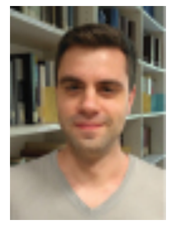
A primer to numerical simulations:
The perihelion motion of Mercury

C. Körber | I. Hammer | J.-L. Wynen | J. Heuer | C. Müller | C. Hanhart






[arXiv:1803.01678] Physics Education Physics Today

quarks Q2C cosmos APRIL MEETING 2019
April 13-16, 2019 Denver, Colorado
Session L13.00002 | April 14, 2019

Christopher Körber | UC Berkeley & LBNL | Feodor Lynen Fellow

ckoerber@berkeley.edu | www.ckoerber.com

JÜLICH FORSCHUNGSZENTRUM

Berkeley Lab Lawrence Berkeley National Laboratory

Unterstützt von / Supported by Alexander von Humboldt Stiftung / Foundation

Berkeley UNIVERSITY OF CALIFORNIA

A **one-day course** enables students to **program simulations** that demonstrate the effects of **general relativity**

Course is hands-on projects option of (particle) *physics summer school for students* (16-19y, Germany)

Content of Course

- Perihelion motion of Mercury caused by **G**eneral **R**elativity
- Numerical simulation of physics system

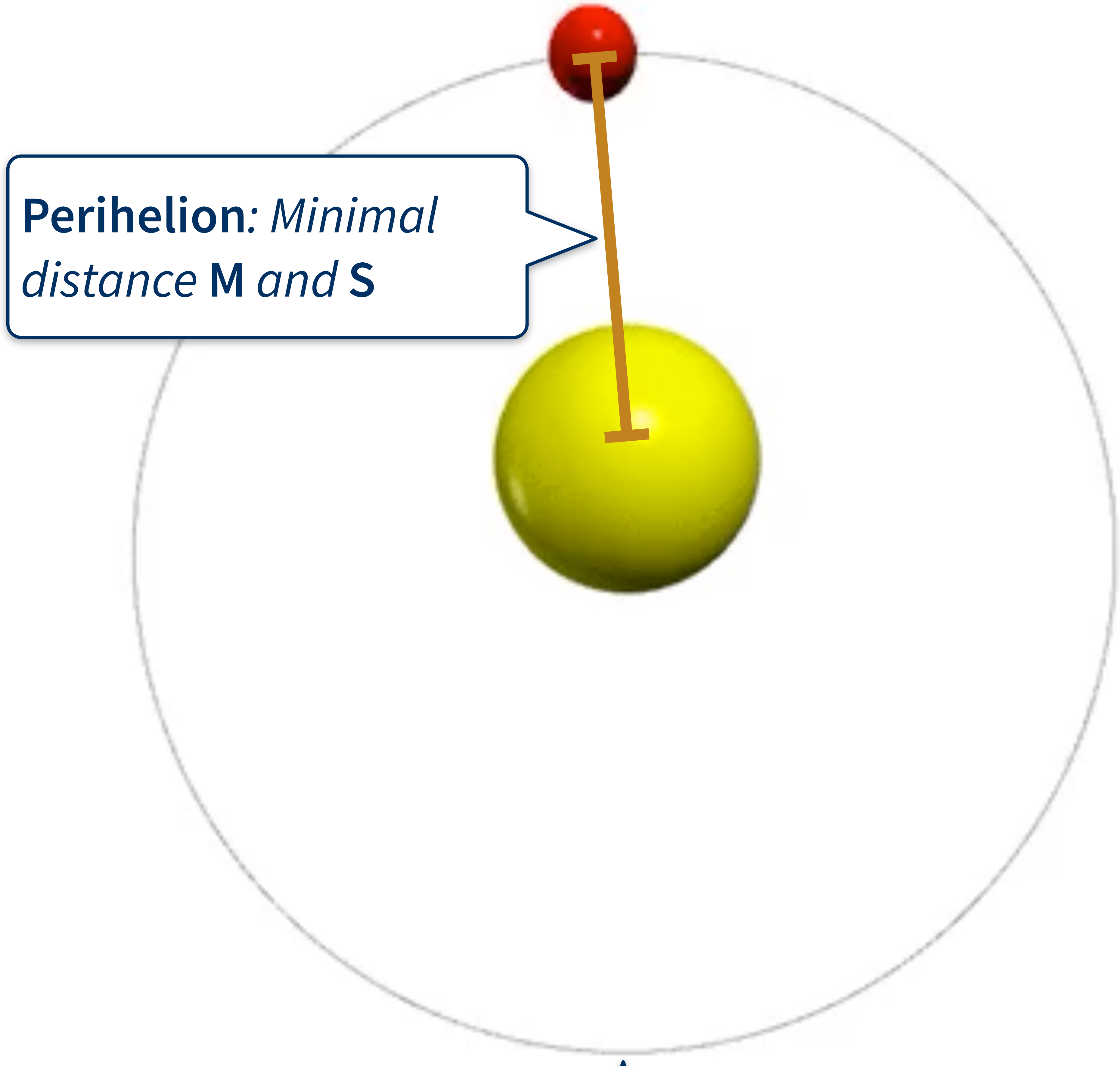
Objectives of Course

Students …

- access *advanced* physics in *early* stage
- have (first) contact with numerical simulations
- develop intuition for physical systems

Physics Motivation

"Classical" Motion of Mercury

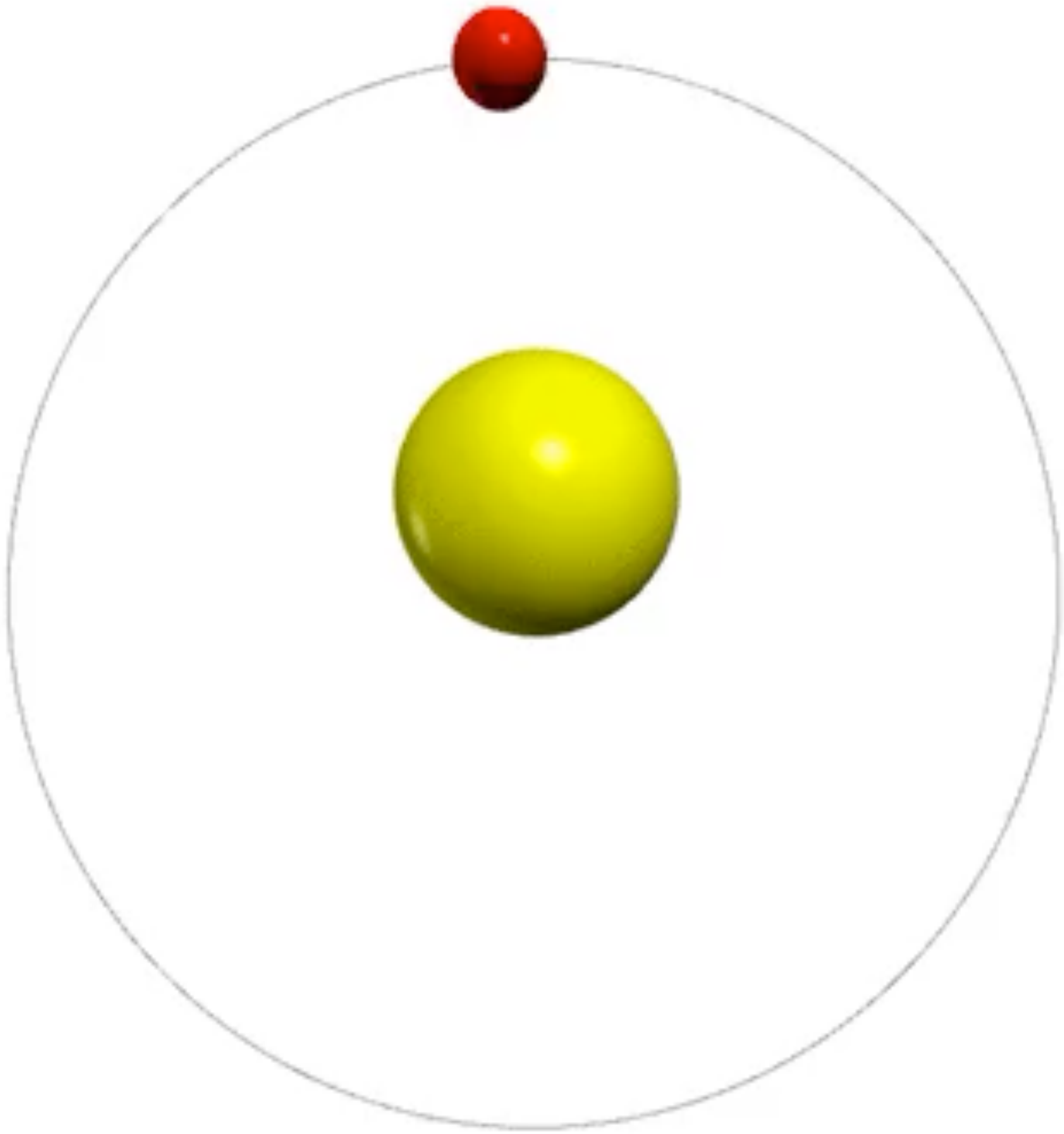


This animation is actual result of course

Physics Motivation

$$\langle r \rangle \sim 46 \cdot 10^6\text{km} \sim 29 \cdot 10^6\text{mi}$$
$$r_S \sim 3\text{km} \sim 1.9\text{mi}$$

"Classical" Motion of Mercury



Newtonian Motion of Mercury

- Stationary orbit*
- Force

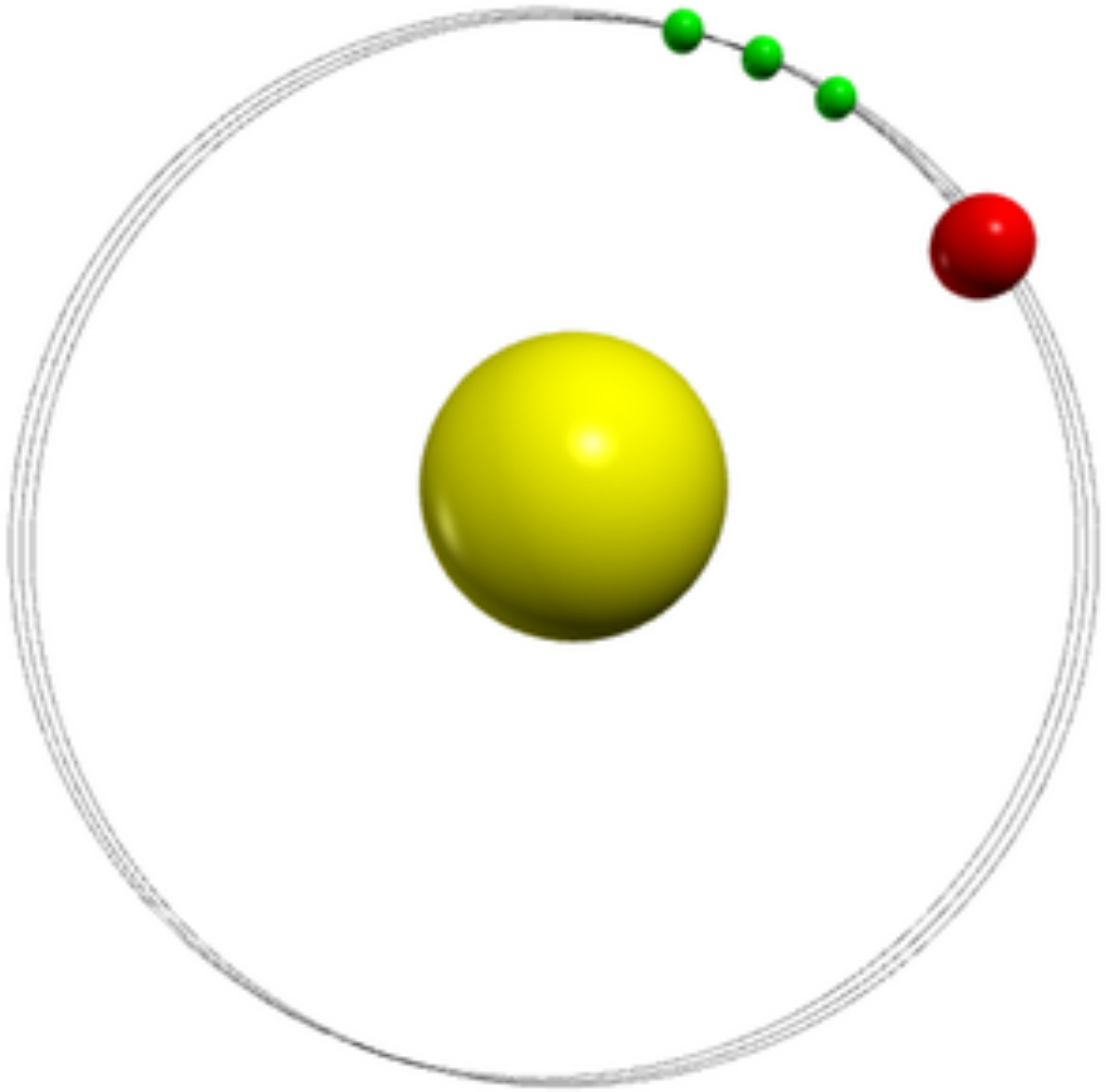
$$\vec{F}_0 = -\frac{mc^2}{2}\frac{r_S}{r^2}\frac{\vec{r}}{r}$$

*Effects of other planets also cause perihelion motion on top of GR.

Physics Motivation

$$\langle r \rangle \sim 46 \cdot 10^6\,\text{km} \sim 29 \cdot 10^6\,\text{mi}$$
$$r_S \sim 3\,\text{km} \sim 1.9\,\text{mi}$$

Observed Motion of Mercury



For visualization: $\alpha, \beta \gg 1$

Newtonian Motion of Mercury

$$\vec{F}_0 = -\frac{mc^2}{2}\frac{r_S}{r^2}\frac{\vec{r}}{r}$$

Motion of Mercury including GR

- Force (include leading extensions)

$$\vec{F}_{\text{GR}} = \vec{F}_0\left(1 + \alpha\frac{r_S}{r} + \beta\frac{r_L^2}{r^2}\right)$$

$$r_L^2/\langle r \rangle^2 \sim 4 \cdot 10^{-8} \quad \text{GR}: \ \alpha = 0\,, \quad \beta = 3$$

J.D. Wells, Effective Theories in Physics (2012), [arxiv:1106.1568]

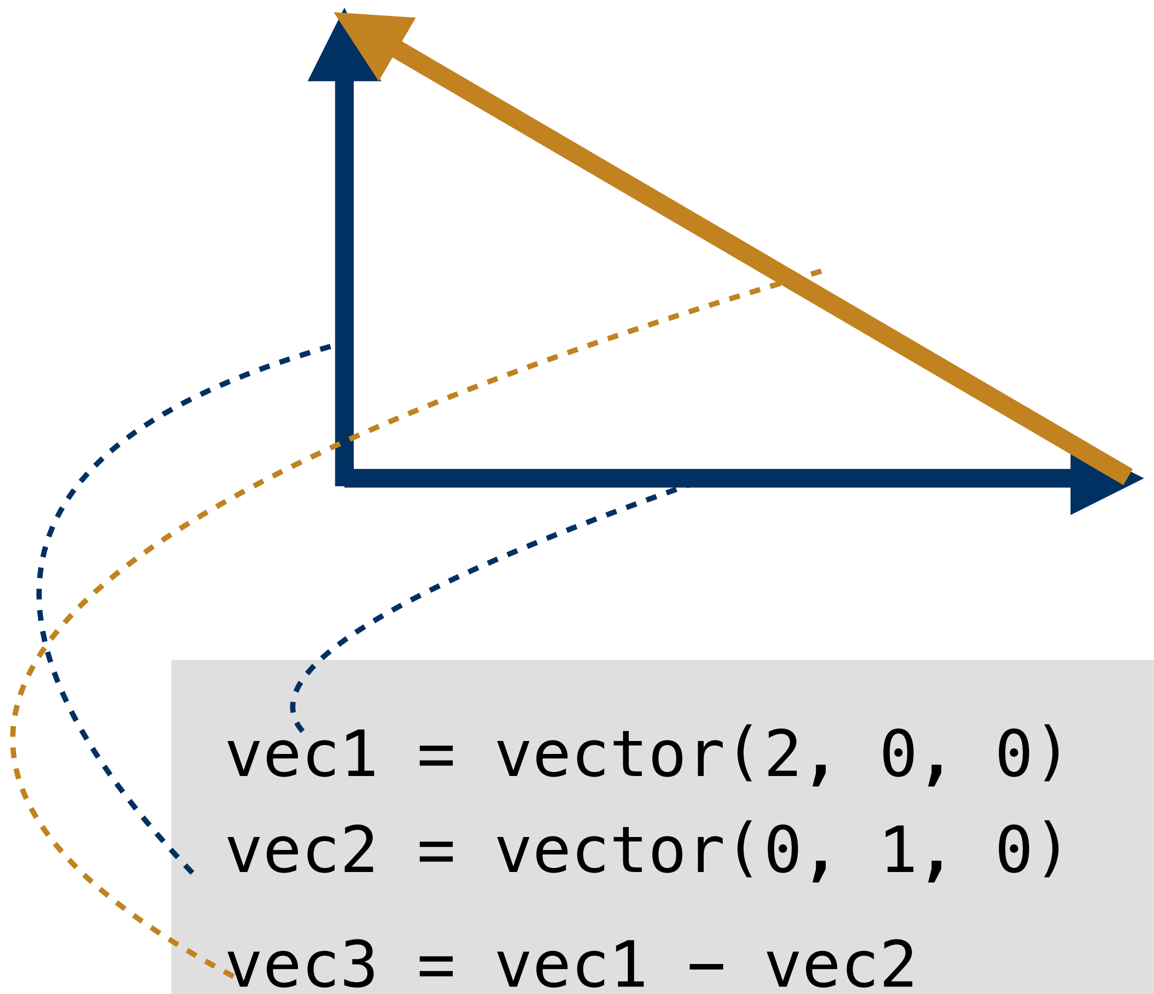
Methods & Software

Programming language: [Python][1] (Module: [VPython][2])

- Open Source (free to use on any platform)
- *Relatively* easy to install (guide[3]), or can be run online without any install[4]
- *Relatively* easy to use: dynamic typing, extensive documentation & many examples
- Is commonly used in science & industry

Important concepts

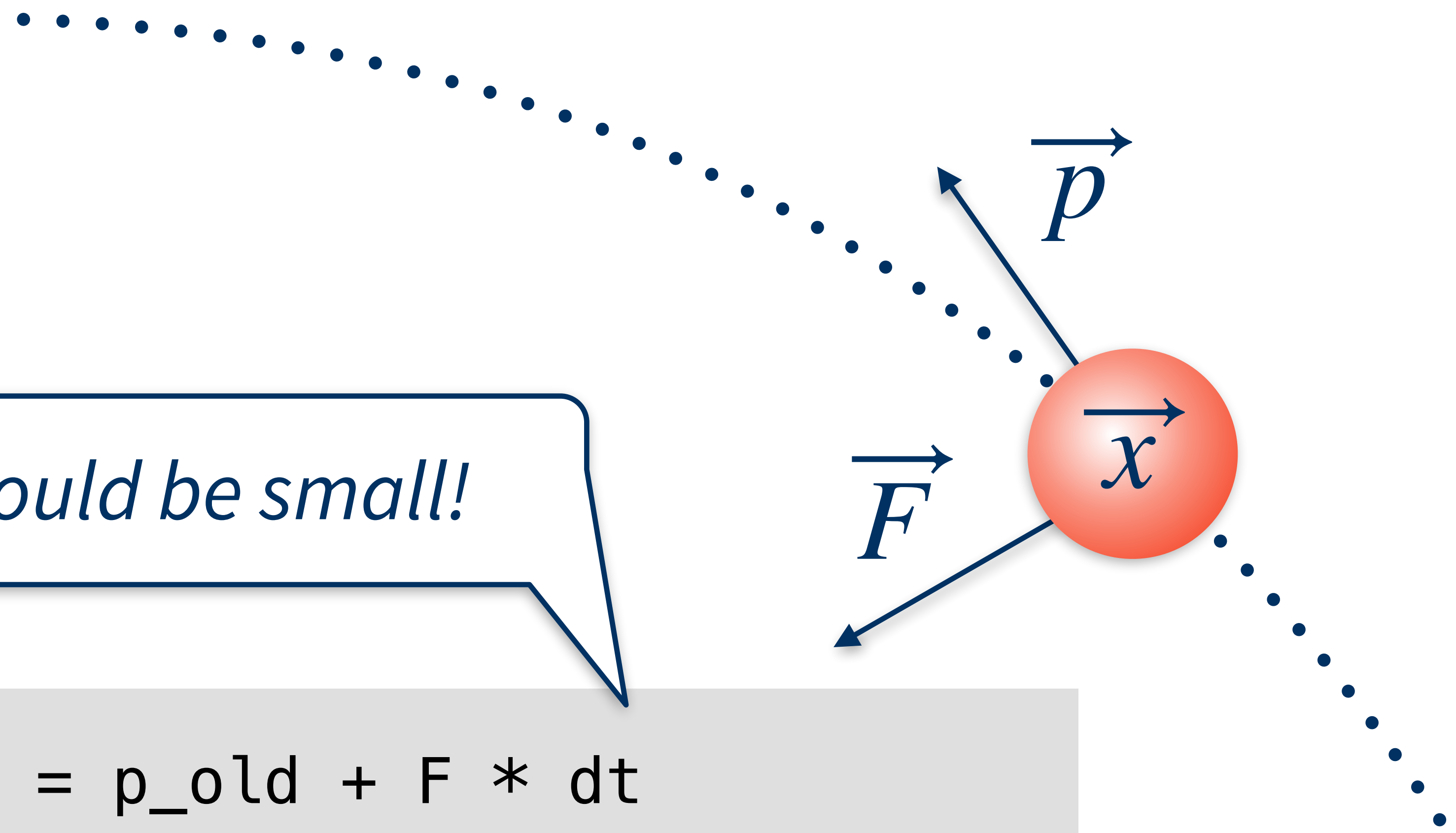
- Vector additions



```
vec1 = vector(2, 0, 0)
vec2 = vector(0, 1, 0)
vec3 = vec1 − vec2
```

- Change of momenta

$$\dot{\vec{p}} = \vec{F}$$



`dt` *should be small!*

```
p_new = p_old + F * dt
x_new = x_old + p_new / m * dt
```

[1] www.python.org
[2] www.vpython.org
[3] github.com/ckoerber/perihelion-mercury
[4] www.glowscript.org

(Almost) Source Code

```
from vpython import *

# define parameters: rS, rL2, c_a, dt, …
# set up graphics objects & initial vectors: x0, v0, …
M = sphere(pos=x0)
M.velocity = v0

def evolve_mercury(x, v, alpha, beta):
   acc_mag = c_a / x.mag**2
   acc_vec = -x / x.mag * acc_mag

   v_new = v + acc_vec * dt
   x_new = x + v_new * dt

   return x_new, v_new

while True:
   M.pos, M.velocity = evolve_mercury(M.pos, M.velocity, alpha, beta)
```

(Almost) Source Code

```python
from vpython import *

# define parameters: rS, rL2, c_a, dt, …
# set up graphics objects & initial vectors: x0, v0, …
M = sphere(pos=x0)
M.velocity = v0

def evolve_mercury(x, v, alpha, beta):
   acc_mag = c_a / x.mag**2 * (1 + alpha * rS / x.mag + beta * rL2 / x.mag**2)
   acc_vec = -x / x.mag * acc_mag

   v_new = v + acc_vec * dt
   x_new = x + v_new * dt

   return x_new, v_new

while True:
   M.pos, M.velocity = evolve_mercury(M.pos, M.velocity, alpha, beta)
```

Template Code (Example 2)

```
from vpython import *

# define parameters: rS, rL2, c_a, dt, …
# set up graphics objects & initial vectors: x0, v0, …
M = sphere(pos=x0)
M.velocity = v0

def evolve_mercury(x, v, alpha, beta):
  # compute the acceleration based on alpha, beta and x


  # update the velocity vector v to obtain v_new
  # update the position vector x to obtain x_new

  return x_new, v_new

while True:
  M.pos, M.velocity = evolve_mercury(M.pos, M.velocity, alpha, beta)
```

Template Code (Example 2)

```
from vpython import *

# define parameters: rS, rL2, c_a, dt, …
# set up graphics objects & initial vectors: x0, v0, …
M = sphere(pos=x0)
M.velocity = v0

def evolve_mercury(x, v, alpha, beta):
  # compute the acceleration based on alpha, beta and x


  # update the velocity vector v to obtain v_new
  # update the position vector x to obtain x_new

  return x_new, v_new

while True:
  # Use evolve_mercury to update the position & velocity of M
```

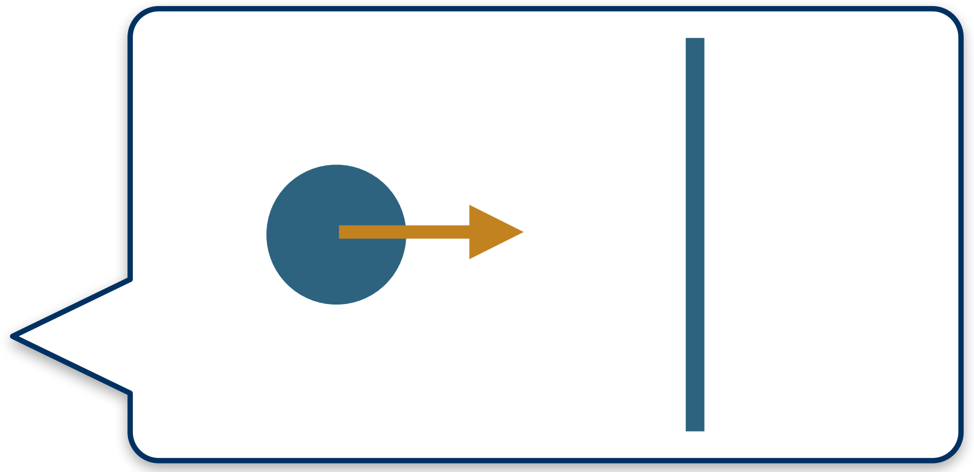
Course Details

Framework

- **Group size:** ~15 students work independently (groups of 2-3), 2 tutors for Q&A
- **Workload:** 2h + 4h session, intermediate breaks
- **Closing:** 10min student presentation (to other summer school courses)

Schedule

- **Initial lecture:** ~ 30min
  - Mercurys orbit: what is observed
  - How to describe this: vector additions & Newton's laws
- **Familiarizing with VPython:** ~ 1.5h, "ball in the box" example
  - Step-by-step instruction notes[1]



- **Simulation of orbit:** Rest of course
  - Tutors describe form of gravitational force from GR
  - Provide template program[2]
  - Students experiment with system (changing masses, initial velocities, …)

[1] www.glowscript.org/docs/VPythonDocs/VPython_Intro.pdf [2] github.com/ckoerber/perihelion-mercury

Observations

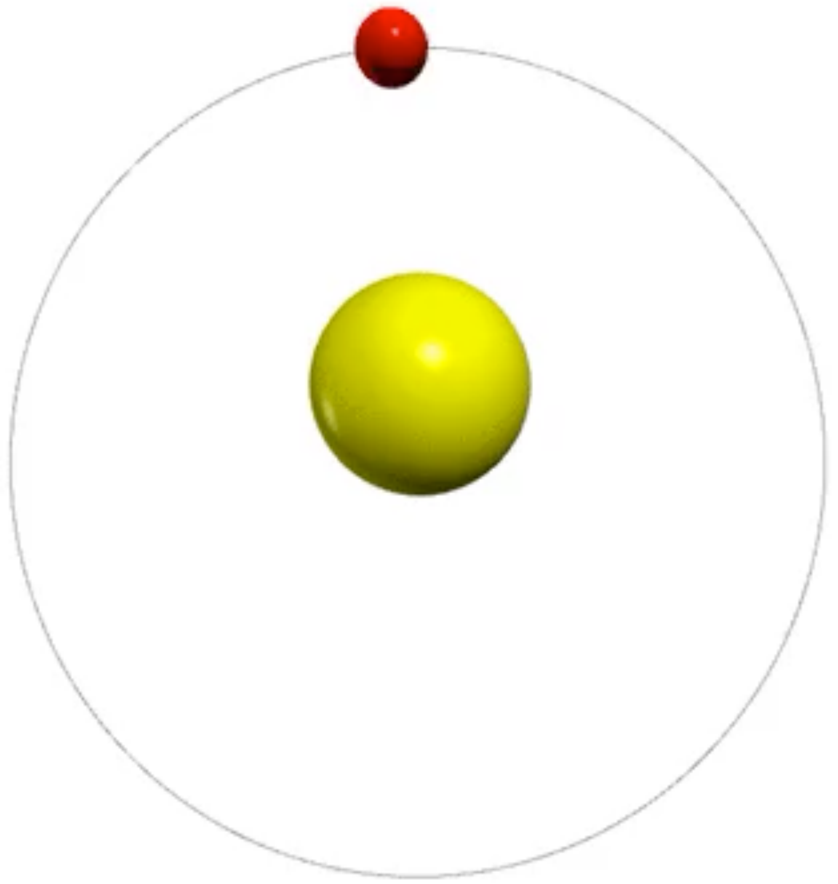
Different learning pace

- TAs actively support slower groups
- Advanced tasks for advanced groups
  - Change initial parameters (mass, location, speed, …)
  - Add additional planets
  - Investigate numerical artifacts
- Advanced students supported groups with less progress

Interest of students

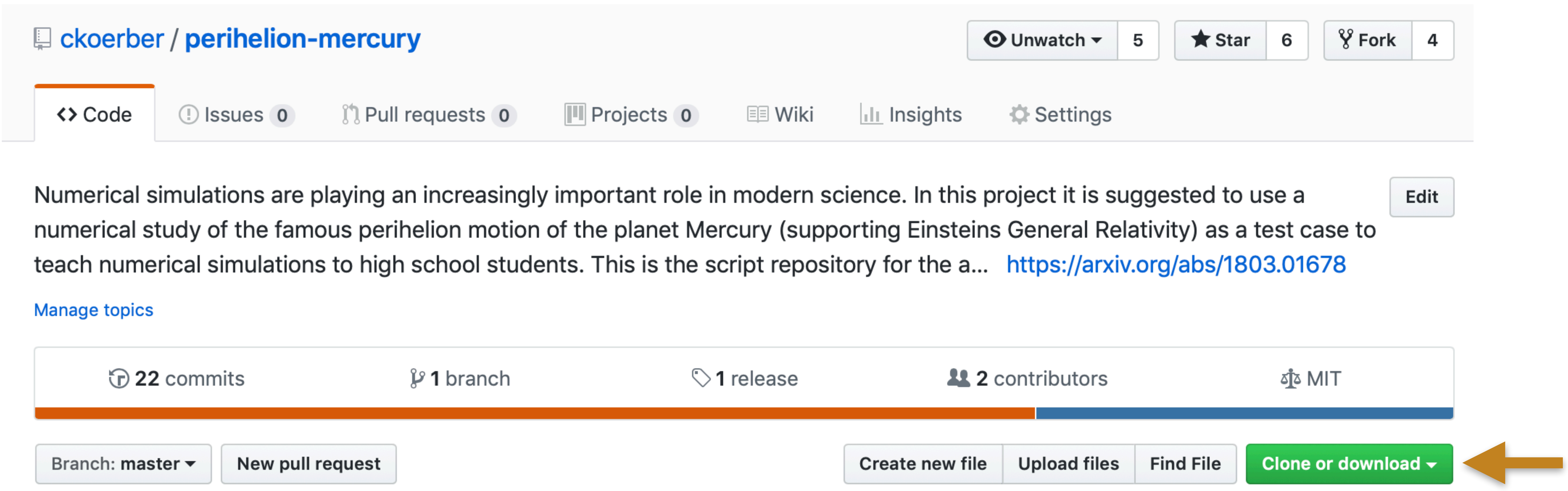
- Sharing of results with other courses
- Students introduced new ideas
  - Visualization of perihelion
  - Criteria for measuring perihelion location
  - Planets in different planes

Sources | Access



Installation guide, working examples & templates

- github.com/ckoerber/perihelion-mercury



Description of course

[arXiv:1803.01678]　　Phys. Ed. vol53.5 p055007 (2018)　　Physics Today

Contact

- ckoerber@berkeley.edu