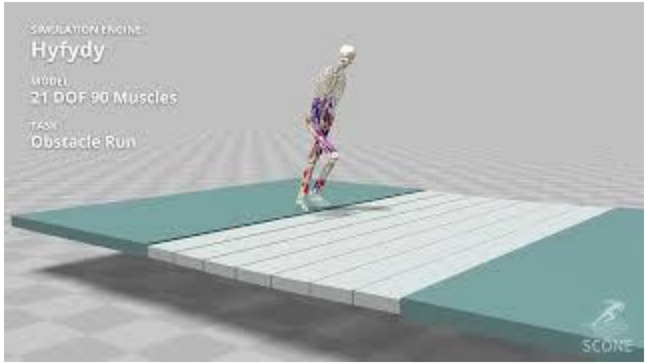
How does simulation work?

Thousands of things we can do in a simulation



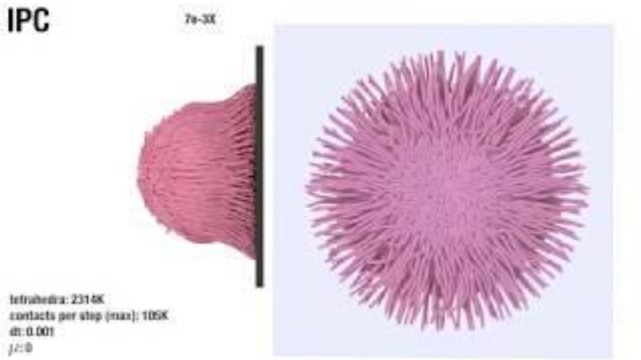


Table of Contents

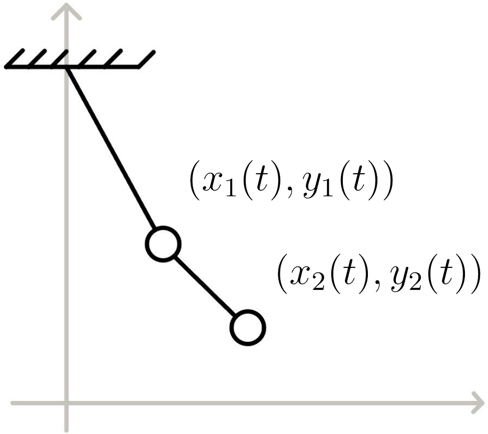
State Parameterisation

How to describe state of a system?

State Parameterisation - General Idea

General idea: **choose a set of parameters (coordinates) → values of these parameters determine how the state of the system changes in time**

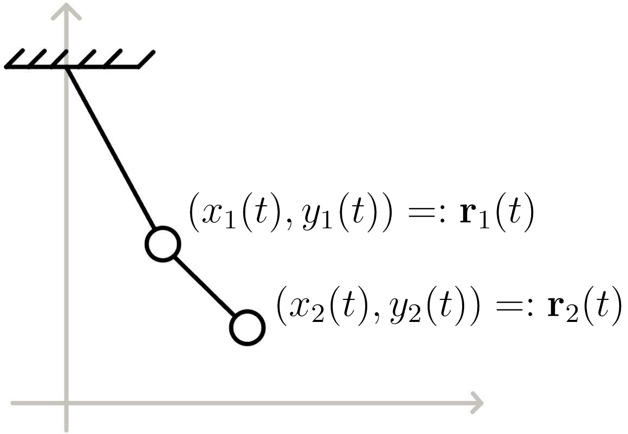
Example - Double Pendulum and Cartesian Coordinates:



Derivatives: Velocity, Acceleration, ...

We care not only about the current values, but also how they change:
velocity, **acceleration** (or higher order derivatives)

<u>Example - Double Pendulum and Cartesian Coordinates:</u>



Velocities of the two masses:

$$\dot{\mathbf{r}}_1(t) := \frac{d\mathbf{r}_1(t)}{dt}, \dot{\mathbf{r}}_2(t) := \frac{d\mathbf{r}_2(t)}{dt}$$

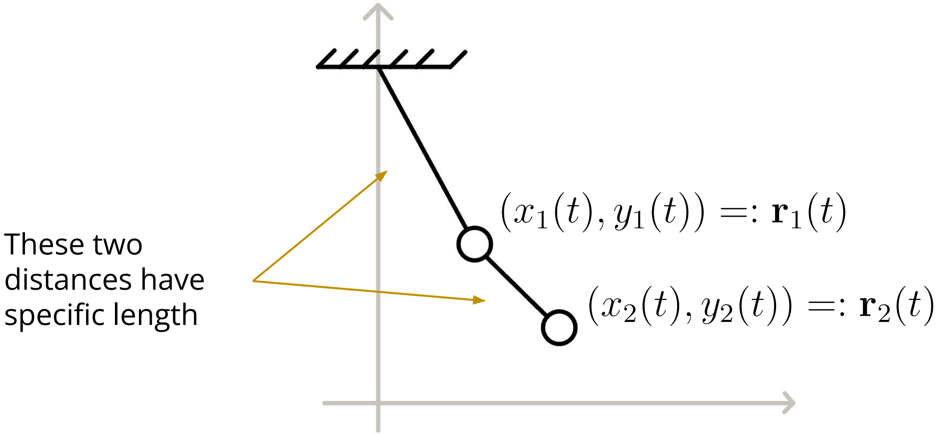
Generalized Coordinates - Holonomic Constraints

Usually the system has to satisfy **constraints**, i.e. the state of the system is somehow restricted

Holonomic constraints → do not depend on the derivatives: $$f(\mathbf{r}_k(t), t) = 0$$

May in general change in time! E.g. changing the length of a pendulum

<u>Example:</u>

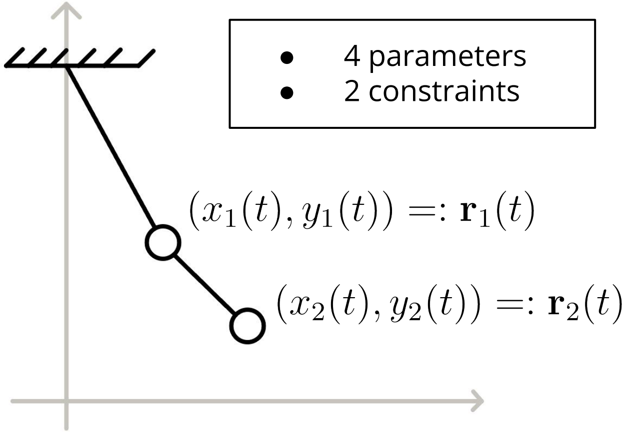


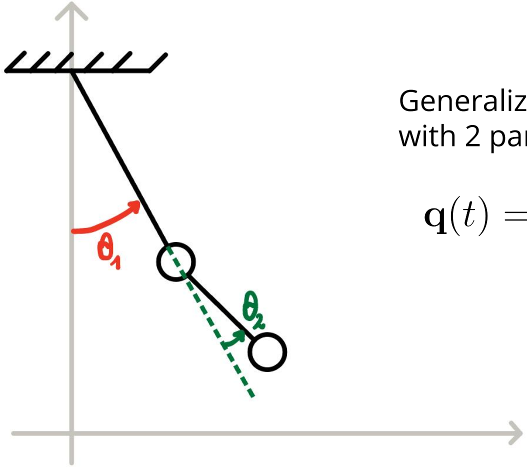
Generalized Coordinates - Degrees of Freedom

- Holonomic constraints → we do not need all parameters
- Degrees of Freedom (DoF) = # parameters - # holonomic constraints
- Generalized coordinates → choose parameters s.t. #parameters = #DoF

<u>Example:</u>

- 4 parameters
- 2 constraints





Generalized coordinates with 2 parameters:

$$\mathbf{q}(t) = (\theta_1(t), \theta_2(t))$$

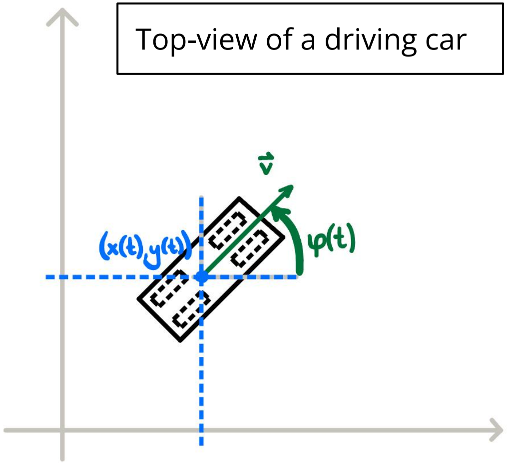
Time Evolution

How to describe how the system is changing?

System Evolution - Constraints

A changing system may have to satisfy more sophisticated constraints → non-holonomic constraints: $g(\mathbf{q}(t), \dot{\mathbf{q}}(t), t) = 0$

<u>Example:</u>

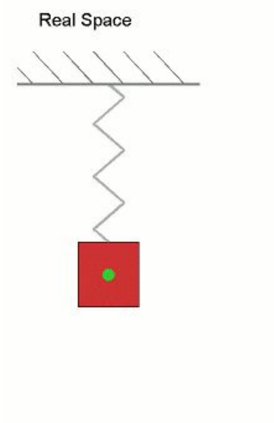


$$\mathbf{q}(t) = (x(t), y(t), \phi(t)) \implies \dot{\mathbf{q}}(t) = ||\mathbf{v}||(\hat{\mathbf{x}} \cos\phi(t) + \hat{\mathbf{y}} \sin\phi(t))$$

Differential Equations to Describe Time Evolution

- In general higher order derivatives may describe not only constraints but the evolution of the state in time → n^{th} order Ordinary Differential Equations (ODEs)

<u>Example:</u>

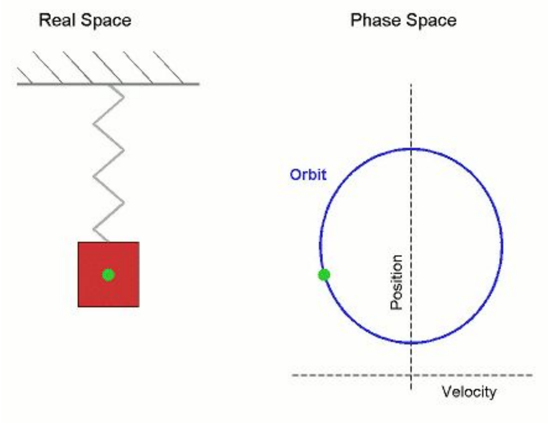


$$\mathbf{F} = m\mathbf{a} \implies \ddot{\mathbf{x}}(t) = -\frac{k}{m}x(t)$$

Differential Equations to Describe Time Evolution

- In practice we can always translate n^{th} order DE^{qs} to a 1^{st} order ODE by changing the state description

<u>Example:</u>



$$\mathbf{q} = \begin{bmatrix} x \\ v \end{bmatrix}$$

$$\Downarrow$$

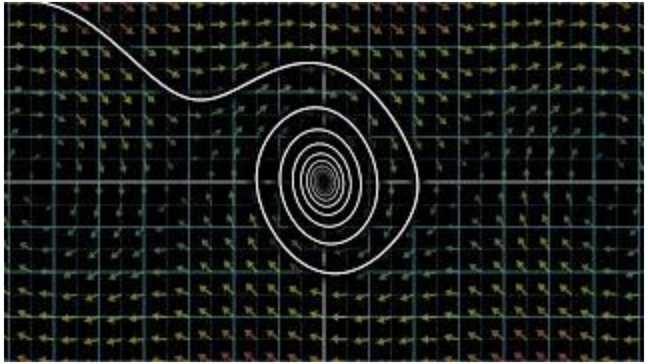
$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} v \\ -\frac{k}{m}x \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} = A\mathbf{q}$$

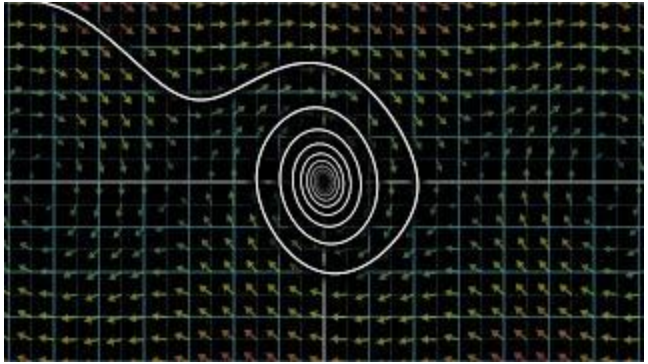
Differential Equations to Describe Physics

- So in general motion can be described as 1st order ODE:

$$\dot{\mathbf{q}} = f(\mathbf{q}(t),\, t)$$

- Think of a vector field *f* (possibly changing in time) determining how the state changes:





Why bother? Second Example - Drone Control

$$a = \frac{F}{m}$$

$$\Downarrow$$

$$\ddot{z} = \underbrace{\frac{k_{prop}}{m} z - \frac{k_{prop}}{m} H}_{\frac{F_p}{m}} - g - \mu \dot{z}$$

Air resistance / Derivative Term

Gravity

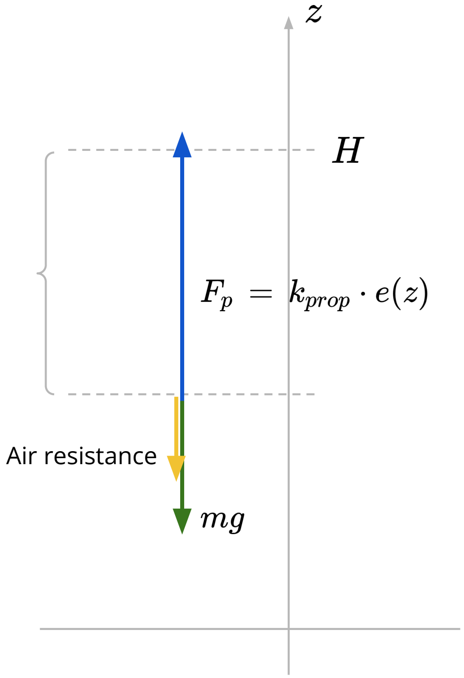
Some renaming for the constants and we get...

$$\ddot{z} = az - \mu \dot{z} - b$$

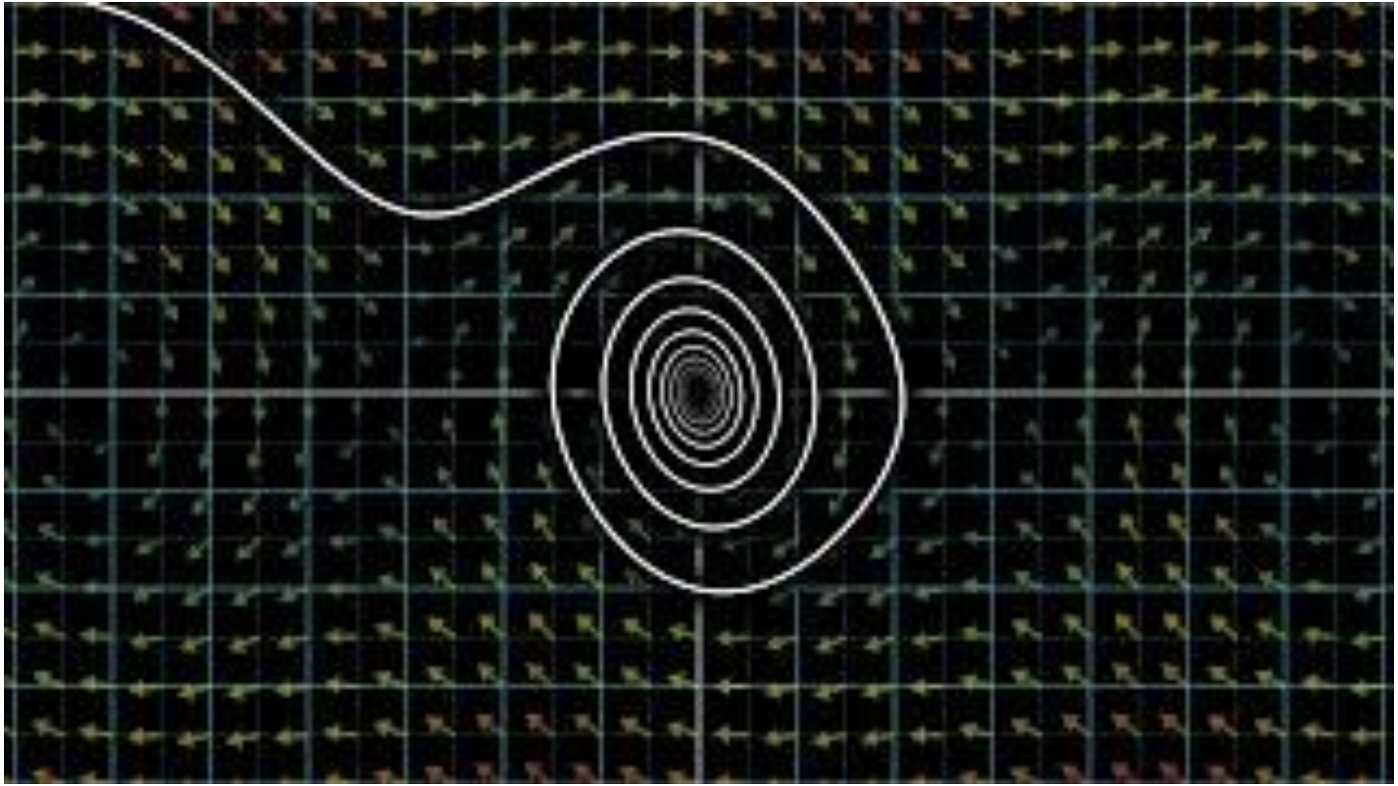
Harmonic motion!

signed error:

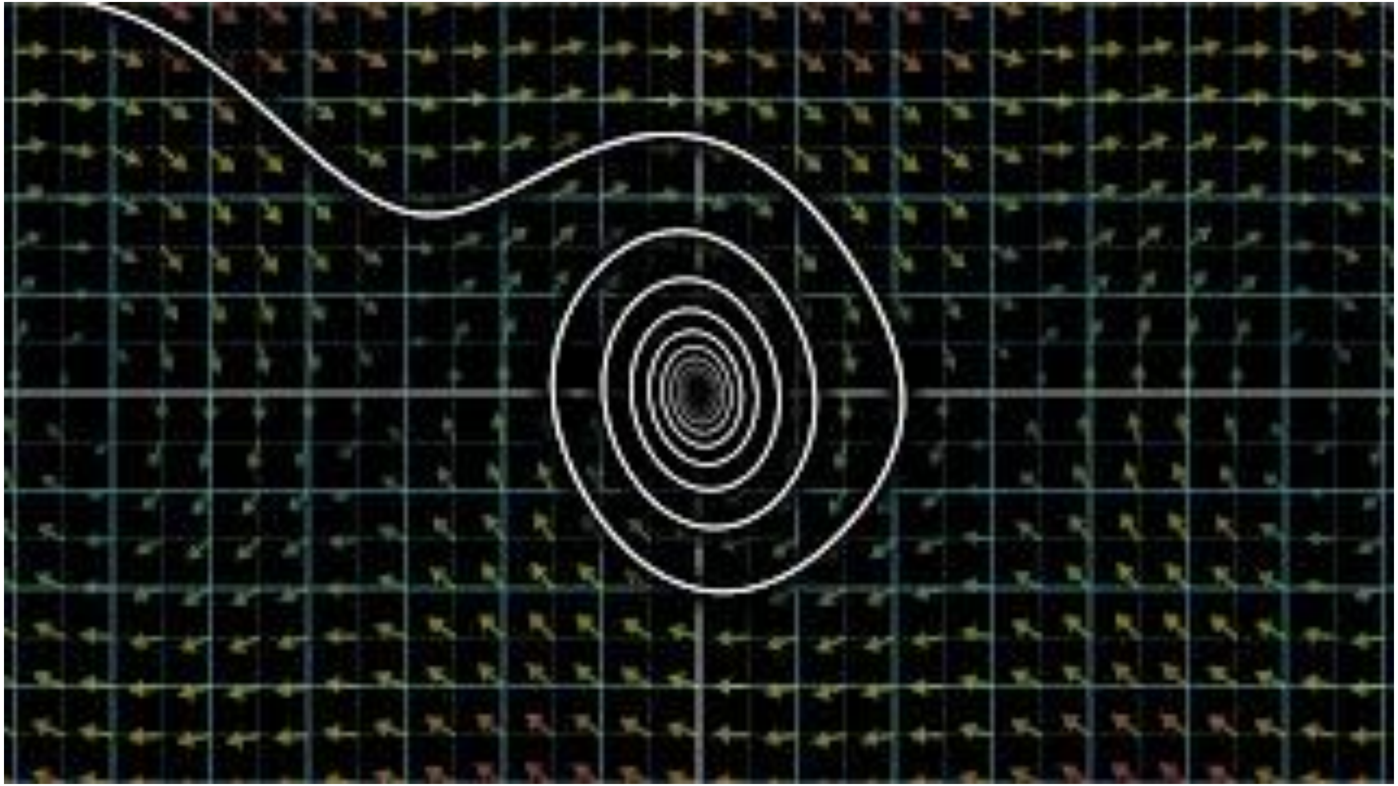
$$e(z) = z - H$$



Third Example - Pendulum - Physics



Third Example - Pendulum - Phase Space



Numerical Integration

How to approximate how the system is changing?

Numerical Integration

- Assume vector field determining system evolution is known, i.e. RHS of: $\dot{\mathbf{q}} = f(\mathbf{q}, t)$
- Approximate derivative
- Determine the next state using obtained approximation

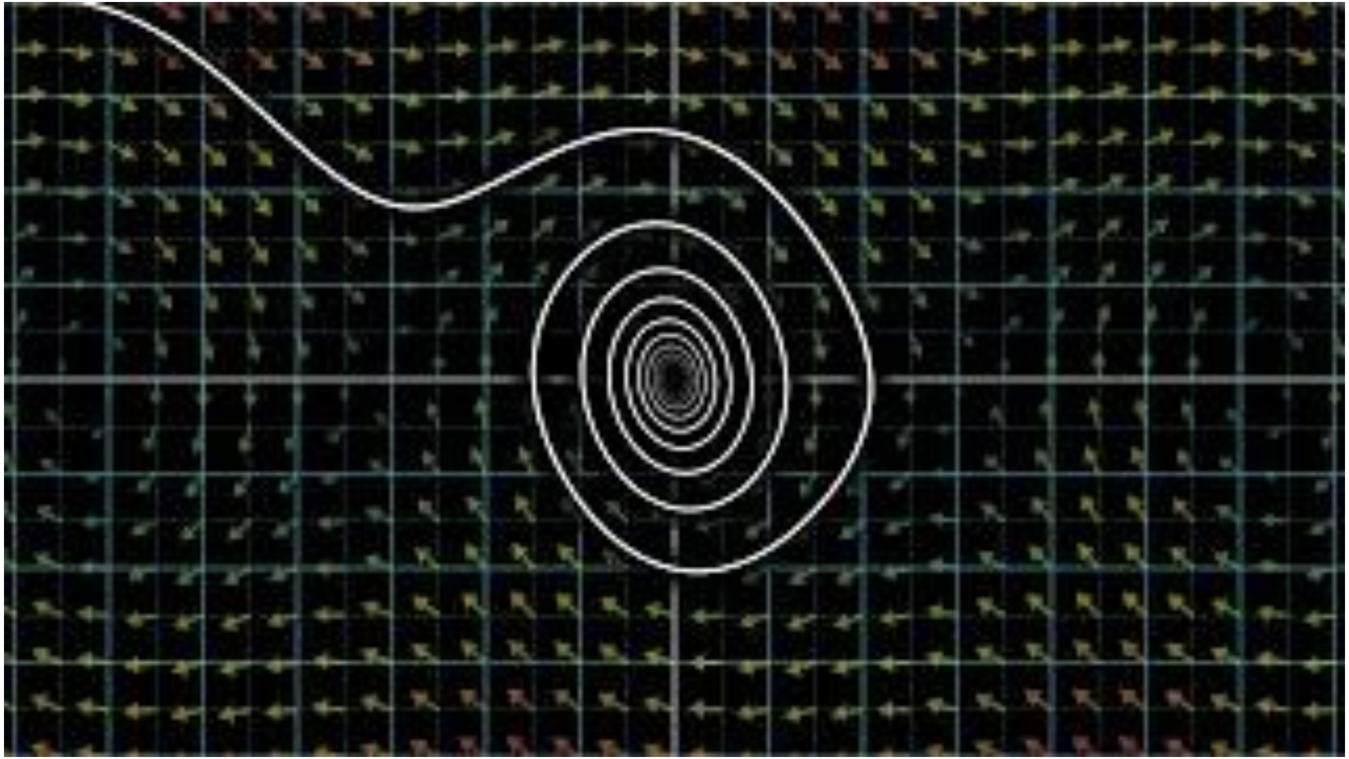
Note that the vector field can be a superposition, e.g. predefined physics + control!

<u>Example - Euler Explicit Integration:</u>

$$f(\mathbf{q}(t), t) = \dot{\mathbf{q}}(t) \approx \frac{\mathbf{q}(t + \Delta t) - \mathbf{q}(t)}{\Delta t} \implies \mathbf{q}(t + \Delta t) \approx \mathbf{q}(t) + \Delta t \cdot f(\mathbf{q}(t), t)$$

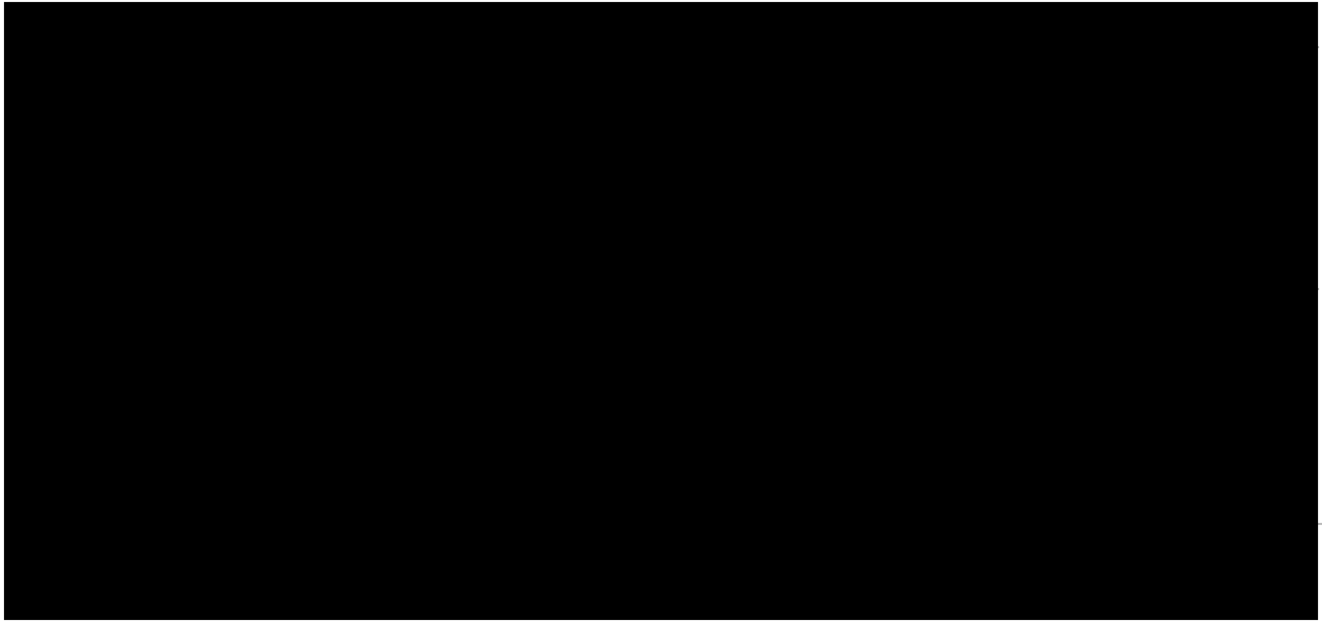
In practice much better approximation schemes are used!

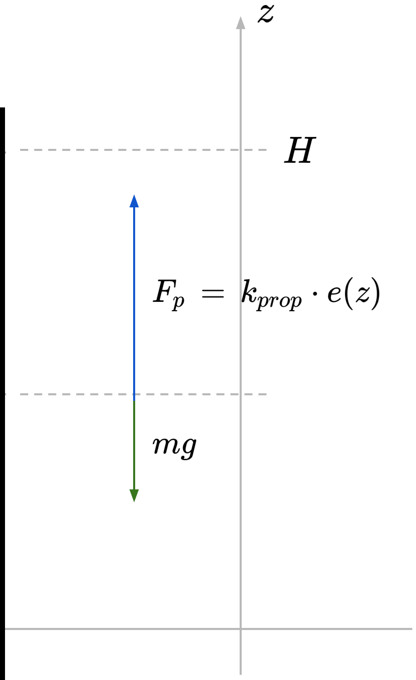
Let’s visualise this and see what can go wrong



Example: Wrong Timestep for Drone Control (P)

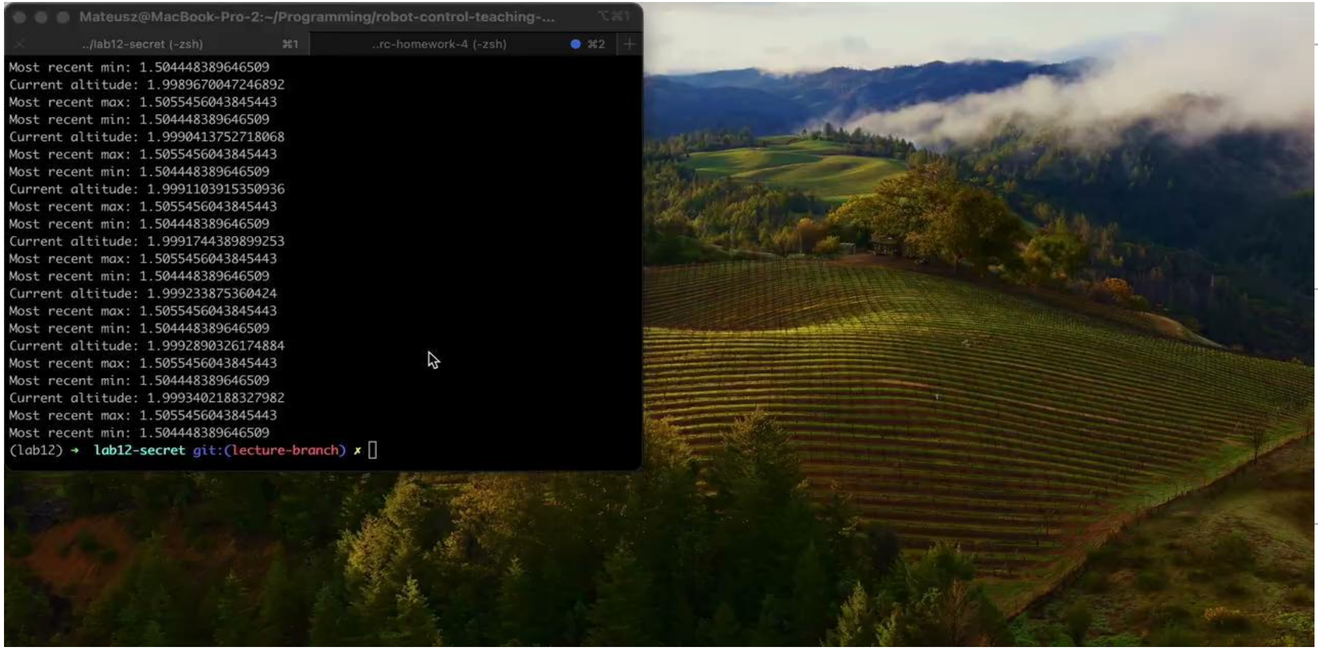
Original correct version with **timestep = 0.01**

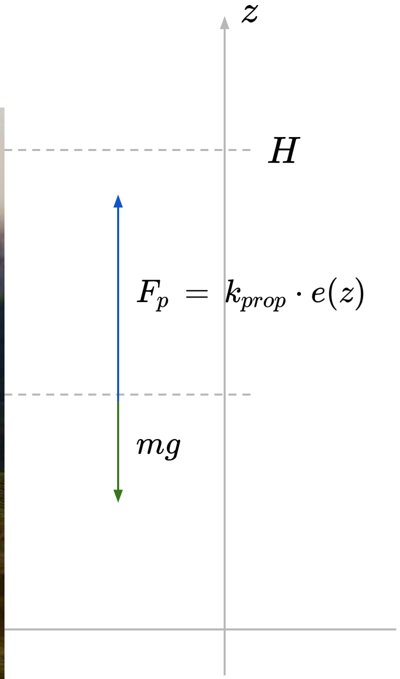




Example: Wrong Timestep for Drone Control (P)

Timestep increased just to **0.04**!





Precision of Numerical Schemes

Remember that we want to approximate the derivative:

$$f(\mathbf{q}(t), t) = \dot{\mathbf{q}}(t) \approx \frac{\mathbf{q}(t + \Delta t) - \mathbf{q}(t)}{\Delta t}$$

<u>Forward Euler</u>

$$\mathbf{q}(t + \Delta t) = \mathbf{q}(t) + \Delta t \cdot \dot{q}(t) + \frac{\Delta t^2}{2!} \ddot{q}(t) + \frac{\Delta t^3}{3!} \dddot{q}(t) + \ldots$$

$$\frac{\mathbf{q}(t + \Delta t) - \mathbf{q}(t)}{\Delta t} = \dot{q}(t) + o(\Delta t)$$

Precision of Numerical Schemes

Remember that we want to approximate the derivative:

$$f(\mathbf{q}(t),t) \;=\; \dot{\mathbf{q}}(t) \approx \frac{\mathbf{q}(t+\Delta t)-\mathbf{q}(t)}{\Delta t}$$

<u>Backward Euler</u>

$$\mathbf{q}(t-\Delta t) = \mathbf{q}(t) \;-\Delta t\cdot\dot{q}\,(t) + \frac{\Delta t^2}{2!}\ddot{q}\,(t) - \frac{\Delta t^3}{3!}\dddot{q}\,(t) + \ldots$$

$$\frac{\mathbf{q}(t)-\mathbf{q}(t-\Delta t)}{\Delta t} \;=\dot{q}\,(t)+o(\Delta t)$$

Precision of Numerical Schemes

Remember that we want to approximate the derivative:

$$f(\mathbf{q}(t), t) \;=\; \dot{\mathbf{q}}(t) \approx \frac{\mathbf{q}(t+\Delta t) - \mathbf{q}(t)}{\Delta t}$$

<u>Central Difference</u>

$$\mathbf{q}(t+\Delta t) = \mathbf{q}(t) \; + \Delta t \cdot \dot{q}\,(t) + \frac{\Delta t^2}{2!}\ddot{q}\,(t) + \frac{\Delta t^3}{3!}\,\dddot{q}\,(t) + \ldots$$

$$\mathbf{q}(t-\Delta t) = \mathbf{q}(t) \; - \Delta t \cdot \dot{q}\,(t) + \frac{\Delta t^2}{2!}\ddot{q}\,(t) - \frac{\Delta t^3}{3!}\,\dddot{q}\,(t) + \ldots$$

$$\mathbf{q}(t+\Delta t) - \mathbf{q}(t-\Delta t) = 2\Delta t \cdot \dot{q}\,(t) + 2\frac{\Delta t^3}{3!}\,\dddot{q}\,(t) + \ldots$$

$$\frac{\mathbf{q}(t+\Delta t) - \mathbf{q}(t-\Delta t)}{2\Delta t} \;= \dot{q}\,(t) + o\big(\Delta t^2\big)$$

More on these topics

AKA References

References

- Lectures on numerical differentiation and integration: https://www.youtube.com/playlist?list=PLMrJAkhIeNNTYaOnVI3QpH7jgULnAmvPA
- https://research.nvidia.com/labs/toronto-ai/vid2player3d/
- https://hungyuling.com/character-controllers-motion-vaes/
- https://vladlen.info/publications/continuous-character-control-with-low-dimensional-embeddings/
- https://robotic-pretrained-transformer.github.io/
- https://gengshan-y.github.io/ppr/
- https://www.albertpumarola.com/research/D-NeRF/index.html