Dynamics

PHYS 246 class 2

https://lkwagner.github.io/IntroductionToComputationalPhysics/intro.html

```
from google.colab import drive
drive.mount('/content/drive')
!cp /content/drive/MyDrive/Colab\ Notebooks/Dynamics.ipynb ./
!jupyter nbconvert --to HTML "Dynamics.ipynb"
```

Announcements/notes

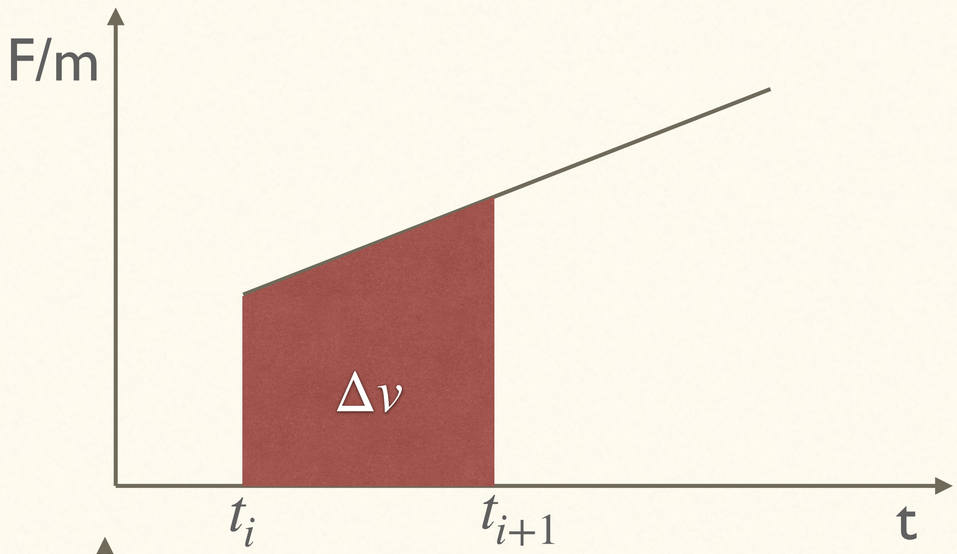
- First assignment is due tonight on canvas

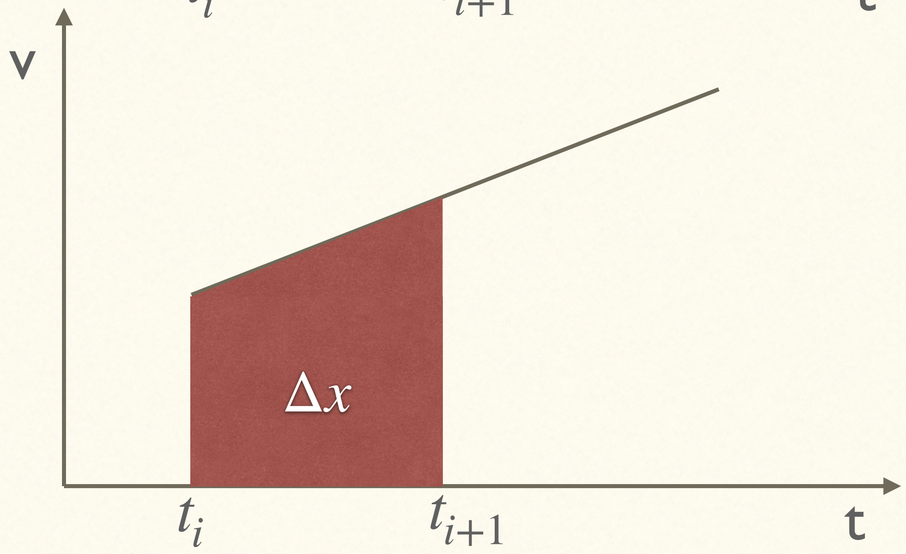
- Check new PDF generation guide on the website
(if you already submitted and the PDF is clear and legible, you are ok)

- Note that there is not much partial credit. Grade will be based on number of correct entries. A few correct things are better than a lot of incorrect things.

- Related to above, make sure that the first part of your assignment is correct before moving on -- the end depends on the beginning.

Dynamics by integration





Suppose we know the position and velocity at some time t_i, and want to know these quantities at a future time t_{i+1}

Newton says:

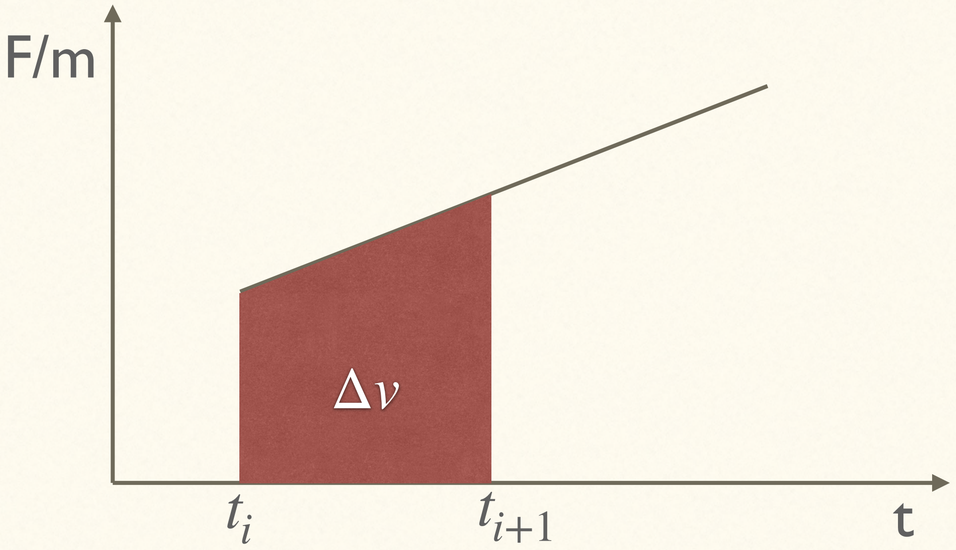
$$F = m\frac{dv}{dt}$$

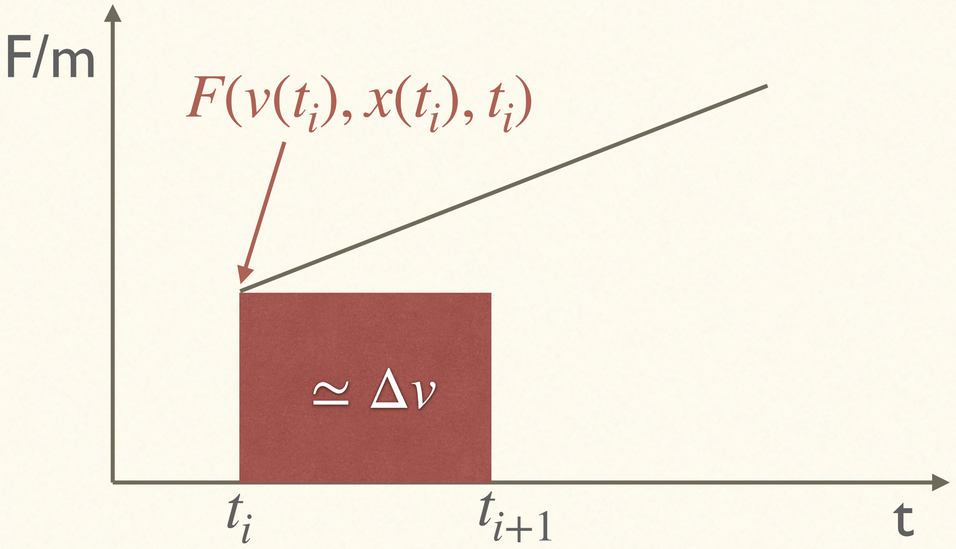
So

$$v(t_{i+1}) - v(t_i) = \int_{t_i}^{t_{i+1}} \frac{F}{m} dt$$

If we can compute this integral we know what the velocity will be. Similar for position.

Euler integration





W know the position and velocity at some time t_i, and want to know these quantities at a future time t_{i+1}

If we can compute $F(v, x, t)$, then we can approximate this integral.

Error is proportional to Δt:

$$F = F(t_i) + \frac{dF(t_i)}{dt}\Delta t + \frac{1}{2}\frac{d^2F(t_i)}{dt^2}(\Delta t)^2 + \ldots$$

Euler algorithm

Start with $x(t_0), v(t_0)$ (may be vectors)

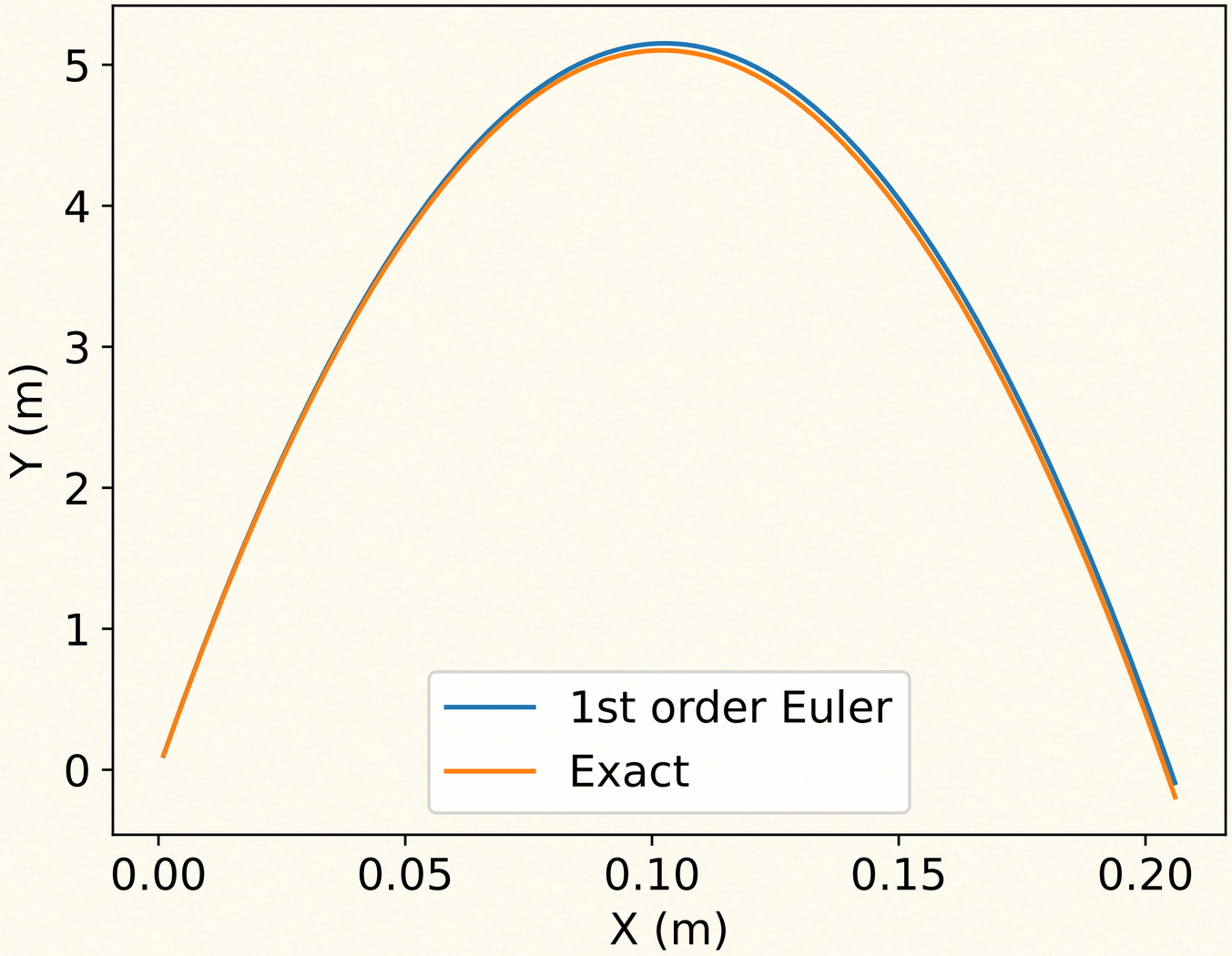
First find velocity

$$v(t_{i+1}) = v(t_i) + \frac{F(v(t_i), x(t_i), t_i)}{m}\Delta t$$

Then position

$$x(t_{i+1}) = x(t_i) + v(t_i)\Delta t$$

Iterate until you go for long enough..



Force model

Need to compute $F(\mathbf{v}, \mathbf{x}, t)$

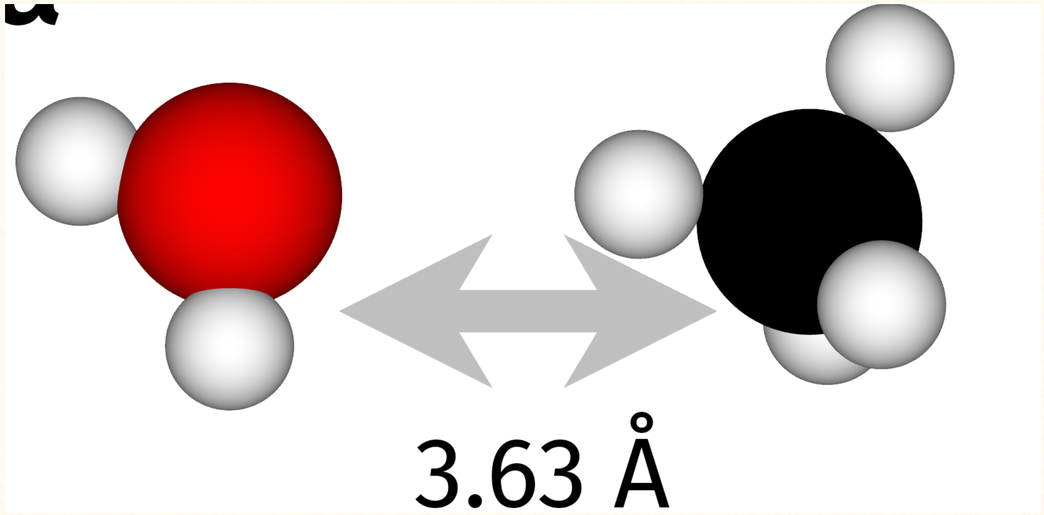
Gravity plus air resistance:
$F(\mathbf{v}, \mathbf{x}, t) = -b\mathbf{v} - mg\hat{y}$

This is why computational approaches are so useful--you can throw any force model at the simulator and it basically is the same effort.

Testing your code: discussion

How can we be confident that our math and implementation are correct?

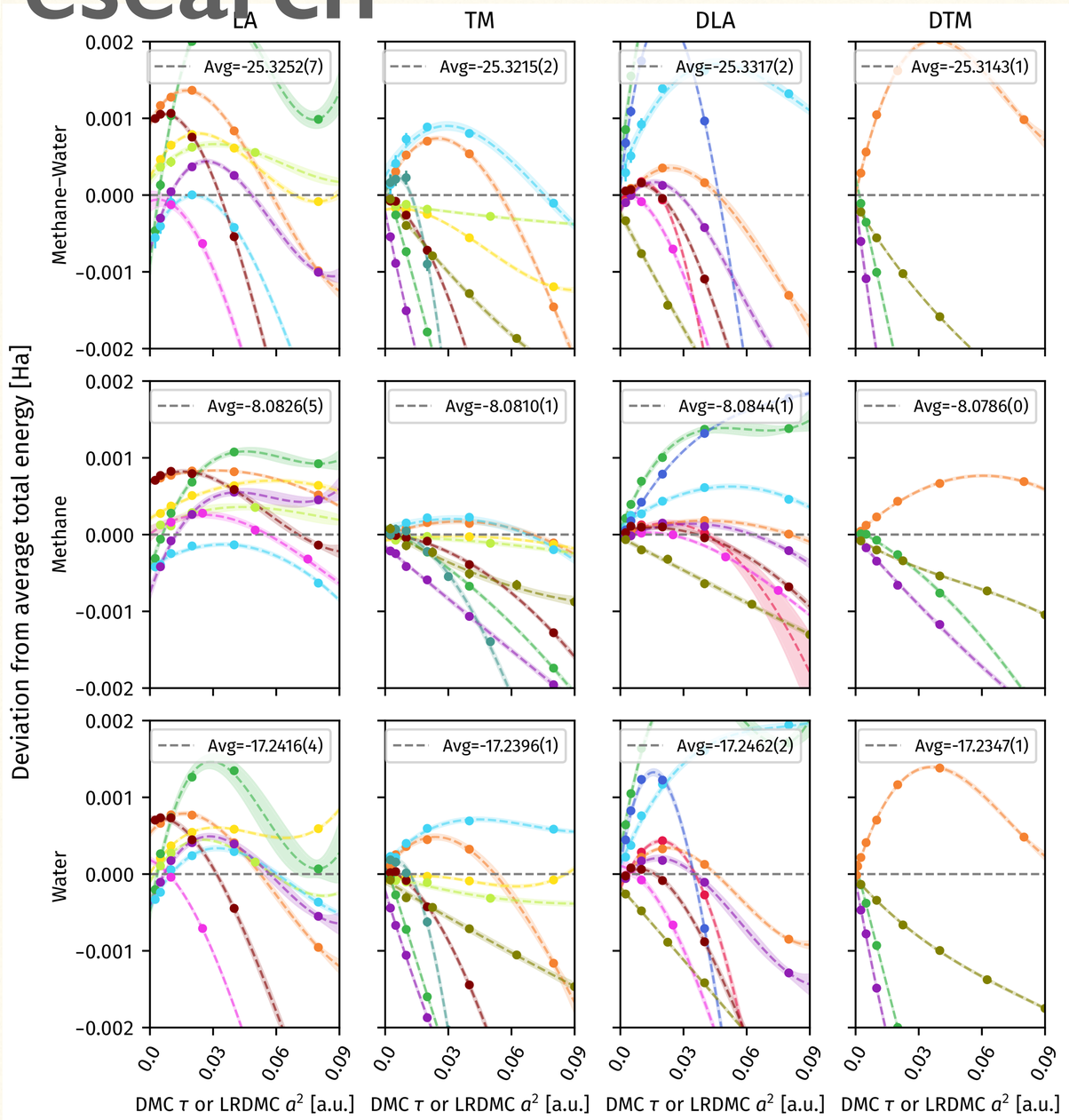
Application of current research



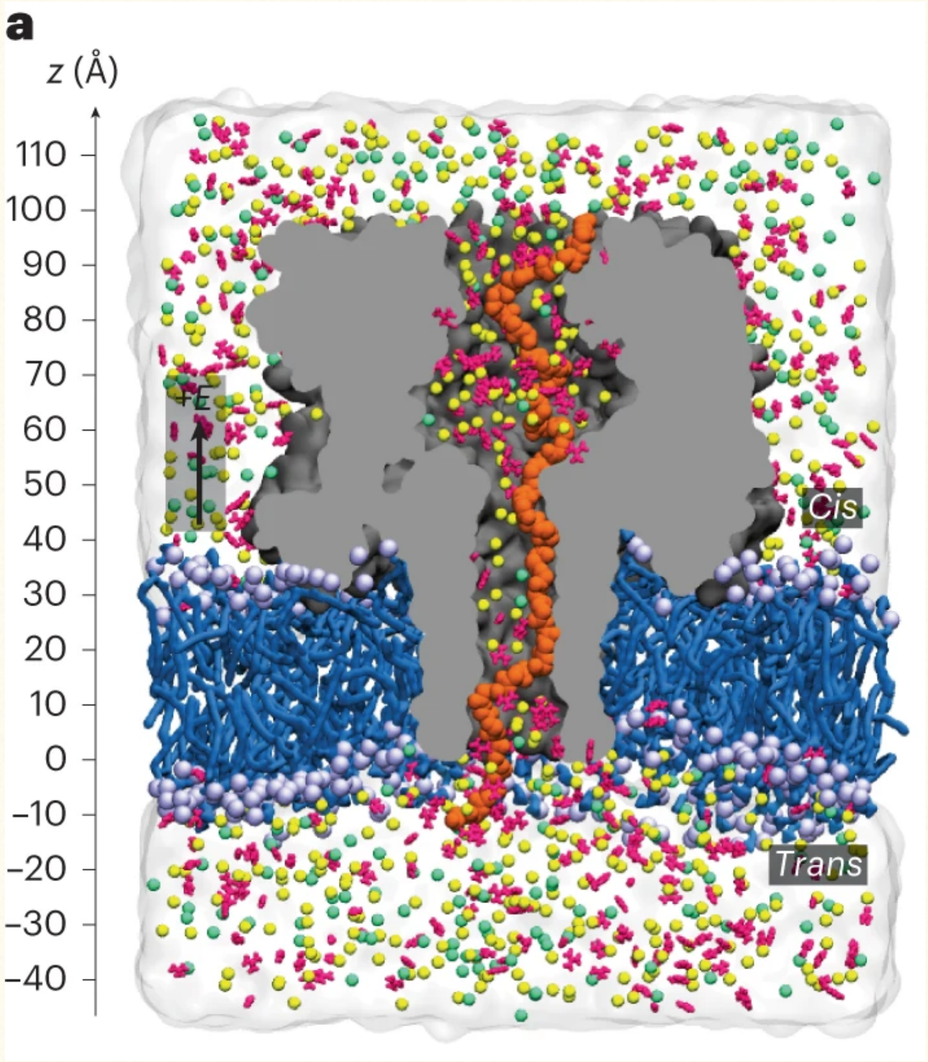
A dozen or so implementations of solutions to the Schrödinger equation in 16*3=48 dimensions.

Limit as $\tau \to 0$ should match!

https://arxiv.org/abs/2501.12950



Molecular dynamics simulations

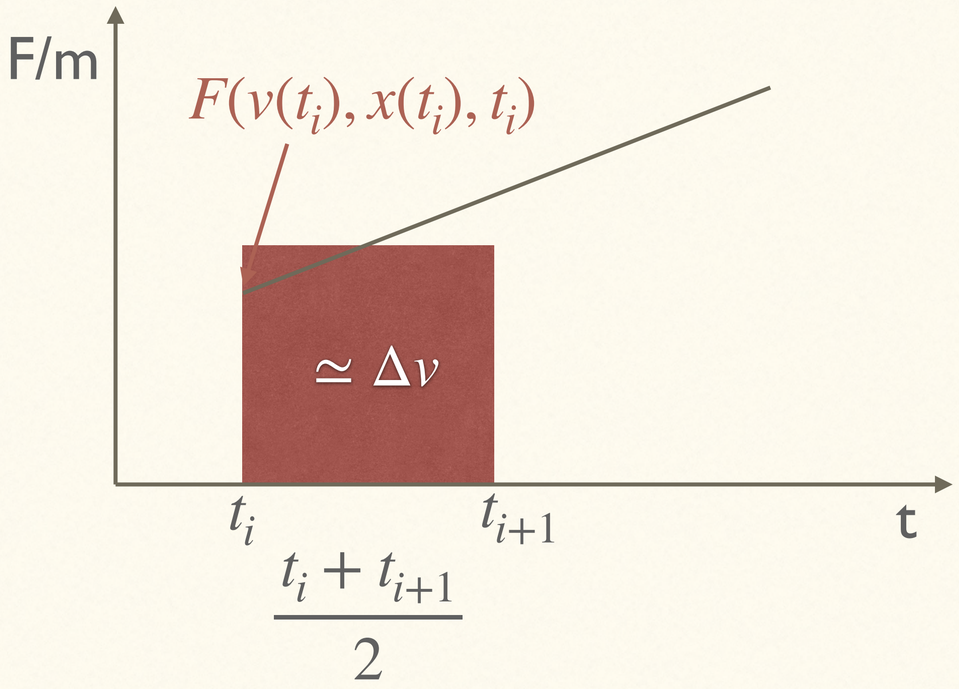


All-atom model of α-hemolysin (gray) containing a fragment of the MBP protein (orange), embedded in a lipid membrane (blue) and submerged in the 1.5 M GdmCl / 1.0 M KCl electrolyte mixture.

~300,000 atoms -- direct simulation of protein dynamics

Nature Biotechnology volume 41, pages 1130–1139 (2023)

Midpoint method (refined Euler)



W know the position and velocity at some time t_i, and want to know these quantities at a future time t_{i+1}

If we can compute $F(v, x, t)$, then we can approximate this integral.

Error is proportional to $(\Delta t)^2$:

$$F = F(t_{mid}) + \frac{dF(t_{mid})}{dt}\Delta t + \frac{1}{2}\frac{d^2F(t_{mid})}{dt^2}(\Delta t)^2 + \ldots$$

Quantifying error (log-log plots)

Typically we say that the error is proportional to Δt^n, but that's the 'leading' term, only valid for small enough Δt.

Notes on Python: lists, dictionaries, and numpy arrays

Python lists: [a,b,c]
- a,b, c can be anything
- "+" means append
- Built-in to python

Numpy arrays
- a,b, c must all be the same type
- "+" means element-wise add
- From the numpy library
- Can be quite fast

Python dictionaries: [a:x,b:y,c:z]
- look up tables
- a,b,c have to be immutable (strings, numbers, tuples)
- x,y,z can be anything
- No '+' operator