Summary of course

Morten Hjorth-Jensen Email morten.hjorth-jensen@fys.uio.no

epartment of Physics and Astronomy and Facility for Rare Ion Beams and National Superconducting Cyclotron Laboratory, Michigan State Universit

Apr 24, 2020

What? Me worry? No final traditional exam in this course!

What did I learn in school this year?

Does this figure that match the experiences you have made this semester?

©Zits Partnership

Topics we have covered this year

- How to derive equations of motion based on forces
- Inertial frames and their relation to accelerating and rotating frames (non-intertial frames)
- Forces, work, energy, angular momentum, linear momentum and conservation laws
- Various types of motions, falling objects, objects moving in various fields
- Analysing energy diagrams and defining effective potential
- Small oscillations, Harmonic oscillator potential and equations of motion
- Transformation of variables that allow for analytical solutions, example two-body problems
- Central forces and two-body problems, center-of-mass and relative coordinates as reference frame
- Two-body scattering problems, classical scattering cross section
- Variational calculus and Lagrangian formalism
- Deriving equations of motion from Lagrangian formalism only
- Lagrangian formalism with constraints

And how to solve these problems

We have studied many systems numerically, from falling objects with and without friction/air resistance, small oscillations (harmonic oscillator), gravitational problems and other central force problems, rotations and the classical pendulum.

- Euler-Cromer and Velocity-Verlet as energy conserving algorithms (time-independent forces)
- Runge-Kutta family of algorithms for time-dependent forces
- Numerical integration using the Trapezoidal, midpoint and Simpson's rule.

One program to rule them all?

```
DeltaT = 0.001
#set up arrays
tfinal = 10 # in years
n = ceil(tfinal/DeltaT)
# set up arrays for t, a, v, and x
t = np.zeros(n)
v = np.zeros((n,3))
r = np.zeros((n,3))
# Initial conditions as compact 3-dimensional arrays
r0 = np.array([1.0,0.0,0.0])
v0 = np.array([0.0,2*pi,0.0])
r[0] = r0
v[0] = v0
# Start integrating using Euler's method
for i in range(n-1):
    # Set up the acceleration for force that depends only on position
    a =  Force(r)/mass
    # update velocity, time and position using the Euler-Cromer forward
    v[i+1] = v[i] + DeltaT*a
    r[i+1] = r[i] + DeltaT*v[i]
    t[i+1] = t[i] + DeltaT
```

The overarching idea is that as soon as we have analyzed the forces at play, we can reuse our programs (algorithms) with new forces, without having to recur to the few analytical cases we can solve.

Best wishes to you all and thanks so much for your heroic efforts this semester