Department of Physics

Computational Physics

Exercise Sheet 02

Rene-Marcel Lehner

SuSe 2023

Disclaimer[5]

The emergence of ChatGPT presses us to rethink how we work and learn. Especially with assignments like these, ChatGPT can be heavily relied on. Tasks to be relied on include coding, generating text, sourcing information and even interpreting or predicting results and underlying principles of physics.

Consequently, it is the responsibility of the student to carefully consider how to use this emerging tool. However, using a tool like ChatGPT can also potentially enhance the workflow dramatically; without diminishing or disrupting learning, or even improving it.
To clarify in what context ChatGPT is used in this assignment and to help the tutors create a more distinguished assessment, a brief explanation of the usage is provided.

1. If an enumeration like this appears, it does **not** mean that it was created by, sourced or directly copied from ChatGPT.

2. **Text improvements** like wording, phrases, more variety in linking words and semantics as well as grammar. These changes are rather small and aim to improve sentence flow and readability.

3. **Information** on topics or methods are often sourced from ChatGPT. Its ability to summarize topics is immensely helpful and makes learning more effective. The information is always cross-checked with the literature.

4. **Code** is **never** directly copied from or completely provided by ChatGPT. Some code is generated for demonstration purposes, but the focus lies on explaining the syntax and suggesting improvements to existing code.

Overall, ChatGPT is used to minimize the work load while improving the learning experience. It is also used to create learning material and feedback, for instance vocabulary lists or feedback on the text or code which was written by the student.
Content that is generated by ChatGPT is marked explicitly; and limited to tasks that shouldn't affect grading.
ChatGPT is a great tool[6], but it also potentially complicates the evaluation process for the tutors.
This disclaimer aims to facilitate the grading process.

[5]This disclaimer was not written, revised or assisted by ChatGPT in any way.
[6]Word count of "ChatGPT" in this assignment: 15

E1: Comprehension questions[7]

Task:

1. What is a symplectic integration method? Which benefits do they offer?

A symplectic integration method is a numerical technique used for solving Hamiltonian systems, which often arise in classical mechanics and other areas of physics. These methods are designed to preserve the geometric structure of the phase space, which is particularly important for long-time simulations and the study of the qualitative behavior of dynamical systems.
The term "symplectic" comes from the underlying mathematical structure called a symplectic manifold, which is the natural geometric setting for Hamiltonian systems. The preservation of this structure is what makes symplectic integrators advantageous over other numerical methods in certain applications.
Benefits of symplectic integration methods include:

1. Long-term stability: Symplectic integrators are well-suited for long-time simulations because they exhibit better energy conservation and bounded error growth compared to non-symplectic methods. This helps to avoid the accumulation of numerical errors that could lead to unphysical results.

2. Preservation of geometric properties: By preserving the symplectic structure of the phase space, these integrators maintain important geometric properties, such as the conservation of certain quantities (e.g., angular momentum) and the topology of the trajectories.

3. Qualitative accuracy: Symplectic integrators can capture the qualitative behavior of Hamiltonian systems more accurately than non-symplectic methods. This is particularly important for studying phenomena like the motion of celestial bodies, where the long-term behavior of the system is of interest.

4. Efficiency: Symplectic integrators are often more computationally efficient than other methods for solving Hamiltonian systems, especially when the conservation of geometric properties is crucial.

Some popular symplectic integration methods include the Verlet algorithm, the Stormer-Verlet method, and various higher-order methods like the Ruth or Yoshida integrators. These methods have been widely used in molecular dynamics, celestial mechanics, and other fields where the accurate simulation of Hamiltonian systems is essential.

[7]100% ChatGPT.

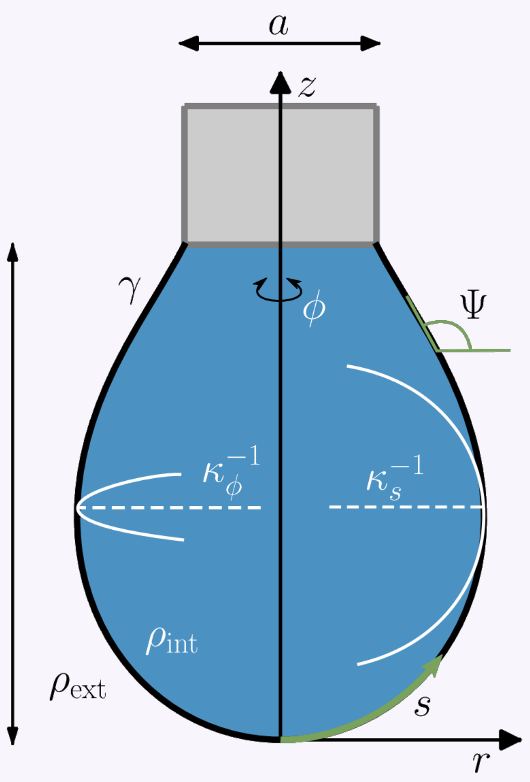
E2: Shape equations of a liquid droplet

Task:

A fluid droplet with density ρ is suspended from a capillary with diameter a. The force balance between surface tension γ, pressure difference across the interface $p(z)$ and gravity g yields the Young-Laplace equation, which links the principal curvatures of the interface κ_s and κ_ϕ with the pressure and the surface tension:

$$p(z) = p_L - \rho g z = \gamma\left(\kappa_s + \kappa_\phi\right) \qquad (1.1)$$

where p_L is the pressure difference across the interface at the drop apex and $\rho g z$ the hydrostatic pressure contribution. We will focus on axisymmetric drops, such that a parametrization of the interface is possible with a line segment that can be rotated around the z-Axis to give the full contour of the drop. The line segment has an arc length coordinate s and an arc angle $\Psi \in [0, \pi]$, which is measured relative to the horizontal plane.



The shape equations thus consist of two purely geometric equations and an additional equation from (1.1):

$$\begin{aligned}
\frac{\mathrm{d}r}{\mathrm{d}s} &= \cos(\Psi) \\
\frac{\mathrm{d}z}{\mathrm{d}s} &= \sin(\Psi) \\
\frac{\mathrm{d}\Psi}{\mathrm{d}s} &= \kappa_s = \frac{p_L}{\gamma} - \frac{\rho g z}{\gamma} - \frac{\sin\Psi}{r}
\end{aligned} \qquad (1.2)$$

a) Use the surface tension γ and the capillary diameter a as scales to non-dimensionalize the shape equations (1.2).

b) The shape equation $\frac{\mathrm{d}\Psi}{\mathrm{d}s}$ contains a numerical singularity at the apex $(s = 0)$, where $\sin\Psi = 0$ and $r = 0$. Show that the limit of this shape equation is given by:

$$\frac{\mathrm{d}\Psi}{\mathrm{d}s}(s \to 0) = \frac{p_L - \rho g z}{2\gamma}$$

and use this result in your numerical solution.

c) Integrate the (dimensionless) shape equations with a 4th order Runge-Kutta method and solve for the shapes with $p_L a/\gamma = 2$ and $\rho g a^2/\gamma = 0.1$ with the addtional conditions:

Zur Verifikation: Vereinfachtes Problem überlegen, bspw. Gravitation ausschalten.

$$r(s=0)=0$$
$$z(s=0)=0$$
$$\Psi(s=0)=0$$
$$r\left(s=s_{\max}\right)=a/2$$

Hint: The condition $r\left(s=s_{\max}\right)=a/2$ can be satisfied by stopping the integration at an arbitrary point where $r(s)=a/2$. Also include solutions where $r(s)=a/2$ is fulfilled multiple times during the integration.

d) Explore the parameter space of the problem and record a pressure volume curve for the solutions with $\rho g a^2/\gamma=0.5$ and $p_L a/\gamma\in[0,5]$.

Non-dimensionalization of the shape equations[8]

Non-dimensionalizing a problem (in this instance, the shape equations) prior to solving it with a computer hosts a number of benefits, ranging from improved efficiency to increased comparability of the results.
We discussed this already in our tutorial group. Since these submissions also serve as learning material for the oral examination, the advantages of non-dimensionalization are outlined more rigorously here.
The keywords are **comparability, efficiency, numerical stability** and **simplification**.

1. **Comparability of results**: By making equations dimensionless, the results become independent of specific units, which facilitates the comparison of solutions from different applications or systems.
2. **Efficiency**: Dimensionless equations are often simpler and faster to solve since they require fewer computational resources. This is especially important when solving complex or large-scale problems.
3. **Improved numerical stability**: Dimensionless analysis helps balance the orders of magnitude of the involved quantities (as mentioned in the tutorial group). This avoids numerical problems, such as the cancellation of very large or very small numbers. This contributes to increasing the stability and accuracy of the numerical solutions.
4. **Simplification of equations**: Dimensionless analysis eliminates the size ratios and units of the variables, often leading to simpler equations. This facilitates understanding the underlying physics and can also help reduce the number of parameters in the equations.

---

a) The task provides the capillary diameter a (*unit*: length) and the surface tension γ (*unit*: energy/area or force/length) as **characteristic quantities**. The variables are r, s, z and Ψ.

While Ψ is already dimensionless, r, s, z have to be non-dimensionalized. Since these variables refer to lengths, we can use the characteristic quantity a to calculate the dimensionless shape equations.

$$R=\frac{r}{a} \quad , \quad Z=\frac{z}{a} \quad , \quad S=\frac{s}{a} \quad , \quad P_L=\frac{p_L a}{\gamma}=\left[\frac{N}{m^2}\cdot\frac{m}{N}\cdot m\right]=[1]$$

[8]Regarding the advantages: Most of the information was sourced from ChatGPT, but the actual text is **not** a direct copy.

Introducing these substitutions to the original shape equations, we get the dimensionless shape equations:

$$\boxed{\frac{\mathrm{d}R}{\mathrm{d}S} = \frac{\mathrm{d}r}{\mathrm{d}s} = \cos(\overset{\varphi}{\psi}) \quad , \quad \frac{\mathrm{d}Z}{\mathrm{d}S} = \frac{\mathrm{d}z}{\mathrm{d}s} = \sin(\psi)}$$

$$\boxed{\frac{\mathrm{d}\Psi}{\mathrm{d}S} = \kappa_s a = P_L - \frac{\rho g Z a^2}{\gamma} - \frac{\sin\Psi}{R}}$$

b) To calculate the numerical singularity at the apex, a careful inspection of the differential equation (and its dependencies of s) is necessary.

We can emphasize this dependency by writing

$$\lim_{s\to 0}\frac{\mathrm{d}\Psi(s)}{\mathrm{d}s} = \lim_{s\to 0}\left(\frac{p_L}{\gamma} - \frac{\rho g z(s)}{\gamma} - \frac{\sin(\Psi(s))}{r(s)}\right) = \frac{p_L}{\gamma} - \frac{\rho g z}{\gamma} - \lim_{s\to 0}\frac{\sin\left(\Psi(s)\right)}{r(s)} \quad .$$

For the last term we used $\sin\Psi = 0$ for $s = 0$, which renders the first derivative and thus the change of z near $s = 0$ to zero as well; effectively remaining z itself.

Further calculation of the limit leads to "$\frac{0}{0}$", which can be solved analytically by applying **L'Hôspital's rule**. Using the chain rule is vital:

$$\lim_{s\to 0}\frac{\mathrm{d}\Psi}{\mathrm{d}s} = \frac{p_L - \rho g z}{\gamma} - \lim_{s\to 0}\frac{\frac{\mathrm{d}\Psi}{\mathrm{d}s}\cdot\cos(\Psi(s))}{\cancel{1}\cos(\Psi(s))}$$

Consideration of the specifications of the problem, specifically $\Psi \in [0,\pi]$, the interpretation of the figure and $\Psi(s=0) = 0$, we can assume that $\lim_{s\to 0}\cos(\Psi(s)) = 1$, which ultimately leads to

$$\lim_{s\to 0}\frac{\mathrm{d}\Psi}{\mathrm{d}s}(1 + \underbrace{\cos\Psi}_{\to 1}) = \frac{p_L - \rho g z}{\gamma}$$

$$2\lim_{s\to 0}\frac{\mathrm{d}\Psi}{\mathrm{d}s} = \frac{p_L - \rho g z}{\gamma}$$

$$\boxed{\lim_{s\to 0}\frac{\mathrm{d}\Psi}{\mathrm{d}s} = \frac{p_L - \rho g z}{2\gamma}} \quad .$$

c) Integrating the shape equations with the RK4 model generate results as shown in Figure 1.1. There are multiple solutions to $R = 1/2$, which require a precise truncation of the data.

Both drop shapes seem plausible. However, mathematically the second drop is the shape which is calculated by integrating the shape equations, since $\Psi \in [0, \pi]$.

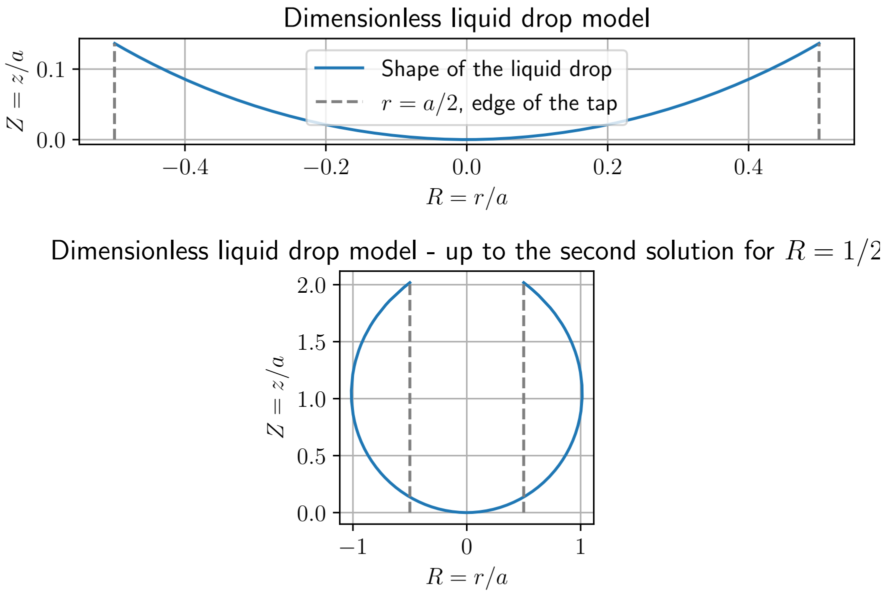
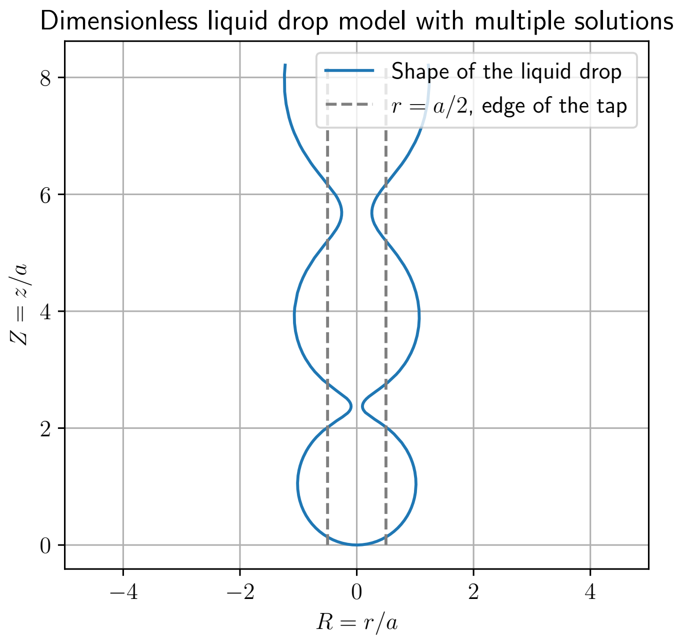


Figure 1.1: c) Dimensionless liquid drop model in the range of $R = \pm 0.5$. The edge of the valve or tap is marked by grey lines. For demonstration purposes, both representations are included.

Figure 1.2 demonstrates the overall behavior of the integrated shape equations when enabling multiple solution. This very same behavior can be used to explain the abrupt changes in volume, as seen in Figure 1.3.



Alle Lösungen für $R = 1/2$ sind gültig!

Figure 1.2: c) Dimensionless liquid drop model with no limits to the solution. Multiple solutions for $R = \pm 0.5$ are displayed by the intersection of the vertical lines with the curve.

d) In this subtask, the volume is computed by implemented a well-known formula for rotating an area around the y-axis, which facilitates the calculation of a three-dimensional volume rather than a mere surface area.

Incorporating the data up to the second solution for $R = 1/2$ posed a significant challenge during the implementation process. Consequently, only the volume for drop shapes up to the first solution for $R = 1/2$ were considered. Nevertheless, the general behavior is expected to remain unaltered.

The volume exhibits a monotonous increase up to $P_L = 4$, within the same order of magnitude. Beyond $P_L = 4$, the behavior alters drastically, characterized by three prominent spikes in volume followed by brief periods of a moderate yet consistent decline.

A closer examination of the drop shapes unveils that the drop shape becomes exceedingly shallow, leading to the formation of elongated drops reminiscent of flowing water. Whether these results are open to interpretation or possess any physical relevance remains a subject for further discussion.

For this calculation, the RK4 algorithm had to be run in multiple iterations, in this case $N = 200$.

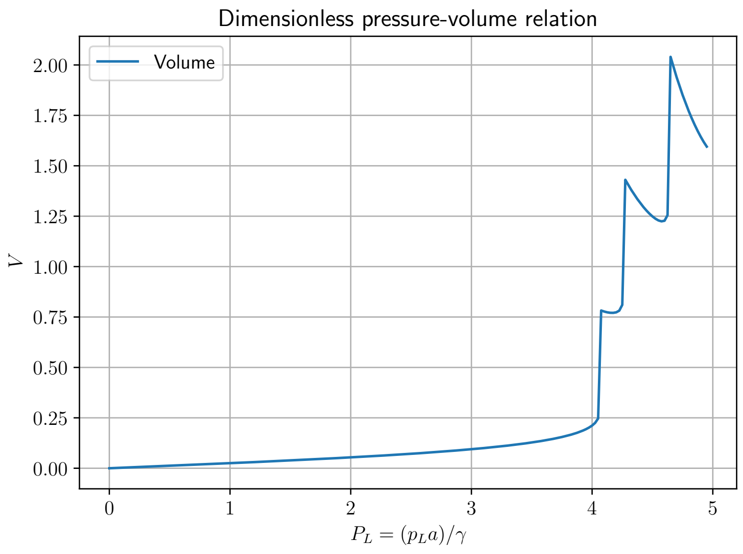


Figure 1.3: d) Dimensionless pressure-volume relation. Past $P_L = 4$, the drop gets too shallow to be confined by the edges of the valve. Elongated drops emerge, which vaguely resemble flowing water.

E3: Two-body problem - Comparison of two integrators

Task:

In this assignment, you will compare the the numerical integration of ordinary differential equations using the Euler- and Verlet algorithm.
Consider two masses m_1, m_2 at positions $\boldsymbol{r}_1, \boldsymbol{r}_2$ and velocities $\boldsymbol{v}_1, \boldsymbol{v}_2$ under influence of Newton's law of gravitation

$$\boldsymbol{F}(\boldsymbol{r}_1, \boldsymbol{r}_2) = -G\frac{m_1 m_2}{|\boldsymbol{r}_1 - \boldsymbol{r}_2|^3}(\boldsymbol{r}_1 - \boldsymbol{r}_2)$$

where the force $\boldsymbol{F}$ acts on the mass m_1 and correspondingly $-\boldsymbol{F}$ on mass m_2. The constant of gravitation can be set to 1 by an appropriate choice of units and the motion is considered only in the orbital plane. Use the initial conditions $m_1 = 1, m_2 = 2, \boldsymbol{r}_1(0) = (0,1)^T, \boldsymbol{r}_2(0) = (0,-0.5)^T$, $\boldsymbol{v}_1(0) = (0.8,0)^T, \boldsymbol{v}_2(0) = (-0.4,0)^T$ and integrate at least up to the time $T = 100$.

a) Implement both mentioned integration methods. Choose one adequate and one too coarse step size for each algorithm and plot the corresponding trajectories $\{(x,y)\}$ in comparison to each other.

b) Choose a reliable step size for each method and compare the resulting run times.

c) After integrating to $T = 100$, reverse time (negative step size) and integrate the trajectories back to $T = 0$. Comment on your result for the different integrators.

All subtasks are calculated using `C++`.
Plots are generated using `Python`.

Euler and Verlet algorithm

Before implementing the numerical methods, characteristic formulas and properties of both algorithms are provided.

Euler

Startwert

$$\begin{aligned} \boldsymbol{r}_{n+1,i} &= \boldsymbol{r}_{n,i} + \boldsymbol{v}_{n,i}\Delta t \\ v_{n+1,i} \quad \boldsymbol{v}_{n,i} &= \boldsymbol{a}_{n,i}\Delta t + v_{n,i} \\ m_i \boldsymbol{a}_{n,i} &= \boldsymbol{F}(\boldsymbol{r}_1, \boldsymbol{r}_2) \quad , \qquad i \in 1,2 \end{aligned}$$

→ Ort, Geschw, <u>gleichzeitig</u> berechnen.

Euler's method is a first order numerical integration method, offering a fast and easy-to-understand implementation. In contrast to Verlet's method, Euler is **not** a symplectic integration method and, therefore, does not preserve the symplectic structure of a Hamiltonian system.

We expect reduced long-term accuracy and stability, particularly when comparing the results to those obtained using Verlet's method. Furthermore, Euler's method is particularly susceptible to a loss in accuracy and stability when implementing a time-reversed integration. The conservation of energy or other quantities is likely to be compromised if the symplectic structure and its associated phase space properties are not preserved.

Verlet

$$
\begin{aligned}
\boldsymbol{r}_{n+1,i} &= 2\boldsymbol{r}_{n,i} - \boldsymbol{r}_{n-1,i} + \boldsymbol{a}_{n,i}\Delta t^2 + \mathcal{O}\left(\Delta t^4\right) \\
\boldsymbol{v}_{n,i} &= \frac{1}{2\Delta t}\left(\boldsymbol{r}_{n+1,i} - \boldsymbol{r}_{n-1,i}\right) + \mathcal{O}\left(\Delta t^2\right) \\
\boldsymbol{r}_{-1,i} &= \boldsymbol{r}_{0,i} - \boldsymbol{v}_{0,i}\Delta t + \frac{1}{2}\boldsymbol{a}_{0,i}\Delta t^2 \quad , \qquad i \in 1, 2
\end{aligned}
$$

✓

Verlet's method preserves the symplectic structure and we expect an improved time-reversed integration. In addition to Euler's algortihm, Verlet uses the **previous** position to update the position and velocity. It is a second-order numerical integration method, which makes it naturally more accurate than Euler's method.

a) **Coarse step size ($\Delta t = 0.1$)**

In this subtask both integration methods are implemented in two iterations. The first iteration uses a more coarse step size to deliberately destabilize the algorithms. The trajectories are similar to rhodonea curves ("Rosettenbahnen"), which commonly occur when a small perturbation is applied to a central potential.This behavior is demonstrated in Figure 1.4. There are no overly apparent differences, neither in accuracy nor stability.

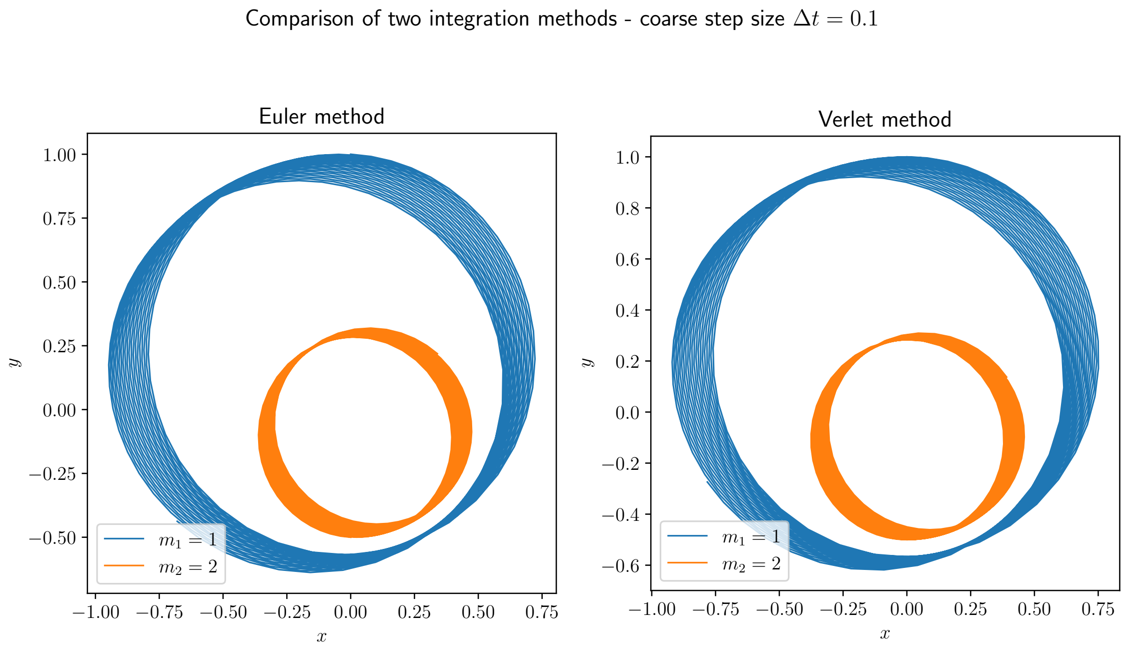


Figure 1.4: a) Choice of a coarse steps size to deliberately destabilize the numerical integration. $\Delta t = 0.1$

Adequate step size ($\Delta t = 0.01$)

By choosing a smaller step size, the overall accuracy of both algorithms can be improved significantly, resulting in very stable trajectories with no apparent deviations.

Furthermore, both algorithms yield remarkably similar results, effectively rendering the integration for both methods equivalent.

The results are demonstrated in Figure 1.5.

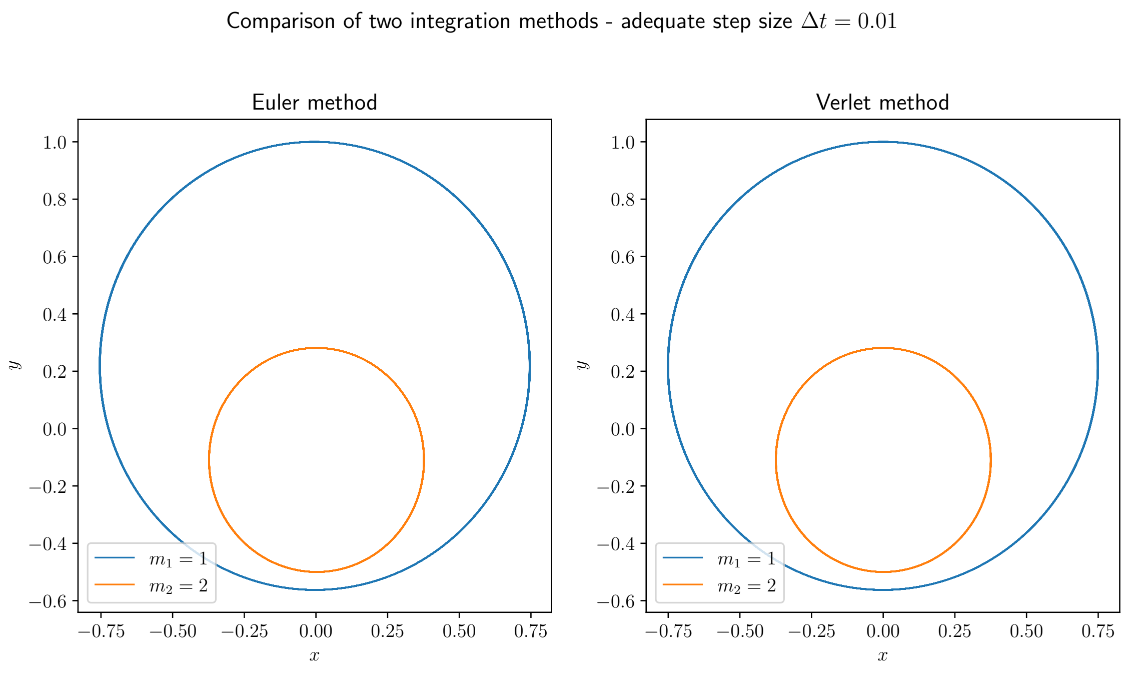


Figure 1.5: a) Choice of adequate step size to demonstrate the potential accuracy of both algorithms. $\Delta t = 0.01$

---

b) **Computation time comparison**

Both integration algorithms are computed using the adequate step size ($\Delta t = 0.01$).

While executing and measuring the computations, major fluctuations in computation times were observed. To mitigate these fluctuations, the computation was executed multiple times. By doing so, it allowed for the calculation of a mean and a standard deviation, which provided a clearer and more expressive representation of the computation times.

Additionally, the execution was done in parallel, which further mitigated potential fluctuations.

Results

```
Time needed for Euler in µs: 1481 +-194
Time needed for Verlet in µs: 1552 +-129
Iterations, N = 1000
Euler's method is 4.6145% faster.
```

c) **Forward and reversed integration ($\Delta t = 0.01$)**

The existing code was adapted to enable time-reversed integrations.

As previously discussed, both algorithms exhibit fundamentally differences in conserving the symplectic structure of Hamiltonian systems. Symplectic structures contribute to the conservation of specific quantities (e.g., energy, momentum, angular momentum) and posses the inherent property of being time-reversible.

Consequently, Verlet's algorithm is expected to produce markedly more accurate and stable results compared to Euler's method.

A comparison of both integrations in Figure 1.6 confirms these assumptions. However, deviations from the reversed trajectories might not be as distinctive as in earlier examinations with a coarser step size.

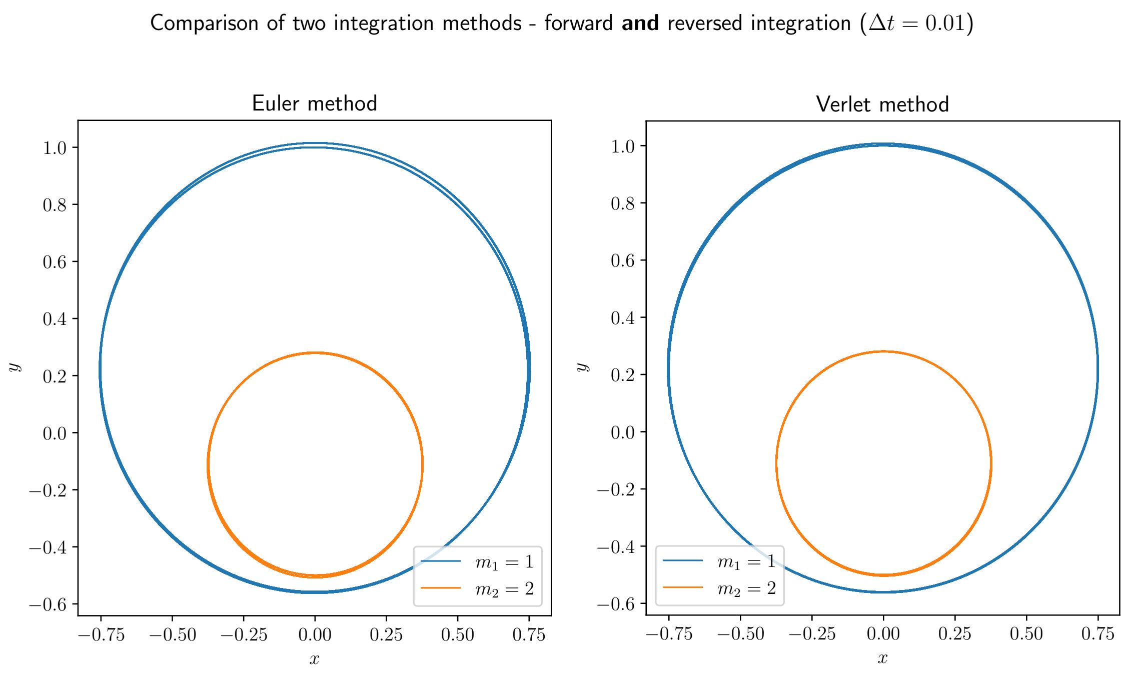


Figure 1.6: c) Time-reversed integration. As discussed, we expect Euler's method to be unstable.

Coding
A more transparent documentation of the execution of the code is provided by printing expressive statements to the console. Additionally, the imported libraries (`#include` s) are clustered for clarification.
If everything works as intended, the console should output the following:

```
./outDebug
a)
File "Euler_dt00_1.csv" written.
1001 lines written.
File "Verlet_dt00_1.csv" written.
1001 lines written.
File "Euler_dt00_01.csv" written.
10000 lines written.
File "Verlet_dt00_01.csv" written.
10000 lines written.

b)
Time needed for Euler in µs: 1493 +-77
Time needed for Verlet in µs: 1569 +-84
Iterations, N = 1000
Euler's method is 4.87266% faster.

c)
File "Euler_reverse_dt00_01.csv" written.
20000 lines written.
File "Verlet_reverse_dt00_01.csv" written.
20000 lines written.
```

To assess the basic structure of the code, an outline[9] is provided:
E3outline.cpp

```
// Basic headers
#include <iostream>
#include <cmath>
// Containers
#include <vector>
#include <tuple>
// File handling
#include <fstream>
// For the time measurement
#include <chrono>
// For mean and std calculation
#include <algorithm>
#include <numeric>

using namespace std;

const double G = 1.0;
const double m1 = 1.0;
const double m2 = 2.0;

tuple<double, double> force(double x1, double y1, double x2, double y2) {
    double dx = x1 - x2;
    double dy = y1 - y2;
    double distance = sqrt(dx * dx + dy * dy);
    double distance_cubed = distance * distance * distance;
    double f = -G * m1 * m2 / distance_cubed;
    return {f * dx, f * dy};
}

vector<tuple<double, double, double, double, double, double, double, double>>
    simulate(string method, double dt, double T, double t = 0.,

        double x1 = 0., double y1 = 1., double x2 = 0., double y2 = -0.5,

        double vx1 = 0.8, double vy1 = 0., double vx2 = -0.4, double vy2 = 0.) {
    // Calculate initial acceleration and previous positions for Verlet method
    // ...

    vector<tuple<double, double, double, double, double, double, double, double>>
    results;

    while (dt > 0 ? t <= T : t >= T) {      // Checks whether dt is positive or
    negative (ternary conditional operator)
        // This loop contains a case distinction between the two methods

        // Add current positions and velocities to results vector
        // ...

        // Update positions and velocities based on chosen method (Euler or Verlet)
        // ...
    }

    return results;
}

void write_to_file(string filename, auto results) {
```

[9]The outline was mostly created using ChatGPT. An outline seems optional, potentially not affecting the grading.

```
    // Open output file, check for errors, write data, and close the file
    // ...
}

template <typename T>
double mean(const std::vector<T> &data) {
    return std::accumulate(data.begin(), data.end(), 0.0) / data.size();
}

template <typename T>
double standard_deviation(const std::vector<T> &data) {
    // Calculate standard deviation using mean() function
    // ...
}

int main() {
    double dt = 0.1;
    double T = 100;
    int N = 1000;
    vector<int64_t> elapsed_times_euler;
    vector<int64_t> elapsed_times_verlet;

    // a) Calculate positions, coarse and adequate step size, and write results to
   files
    // ...

    // b) Measure computation time for Euler and Verlet methods
    // ...

    // c) Reverse the simulation and write results to files
    // ...

    return 0;
}
```