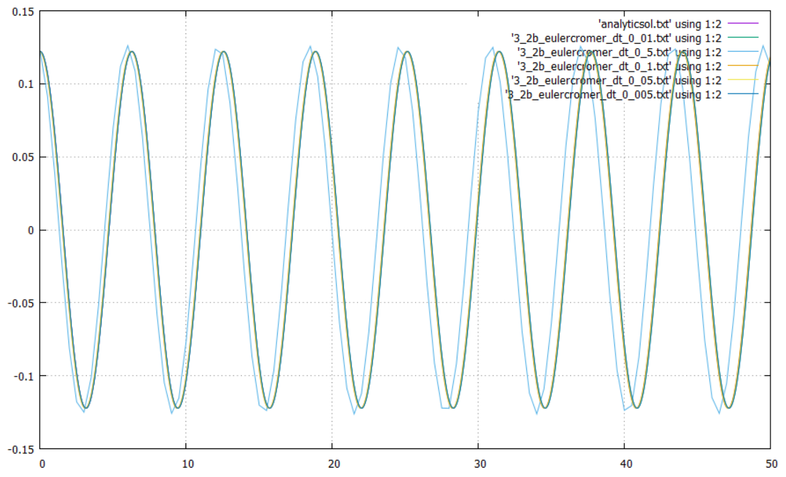


Figure 2.6: θ solutions for eq. (2.1) solved with Euler-Cromer-alg.

Figure 2.6 shows the time propagation for the angle θ for the same equations as given by the Euler-Cromer algorithm. It is visible, that the amplitude does not converge with respect to the analytical solutions. However the solution with the biggest time step shows a phase shift.

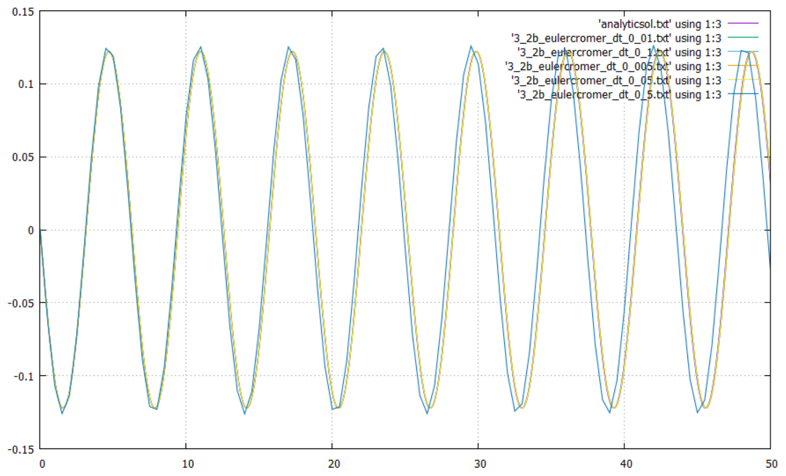


Figure 2.7: ω solutions for eq. (2.1) solved with Euler-Cromer-alg.

The plots for the numerical solutions of the angular velocity have the same appearance and show the same properties.

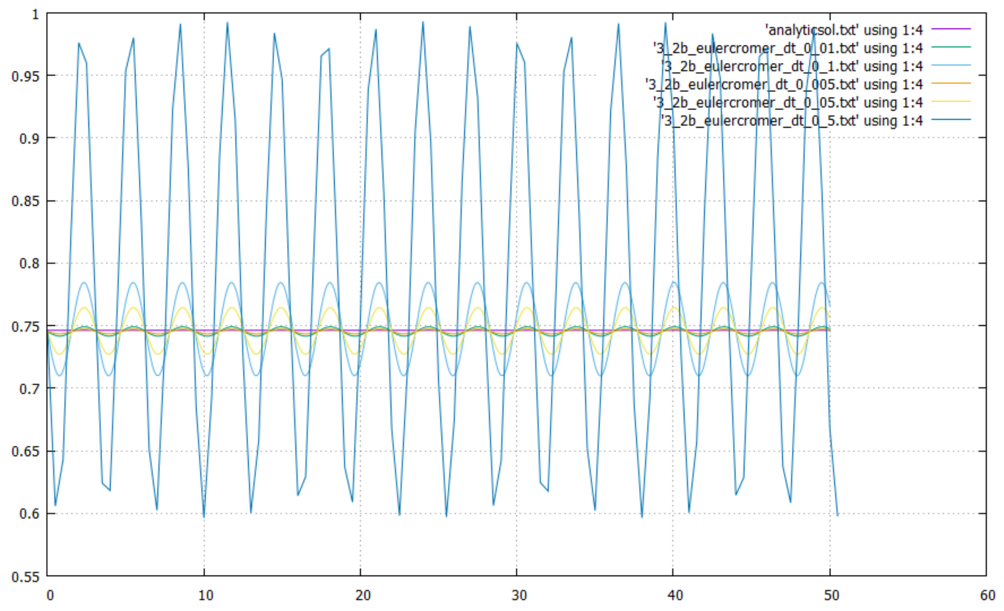


Figure 2.8: Energy solutions for eq. (2.1) solved with Euler-Cromer-alg.

Figure 2.8 shows the Energy of the system as a function of time. As discussed in previous exercises and in the lecture, the energy of the shadow Hamiltonian oscillates around the constant energy of the analytic result. The amplitude of this oscillation grows with the time step, but stays bounded in contrast to the solutions of the usual Euler scheme.

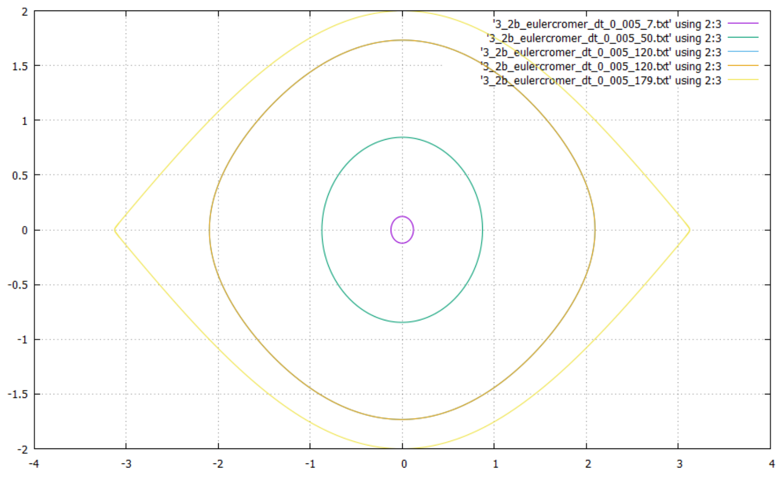


Figure 2.9: Phase space profiles for different starting angles θ_0

Figures 2.9 and 2.10 show the phase space profiles for different initial angles θ_0 and thus for different regimes: harmonic, oscillatory and rotational. The behaviour gets less harmonic, the bigger the amplitudes of the angles get.
A starting angle of $\theta_0 = \pi$ would in principle correspond to an unstable equilibrium. After an infinitesimal displacement however the pendulum would swing in accordance to the light yellow line on the outside of the pendulum (fig 2.9). This energy level in space space divides the oscillatory energies from the rotational energies. The latter have enough energy to swing in circles around the whole range of the configuration angle θ and thus the system propagates along the θ axis of space space without generating bound states.

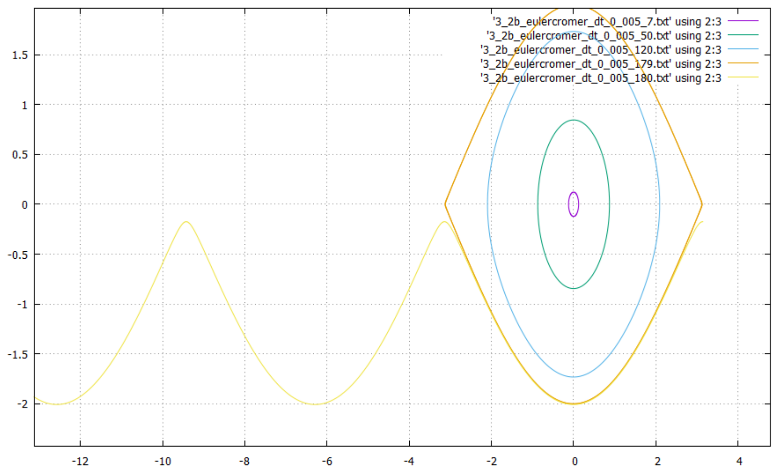


Figure 2.10: Phase space profiles for different starting angles θ_0