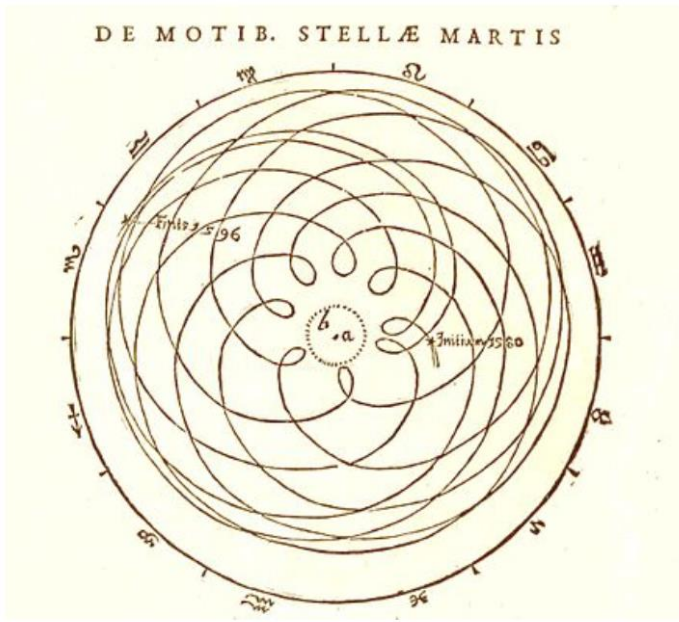
Recap: Week 1

- Errors: *floating point* and *truncation*.
- Simple projectile motion problem.
- Introduced *non-dimensionalisation*, a standard procedure for numerical methods.
- Introduced *Euler's method*, a simple forward difference approximation to the derivative.
  - $\mathbf{r}_{n+1} = \mathbf{r}_n + \tau \mathbf{v}_n\,, \qquad \mathbf{v}_{n+1} = \mathbf{v}_n + \tau \mathbf{a}_n.$
- *Tutorial* (`proj_euler.m`):
  - Evaluated how local and global errors scale with τ.
  - Euler's method worked ok!
  - Implemented(!) the *midpoint method*, which provides an exact numerical method for the constant acceleration case.
    - $\mathbf{v}_{n+1} = \mathbf{v}_n + \tau \mathbf{a}_n, \mathbf{r}_{n+1} = \mathbf{r}_n + \frac{1}{2}\tau\left(\mathbf{v}_n + \mathbf{v}_{n+1}\right)$
    - Requires the velocity update to be done before the position update.

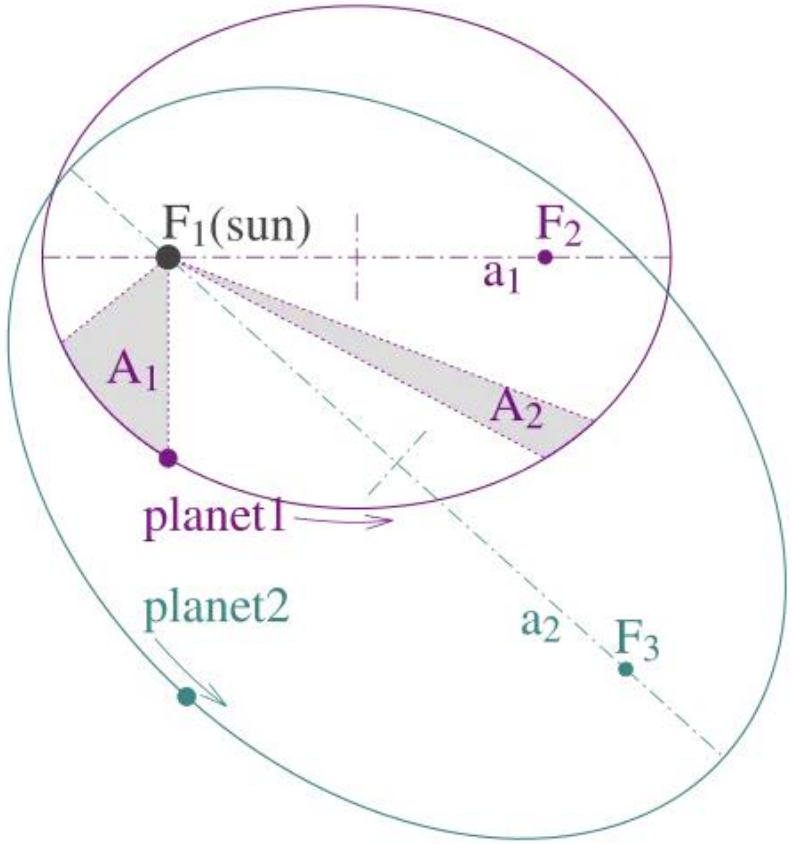
TODAY

- This frilly collar did some incredible early science:
  - Transformed the ancient tradition of physical cosmology by treating astronomy as part of physics (not mathematics as studied within arts).
  - His laws of planetary motion were foundational (elliptical orbits).
  - Check out some orbits around Earth.



Lecture 2: The Kepler problem

- Motion in a central gravitational force, e.g., planets/comets around Sun.
- *Non-dimensionalisation* simplifies the formulation.
- Euler's method (forward-difference) is unsatisfactory.
  - Introduce the *centered-difference approximation*.
- Yields a new method: *Verlet integration*.
  - Accurately integrates Keplerian orbits with near-conservation of total energy 😯
- *Code* kepler_verlet.ipynb, kepler_dynamics.ipynb



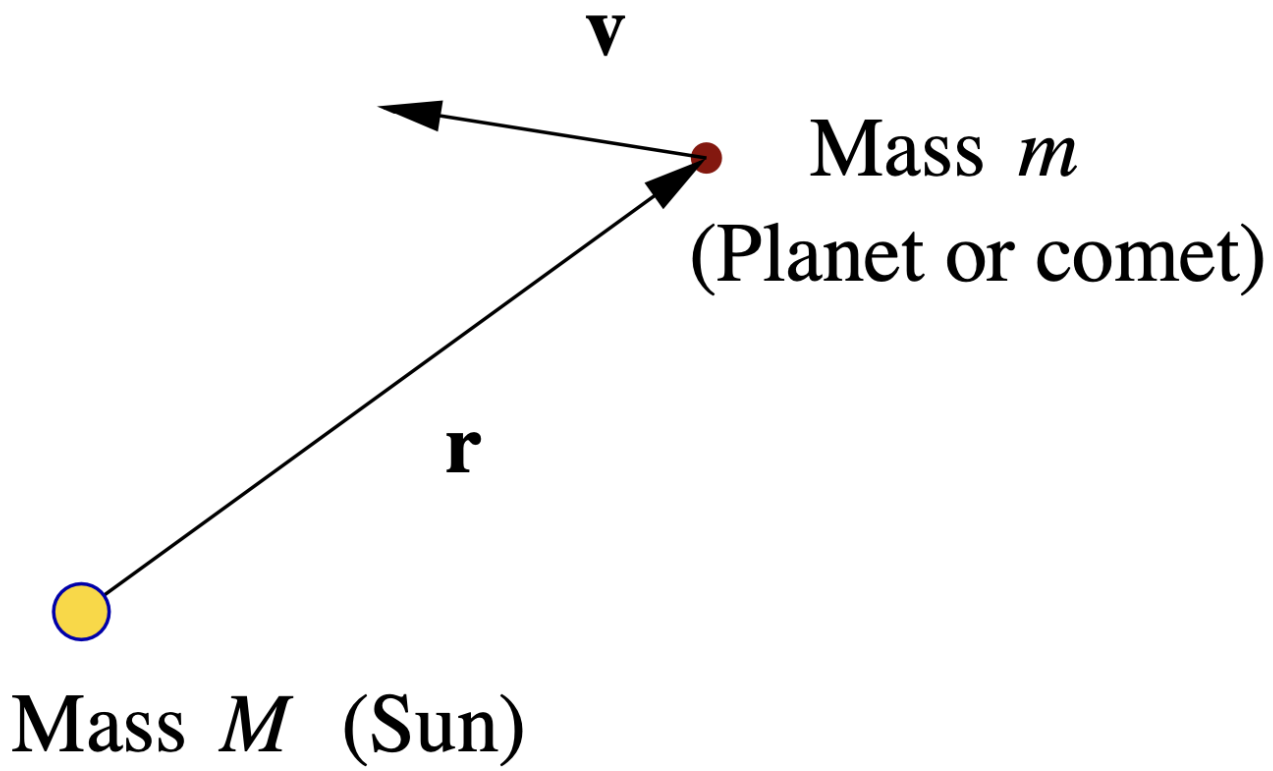
Recall: Dynamics problems

$$\frac{d\mathbf{r}}{dt} = \mathbf{v}\,, \qquad \frac{d\mathbf{v}}{dt} = \mathbf{a}(\mathbf{r}, \mathbf{v}, t)\,.$$

- Describe the motion of particle(s).
- Numerical methods (e.g., Euler) allow us to step forward in time (time step of size τ) and piece together an approximate trajectory:
  - $(\mathbf{r}_1, \mathbf{v}_1) \xrightarrow{\tau} (\mathbf{r}_2, \mathbf{v}_2) \xrightarrow{\tau} \cdots$
  - Local truncation error in each step.

The Kepler problem

- Dynamics of some object (e.g., a planet or comet) of mass m under the gravitational force of the Sun.
- Assume the Sun (mass $M \gg m$) is fixed at the origin.
- Paths followed are called *orbits*.
  - Stable orbits trace out ellipses with the Sun at a focus.



Gravitational acceleration and the Kepler problem

- Acceleration: $\mathbf{a}(\mathbf{r}) = -\frac{GM}{r^2}\hat{\mathbf{r}} = -\frac{GM}{|\mathbf{r}|^3}\mathbf{r}$
  - Depends only on position, $\mathbf{r}$, and points towards the origin (Newton, 1687).
  - $G \approx 6.67 \times 10^{-11}\,\mathrm{kg^{-1}m^3s^{-2}}$.
- So the *Keplerian equations of motion*:

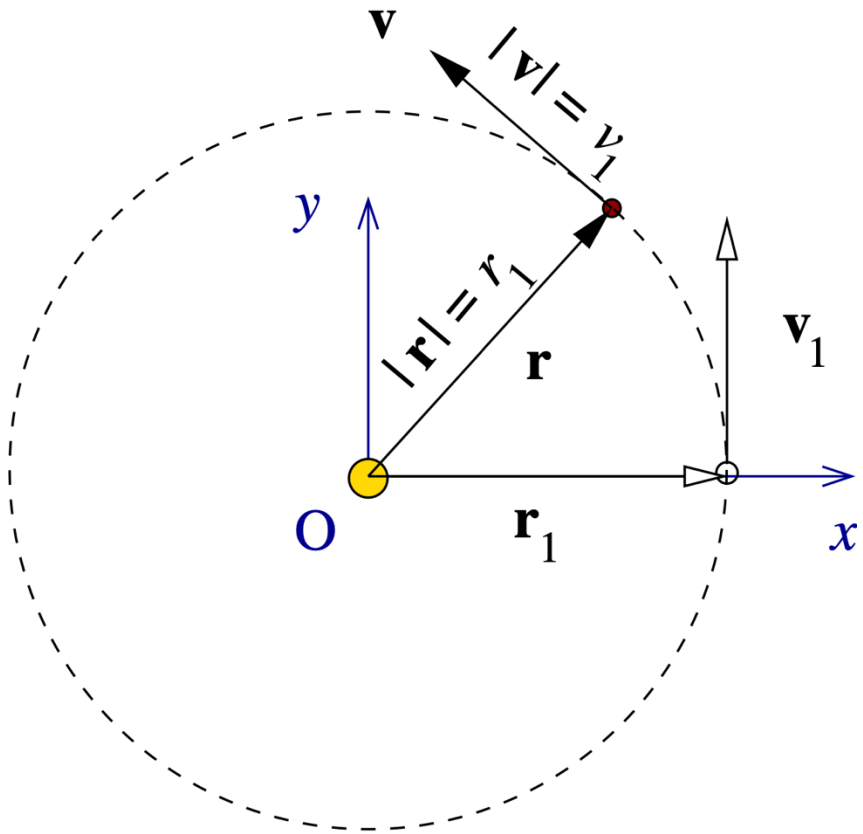
$$\boxed{\frac{d\mathbf{r}}{dt} = \mathbf{v}\,, \qquad \frac{d\mathbf{v}}{dt} = -\frac{GM}{r^2}\hat{\mathbf{r}}}\,.$$

  - Equivalent to second-order ODE $\frac{d^2\mathbf{r}}{dt^2} = -\frac{GM}{r^3}\mathbf{r}\,.$

- *Our goal:* solve these ODEs to numerically compute planetary orbits
  - Given some initial conditions, $\mathbf{r}(0) = \mathbf{r}_1$ and $\mathbf{v}(0) = \mathbf{v}_1$.
- We will *assess the accuracy* of numerical solutions against known quantities:
  1. Test the *conservation* of physical quantities that we know should be conserved.
     - *Angular momentum*: $\mathbf{L} = \mathbf{r} \times m\mathbf{v}$.
     - *Total energy*: $E = \frac{1}{2}mv^2 - \frac{GMm}{r}$.
     - [**Exercise**: Prove $\frac{d\mathbf{L}}{dt} = 0$ and $\frac{dE}{dt} = 0$].
  2. Compare the simulated trajectory against an *analytic solution*.

Circular Motion

$$F_{\text{grav}} = F_{\text{centripetal}} \quad \Rightarrow \quad \frac{GMm}{r^2} = \frac{mv^2}{r}$$



- We start at $\mathbf{r}(0) = (r_1, 0)$ and $\mathbf{v}(0) = (0, v_1)$.
  - $v_1 = \omega r_1 = \sqrt{\dfrac{GM}{r_1}}$.
- This is our circular motion test solution ⚪.

Non-dimensionalisation

- We *rescale variables* to form new dimensionless versions by dividing by dimensional constants $\bar{\mathbf{r}} = \dfrac{\mathbf{r}}{L_s}$, $\bar{t} = \dfrac{t}{t_s}$, $\bar{\mathbf{v}} = \dfrac{\mathbf{v}}{L_s/t_s}$.
- Rewriting the second-order equation of motion: $\dfrac{d^2\mathbf{r}}{dt^2} = -\dfrac{GM}{r^3}\mathbf{r} \quad \Rightarrow \quad \dfrac{d^2\bar{\mathbf{r}}}{d\bar{t}^2} = -\dfrac{GMt_s^2}{L_s^3}\dfrac{\bar{\mathbf{r}}}{\bar{r}^3}$
- ❓ What's a convenient choice for t_s?
  - 👀 $t_s = \sqrt{\dfrac{L_s^3}{GM}}$, which gives $\dfrac{d^2\bar{\mathbf{r}}}{d\bar{t}^2} = -\dfrac{\hat{\mathbf{r}}}{\bar{r}^2} = -\dfrac{\bar{\mathbf{r}}}{\bar{r}^3}$
- So we get $\boxed{\dfrac{d\bar{\mathbf{r}}}{d\bar{t}} = \bar{\mathbf{v}}\,, \qquad \dfrac{d\bar{\mathbf{v}}}{d\bar{t}} = -\dfrac{\bar{\mathbf{r}}}{\bar{r}^3}\,.}$
  - Thanks non-dimensionalisation: no more G and M to worry about

Conserved quantity: total energy

$$E = \tfrac{1}{2}mv^2 - \frac{GMm}{r}.$$

- We can also write down a non-dimensional form for the total energy, $\overline{E} = \dfrac{E}{E_s}$.
- A natural choice: $E_s = \dfrac{GMm}{L_s}$
- Yields $\overline{E} = \dfrac{1}{2}\overline{v}^2 - \dfrac{1}{\overline{r}}$.

***Summary*: Non-dimensional circular motion test**

- We have $\bar{v}_1 = \bar{r}_1^{-1/2}$, $\bar{\omega}_c = t_s \omega_c = \bar{r}_1^{-3/2}$, and $\bar{T}_c = 2\pi \bar{r}_1^{3/2}$.
- If we set the characteristic length scale for the problem, $L_s = r_1$ (the initial distance of the body from the origin), then…
  - Initial conditions: $\mathbf{r}(0) = (1, 0)$ and $\mathbf{v}(0) = (0, 1)$ yield *circular motion* with $r_c = 1$.
  - Orbits trace out the *unit circle* with period $T_c = 2\pi$.

Euler's method for the Kepler problem,

kepler_dynamics.ipynb

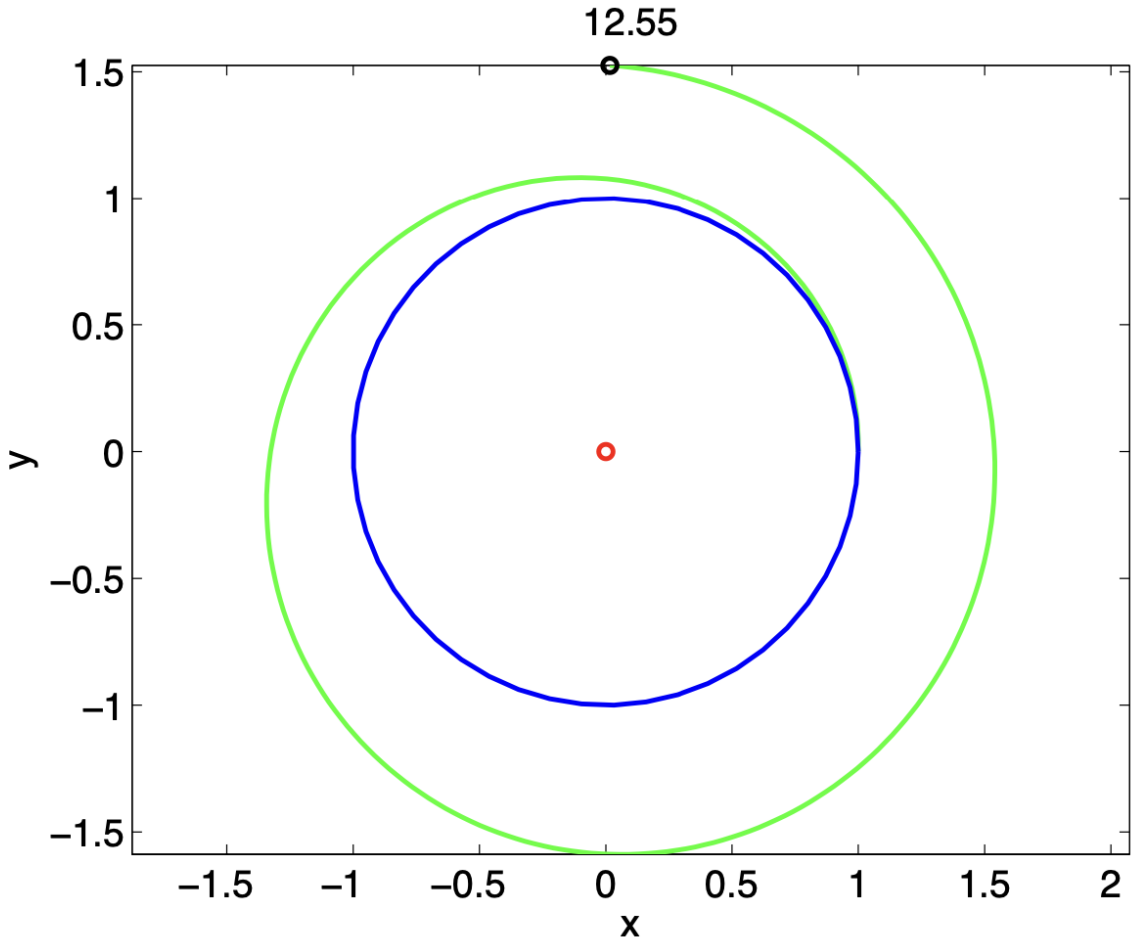
$$\mathbf{r}_{n+1} = \mathbf{r}_n + \tau \mathbf{v}_n \,,$$
$$\mathbf{v}_{n+1} = \mathbf{v}_n + \tau \mathbf{a}_n \,.$$

- Time steps forward discretely, as $t_n = (n-1)\tau$
- The Euler method (forward-difference approximation) worked ok for projectile motion...
  - What about for the Kepler problem?!

- Euler scheme for the Kepler problem is implemented in: kepler_dynamics.ipynb
  - Integrates for a total time 4π (should be two orbits).
  - Default (non-dimensional) time step is $\tau = 0.05$.
  - Code animates the orbit (using `drawnow`).
  - Also plots the analytic solution (unit circle).
  - Calculates non-dimensional $E(t)$ at each time step.
- Let's test it in the demo version, kepler_dynamics.ipynb

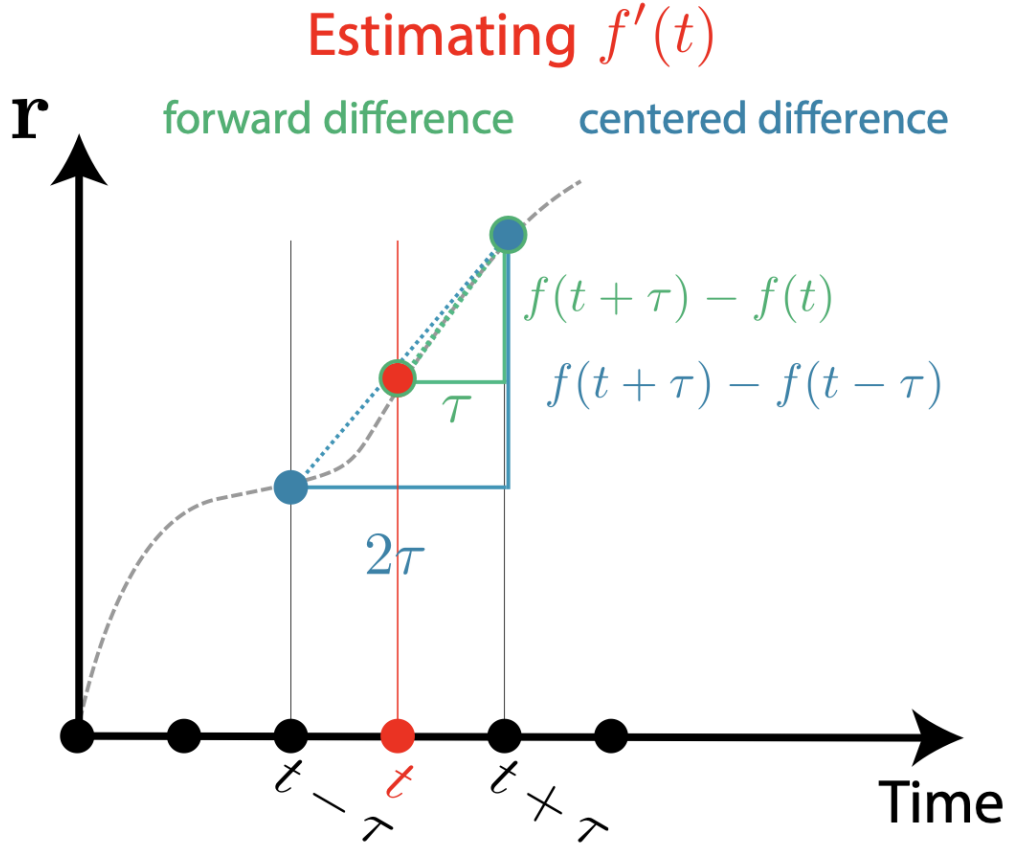
Euler on Kepler

- *It doesn't well at all*
  - The orbit spirals out and the total energy increases.
  - Smaller time steps, τ, only delay the inevitable.



Centered difference approximation

- Euler uses the *forward difference approximation*:
  - $f'(t) = \frac{f(t+\tau) - f(t)}{\tau} + O(\tau).$
- Consider an alternative: the *centered difference approximation*
  - $f'(t) = \frac{f(t+\tau) - f(t-\tau)}{2\tau} + O(\tau^2).$
- ❓ Does the higher-order truncation error–$O(\tau^2)$ instead of $O(\tau)$–mean more or less accurate?



***Derivation*: The centered difference approximation**

- Comes from manipulating the Taylor series:
  - $f(t \pm \tau) = f(t) \pm \tau f'(t) + \frac{1}{2!}\tau^2 f''(t) \pm \frac{1}{3!}\tau^3 f^{(3)}(t) + \ldots$
- Even powers of τ cancel when you compute $f(t+\tau) - f(t-\tau)$.
- We get $f'(t) = \frac{f(t+\tau) - f(t-\tau)}{2\tau} - \frac{1}{6}\tau^2 f^{(3)}(t) + \ldots$
  - Which we can write as $f'(t) = \frac{f(t+\tau) - f(t-\tau)}{2\tau} + O(\tau^2)$
- Truncation errors $O(\tau^2)$ are pretty good

Centered difference approximation: 2nd derivative

$$f''(t) = \frac{f(t+\tau) - 2f(t) + f(t-\tau)}{\tau^2} + O(\tau^2)\,.$$

- A workhorse in this unit!
  - (we use this alot)

A sketch of how to get it

- Decompose the second derivative as: $\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial x}\right)$.
- For the first derivative: let $g(x) = \frac{\partial f}{\partial x} \approx \frac{f(x) - f(x-h)}{h}$
- We can do a forward difference approximation to g as: $\frac{\partial g}{\partial x} \approx \frac{g(x+h) - g(x)}{h}$.
- Putting them together:
  $$\frac{\partial^2 f}{\partial x^2} \approx \frac{f(x+h) - f(x)}{h^2} - \frac{f(x) - f(x-h)}{h^2} = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$
- **Exercise**: Derive it fully from the Taylor-series expansion
  - By expanding about x at $f(x+h)$ and $f(x-h)$.

Updating using the Verlet method

$$\mathbf{r}_{n+1} = 2\mathbf{r}_n - \mathbf{r}_{n-1} + \tau^2 \mathbf{a}_n \,, \qquad \mathbf{v}_n = \frac{\mathbf{r}_{n+1} - \mathbf{r}_{n-1}}{2\tau} \,.$$

- *Notice*: You need to update $\mathbf{r}$ before you can update $\mathbf{v}$.
- *Getting started*: Consider the first step, $n = 1$ for $\mathbf{r}$. This requires $\mathbf{r}_{n-1} = \mathbf{r}(-\tau)$: a value *before the initial condition*?!
  - *Solution:* we can remove $\mathbf{r}_{n-1}$ from the Verlet method equations so that $\mathbf{r}_{n+1} = \mathbf{r}_n + \tau \mathbf{v}_n + \frac{1}{2}\tau^2 \mathbf{a}_n$.
  - (see also the *midpoint method* from Lab 1, Q2).
  - So we can get started (for $n = 1$) using this midpoint equation, and then apply the Verlet update equations for all remaining time steps, $n \geq 2$. 🏃🏃🏃

We can simulate $\mathbf{r}$ dynamics without computing $\mathbf{v}$

$$\mathbf{r}_{n+1} = 2\mathbf{r}_n - \mathbf{r}_{n-1} + \tau^2 \mathbf{a}_n \,, \qquad \mathbf{v}_n = \frac{\mathbf{r}_{n+1} - \mathbf{r}_{n-1}}{2\tau} \,.$$

- If $\mathbf{a} = \mathbf{a}(\mathbf{r})$, the Verlet update equations allow us to evolve $\mathbf{r}$ *without ever calculating* $\mathbf{v}$:
  - $\mathbf{r}_1 \overset{\tau}{\to} \mathbf{r}_2 \overset{\tau}{\to} \mathbf{r}_3 \overset{\tau}{\to} \cdots$.
- Advantageous if we only want to solve for $\mathbf{r}$

Verlet method: Truncation errors

$$\mathbf{r}_{n+1} = 2\mathbf{r}_n - \mathbf{r}_{n-1} + \tau^2 \mathbf{a}_n + O(\tau^4)\,, \qquad \mathbf{v}_n = \frac{\mathbf{r}_{n+1} - \mathbf{r}_{n-1}}{2\tau} + O(\tau^2)\,.$$

- The error terms in the Verlet method equations are deceptive.
- Successive updates are not independent (each update uses the current, $\mathbf{r}_n$ and previous, $\mathbf{r}_{n-1}$ values).
  - 🔥 So you need to write *two successive updates in the form of Taylor expansions* to correctly identify the local truncation errors.
- They are $O(\tau^3)$ for both position and velocity.
  - Global error as $O(\tau^2)$: a second-order method.
- Given its simplicity, this method is quite accurate!

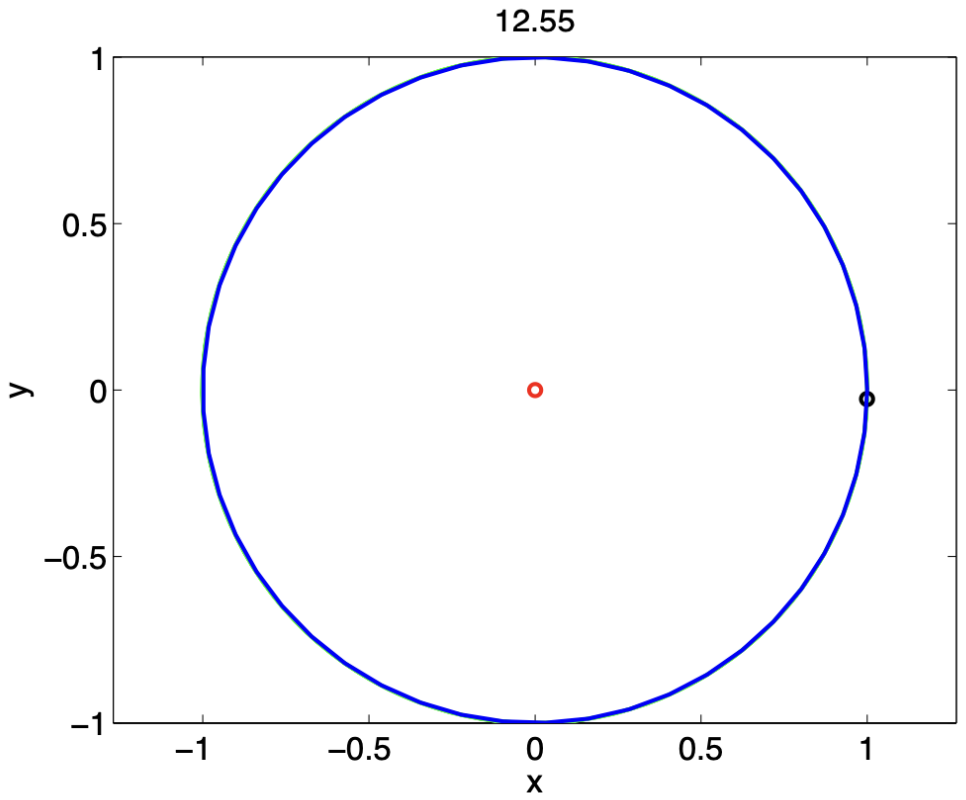
An alternative formulation: Velocity-Verlet

- Another (equivalent) way of writing the Verlet updates is called *Velocity-Verlet*:
- $$\begin{aligned}\mathbf{r}_{n+1} &= \mathbf{r}_n + \tau\mathbf{v}_n + \tfrac{1}{2}\tau^2\mathbf{a}_n\,, \\ \mathbf{v}_{n+1} &= \mathbf{v}_n + \tfrac{1}{2}\tau\left(\mathbf{a}_n + \mathbf{a}_{n+1}\right)\,.\end{aligned}$$
- The position update is the midpoint method (cf. Computer Lab 1, Q2).
- Requires velocity to be calculated (the $\tau\mathbf{v}_n$ term).
- **Exercise**: Derive this scheme.

Verlet solution to the Kepler problem:

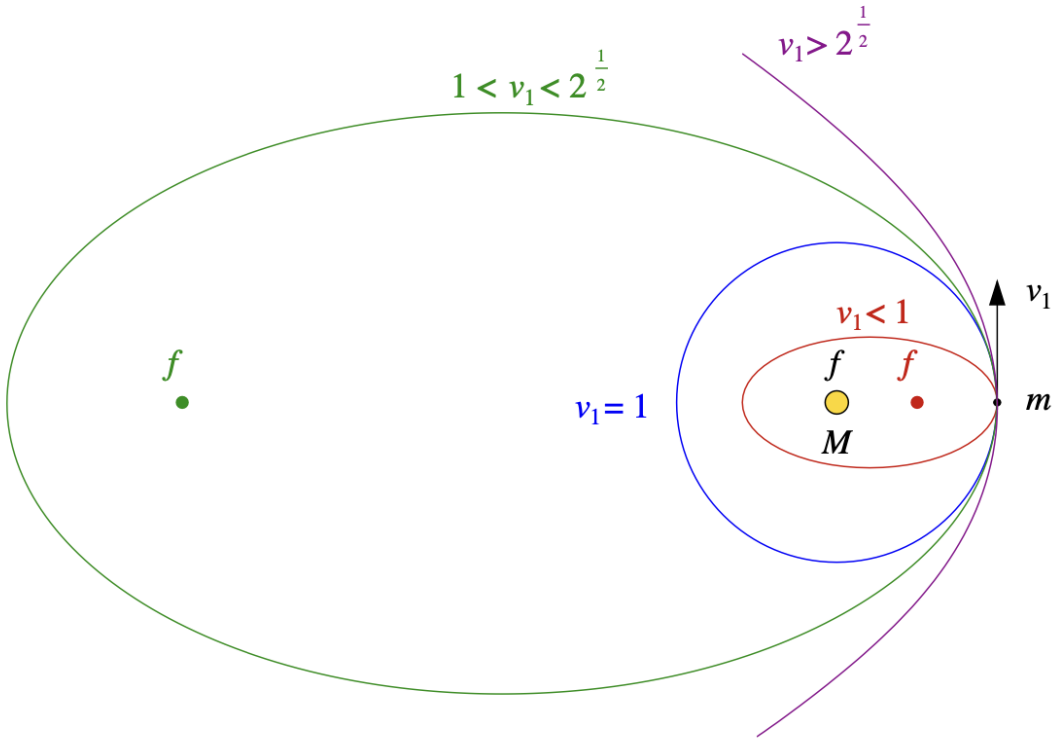
kepler_verlet.ipynb, kepler_dynamics.ipynb

- For $n = 1$ (midpoint method to get things running 🏃), use $\mathbf{r}_{n+1} = \mathbf{r}_n + \tau \mathbf{v}_n + \frac{1}{2}\tau^2 \mathbf{a}_n$.
- Then stardard Verlet updates for $n > 1$:
  $\mathbf{r}_{n+1} = 2\mathbf{r}_n - \mathbf{r}_{n-1} + \tau^2 \mathbf{a}_n$, $\mathbf{v}_n = \frac{\mathbf{r}_{n+1} - \mathbf{r}_{n-1}}{2\tau}$.
- Let's demo it in kepler_verlet.ipynb
  💻
- Accurately integrates the circular test case using $\tau = 0.05$!
- Energy is (nearly) conserved
  - (oscillates $\sim 10^{-7}$)!! 🆒🆒🆒

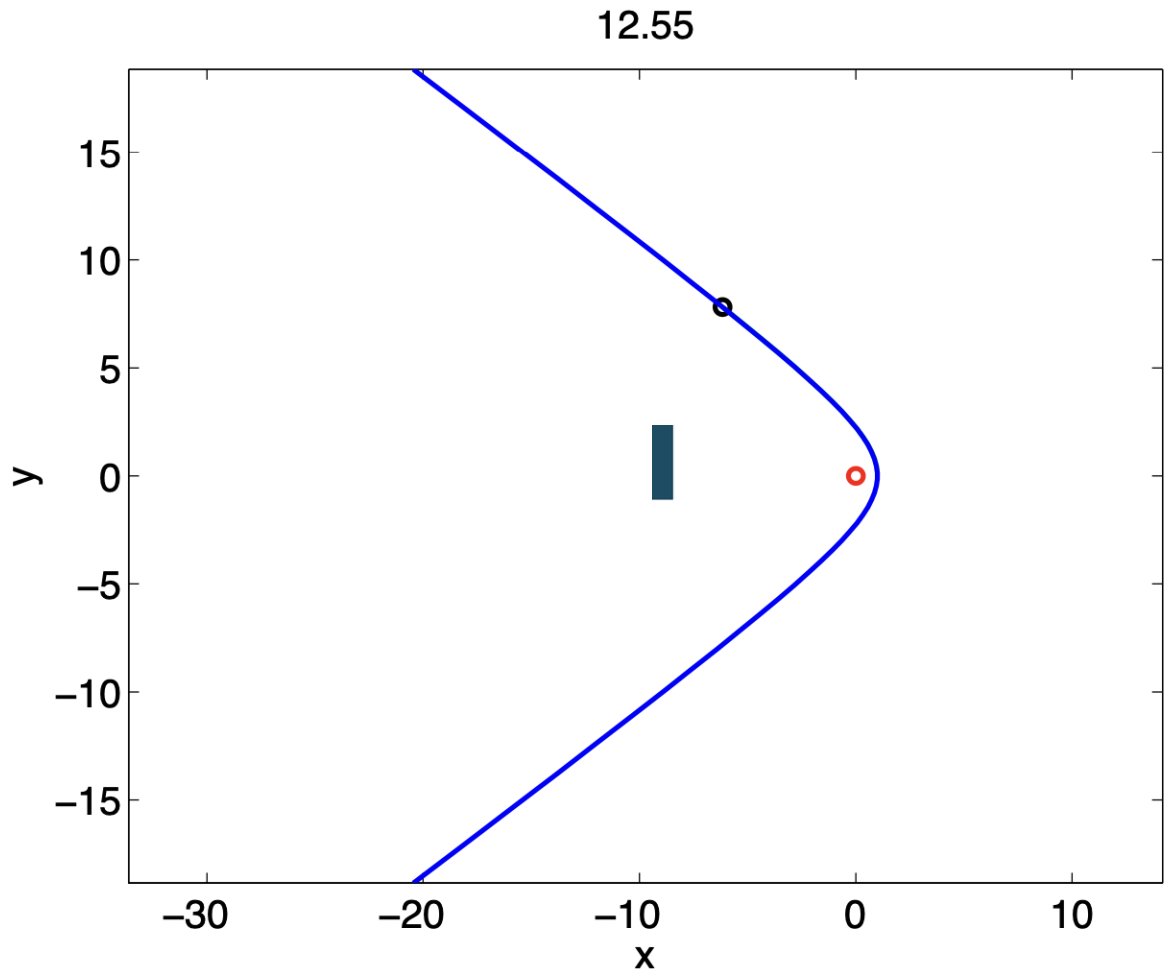


Solutions to Kepler problem: ellipses and hyperbolae

For initial conditions $\mathbf{r}_1 = (1, 0)$ and $\mathbf{v}_1 = (0, v_1)$, we can get different types of solutions depending on v_1:



Example: numerical solution for $v_1 = 1.5$; a hyperbolic orbit.



Key Concepts

The standard Euler method updates the position by considering only the linear dependence on time, effectively assuming constant velocity during each time step. However, in systems experiencing acceleration, the position update should account for the quadratic dependence on time due to the acceleration term.

Euler Method Position Update:

In the explicit Euler method, the position is updated as:

$$\mathbf{r}_{n+1} = \mathbf{r}_n + \mathbf{v}_n \Delta t$$

This formula assumes that the velocity (\mathbf{v}_n) remains constant over the time step (\Delta t), neglecting any changes due to acceleration.

Accurate Position Update with Acceleration:

For a more accurate update, especially in the presence of acceleration, the position should be updated using the kinematic equation that includes the acceleration term:

$$\mathbf{r}_{n+1} = \mathbf{r}_n + \mathbf{v}_n \Delta t + \frac{1}{2}\mathbf{a}_n \Delta t^2$$

Here, $\mathbf{a}_n$ is the acceleration at time step n. This equation accounts for the fact that the velocity is changing due to acceleration, leading to a more accurate position update.

Alternative Methods:

Numerical integration methods like the Verlet integrator incorporate this quadratic term, providing more accurate results for systems with acceleration. The Verlet method updates the position as:

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

This approach inherently considers the effects of acceleration, leading to better energy conservation and trajectory accuracy over time.

In summary, while the Euler method offers simplicity, it lacks the precision required for systems where acceleration plays a significant role. Incorporating the acceleration term in the position update, as done in methods like the Verlet integrator, yields more accurate and reliable results.

Appendix: Analytic solutions to the Kepler problem

- Nice to compare the performance of numerical methods to analytic solution:

  kepler_dynamics.ipynb

- *Cartesian* form of (elliptic) solution for $\mathbf{r}_1 = (1, 0)$ (closest point of approach) and $\mathbf{v}_1 = (0, v_1)$:
  - $\frac{(x + ae)^2}{a^2} + \frac{y^2}{b^2} = 1$
    - eccentricity $e = v_1^2 - 1$
    - semi-major axis $a = \frac{1}{1 - e}$
    - semi-minor axis $b^2 = a^2(1 - e^2)$.

Parametric Elliptic and Hyperbolic Solutions (parameterized by θ)

- *Elliptic solutions*
  - have $e < 1$ or $v_1 < \sqrt{2}$:
  - $$x = -ae + a\cos\theta\,,$$
    $$y = b'\sin\theta \qquad (0 \le \theta \le 2\pi)\,.$$
- Semi-major axis a
- Semi-minor axis $b' = a\sqrt{1-e^2}$.
- Focii $x_\pm = -ae \pm \sqrt{a^2 - b^2}$, so $x_\pm = 0, -2ae$
  - (the Sun is at $x_+ = 0$).

- *Hyperbolic solutions*
  - have $e > 1$ or $v_1 > \sqrt{2}$:
  - $$x = -ae + a\cosh\theta$$
    $$y = b'\sinh\theta \qquad (-\infty < \theta < \infty)\,,$$
- Semi-minor axis $b' = a\sqrt{e^2-1}$.

***Appendix*: Challenge 🔥 (e.g., for 3934 Students)**

- **Exercise**: Think about how you might derive the global error in position ($O(\tau^2)$) for the Verlet Method (which has a local truncation error in position, $O(\tau^4)$).
  - *Hint*: Consider errors at r_{n+1}, r_{n+2}, r_{n+3}, ... and use induction.