PSCB57 - PROF. HANNO REIN

EXAM PREPARATION

GENERAL TIPPS

- I will test your understanding of concepts, not memorization.
- Be able to transfer existing knowledge to a new area.
- Everything from the lectures, the assignments, and tutorials, can be on the final exam.

GENERAL TIPPS

- No complicated calculations, i.e. no calculator needed.
- If a calculation gets difficult, that might be an indication that you made a mistake.
- If there are many questions, answer the questions that you know first. Keep track of the time.

GENERAL TIPPS

- Don’t get confused if a question uses a different symbol than the one we used in the lecture!

GENERAL TIPPS

- It’s ok not to answer a question.
- If you do not understand a question or are unsure what is asked for, raise your hand and ask for clarification. Others might have the same problem.

PSCB57 - PROF. HANNO REIN

PYTHON

PYTHON

- You should be comfortable with reading short python programs
- Understand control structures (`if`/`for`/`while`), variables, lists, built-in functions such as `len`, `print`, etc.

PYTHON

```
def x(l):
    N = len(l)
    r = -1e307
    for i in range(N):
        if r < l[i]:
            r = l[i]
    return r
```

PYTHON

- No need to know detailed syntax for functions
- No need to worry about getting indices on matrices right
- You’re not expected to code up any significant program on paper

PSCB57 - PROF. HANNO REIN

NUMBER FORMATS

INTEGERS

- Fixed number of bits to represent an integer number
- Typical: 16, 32, 64 bits
- Finite ranges:
  $0..2^{16}-1$, $0..2^{32}-1$, $0..2^{64}-1$ (unsigned)
  $-2^{15}..2^{15}-1$, $2^{31}..2^{31}-1$, $2^{63}..2^{63}-1$ (signed)

INTEGERS

- Python 3 does something special:
  It automatically increases the number of bits if you run over the range.
- Other programming languages (including earlier versions of python) behave differently

INTEGERS

What do you use integers for?

- Counters
- Exact calculations

INTEGERS

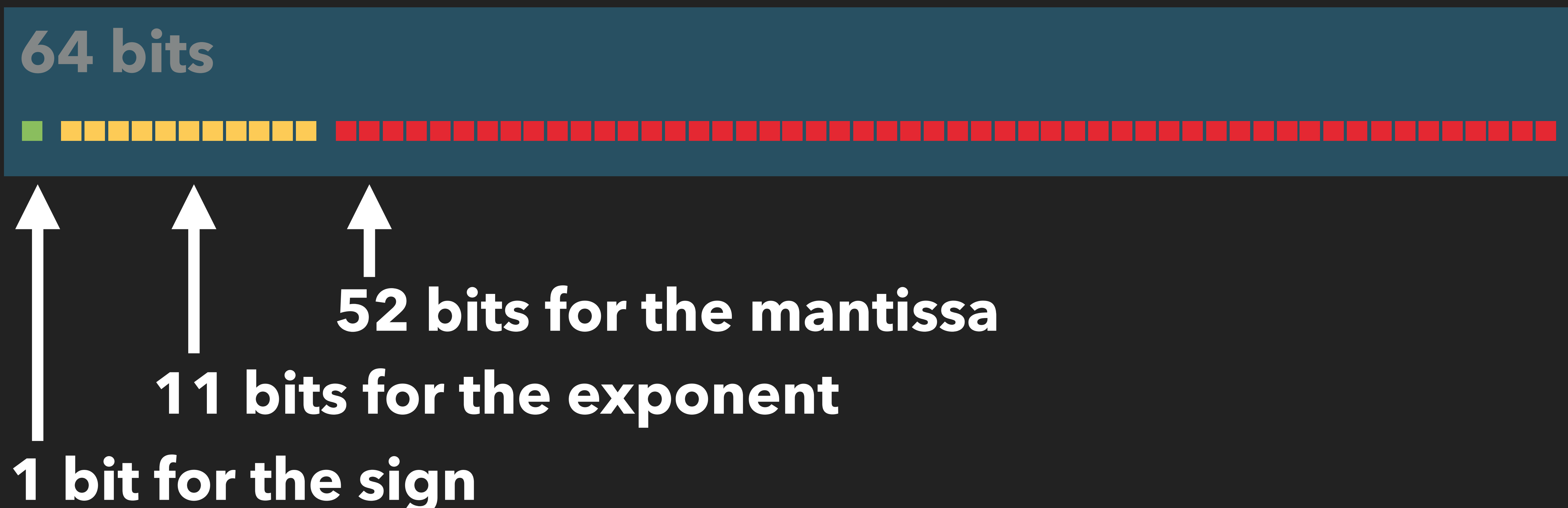
What are integers **not** good for?

- Calculations with a large dynamic range (i.e. most scientific applications!)

FLOATING POINT NUMBERS

- Fixed number of bits (64 for double precision that we focussed on)
- Be able to decode simple binary representations of floating point numbers

FLOATING POINT NUMBERS



$$x = (-1)^s \cdot 2^{e-1023} \cdot (1 + m)$$

FLOATING POINT NUMBERS

Important numbers to remember:

- Range: ~ -1e-308…1e+308
- Precision: ~1e-16

FLOATING POINT NUMBERS

When do operations become problematic?

- 1e+30 + 3.4 = 1e+30
- 1e-16 + 1e-19 = 1.001e-16

FLOATING POINT NUMBERS

How to prepare for the exam?

- Look at the Jupyter notebooks in the course repository.
- Try to convert a few floating point numbers by hand.
- Try to come up with floating point expressions that barely work.

PSCB57 - PROF. HANNO REIN

ALGORITHMIC COMPLEXITY

ALGORITHMIC COMPLEXITY

Idea:

- The time the algorithm takes to complete a calculation depends on some number N
- N can be size of your dataset, number of steps in an integration, or the number of outputs.
- How does the runtime *scale* for large N?

ALGORITHMIC COMPLEXITY

$O(1)$	Constant
$O(\log(N))$	Logarithmic
$O(N)$	Linear
$O(N \log(N))$	Log Linear
$O(N^2)$	Quadratic
$O(N^3)$	Cubic
$O(2^N)$	Exponential

ALGORITHMIC COMPLEXITY

How to determine the complexity of a given piece of code:

- Is it recursive? How many times does it call itself?
- Closely look at for/while loops. Are they nested?
- Focus on the big picture, ignore details.

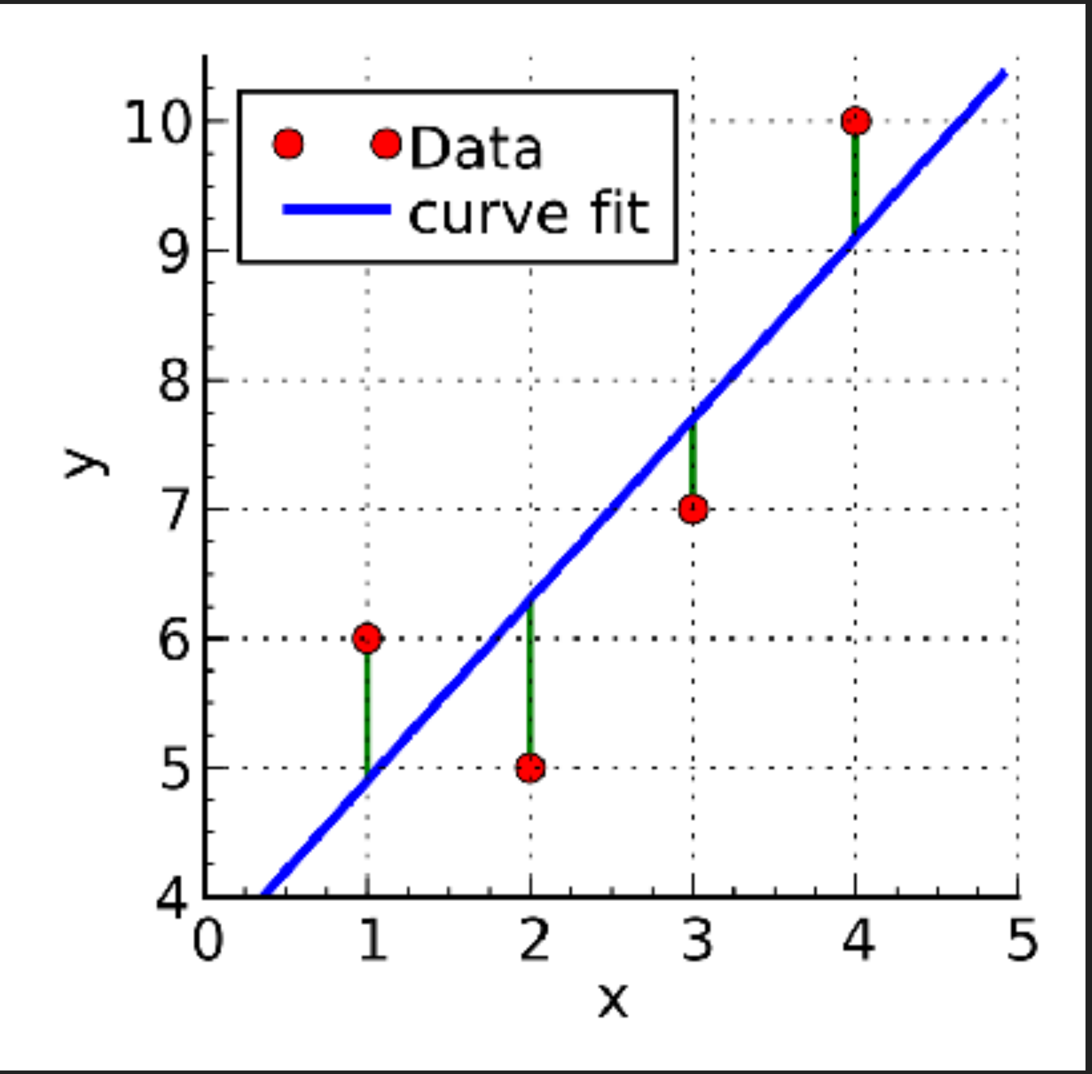
PSCB57 - PROF. HANNO REIN

LINEAR LEAST SQUARE FIT

LINEAR LEAST SQUARE FIT

Definition: minimize the "sum of squares"

$$S = \sum_{i=0}^{N-1} e_i^2$$



LINEAR LEAST SQUARE FIT

- We have a function with a set of free parameters a.
- Want to parameters a to minimize S.
- This lead us to the matrix equation:

$$\underbrace{C^T \cdot y}_{b} = \underbrace{\left(C^T C\right)}_{A} \cdot a$$

LINEAR LEAST SQUARE FIT

- Where the matrix C depends on the function we want to fit and the datapoints. For example:

$$f(t) = a_0 + a_1 \sin\left(\frac{t}{24}2\pi\right) + a_2 \cos\left(\frac{t}{24}2\pi\right)$$

$$\begin{pmatrix} 1 & \sin\left(\frac{x_0}{24}2\pi\right) & \cos\left(\frac{x_0}{24}2\pi\right) \\ 1 & \sin\left(\frac{x_1}{24}2\pi\right) & \cos\left(\frac{x_1}{24}2\pi\right) \\ & \vdots & \\ 1 & \sin\left(\frac{x_{N-1}}{24}2\pi\right) & \cos\left(\frac{x_{N-1}}{24}2\pi\right) \end{pmatrix}$$

LINEAR LEAST SQUARE FIT

- You should be able to construct the matrix C, as well as the vector b and matrix A for arbitrary functions and datapoints.
- You are expected to then solve the linear system of equations *only* if the number of parameters is <=2.

LINEAR LEAST SQUARE FIT

- What does the term *linear* refer to?

$$f(t) = a_0 + a_1 \sin\left(\frac{t}{24}2\pi\right) + a_2 \cos\left(\frac{t}{24}2\pi\right)$$

- Know when you cannot fit a function using a *linear* least square fit.

PSCB57 - PROF. HANNO REIN

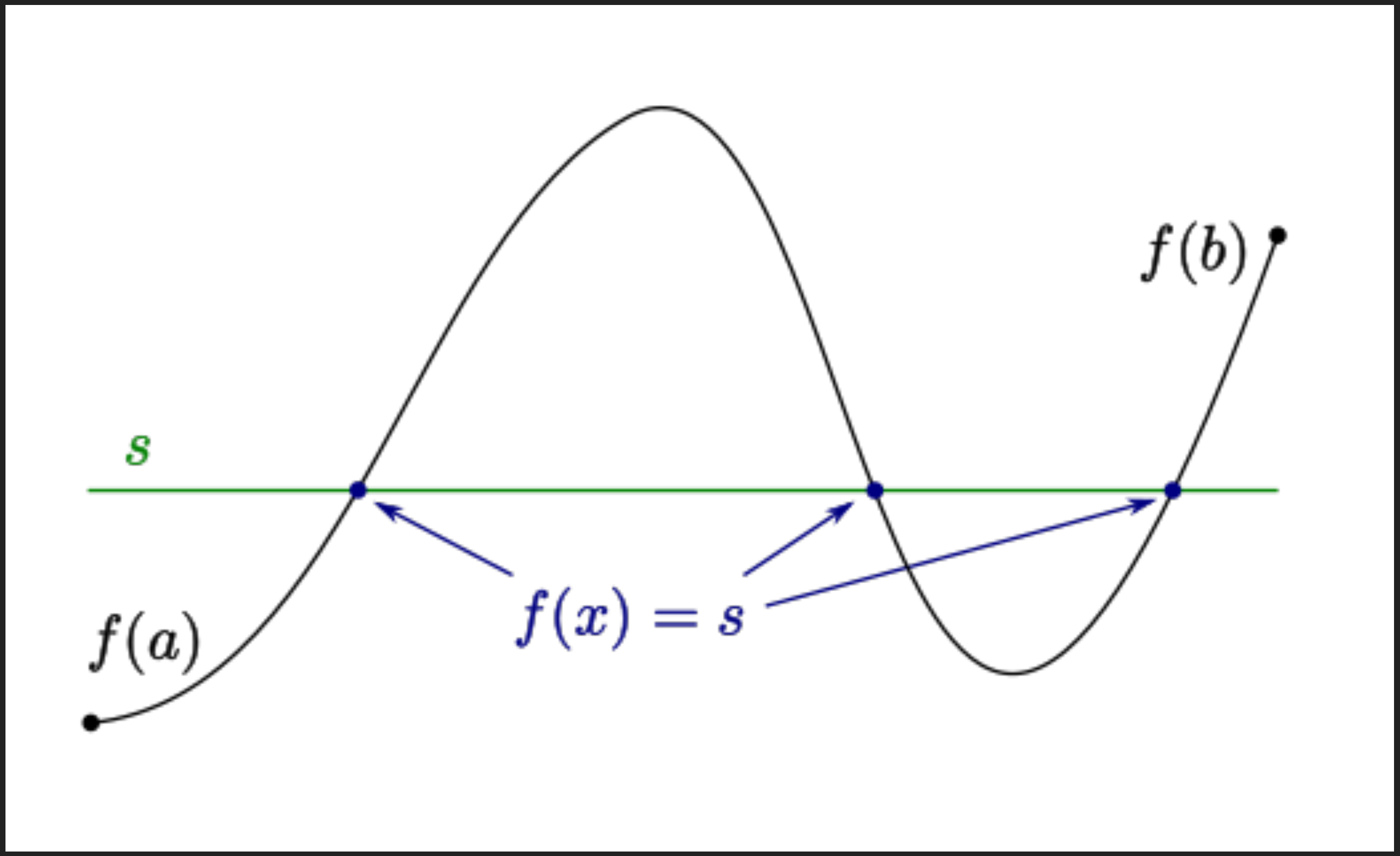
ROOT FINDING METHODS

ROOT FINDING METHODS

- Where do we encounter root finding?
- Least Square Fit
- Optimization methods
- Constrained equations

ROOT FINDING METHODS

- Intermedia value theorem.



- Guarantees existence of a root.

ROOT FINDING METHODS

- Intermedia value theorem directly leads to the bisection method
- Bisection method always works!
- Needs starting interval
- Reduces interval by half at each step

ROOT FINDING METHODS

- How many times do you have to iterate the bisection method when using double floating point precision?
- At most 52 times!

ROOT FINDING METHODS

- Other root finding methods: Newton's method
- Even faster than bisection
- Needs a starting point (no interval)
- Need to know the derivate of the function.
- Might not converge!

ROOT FINDING METHODS

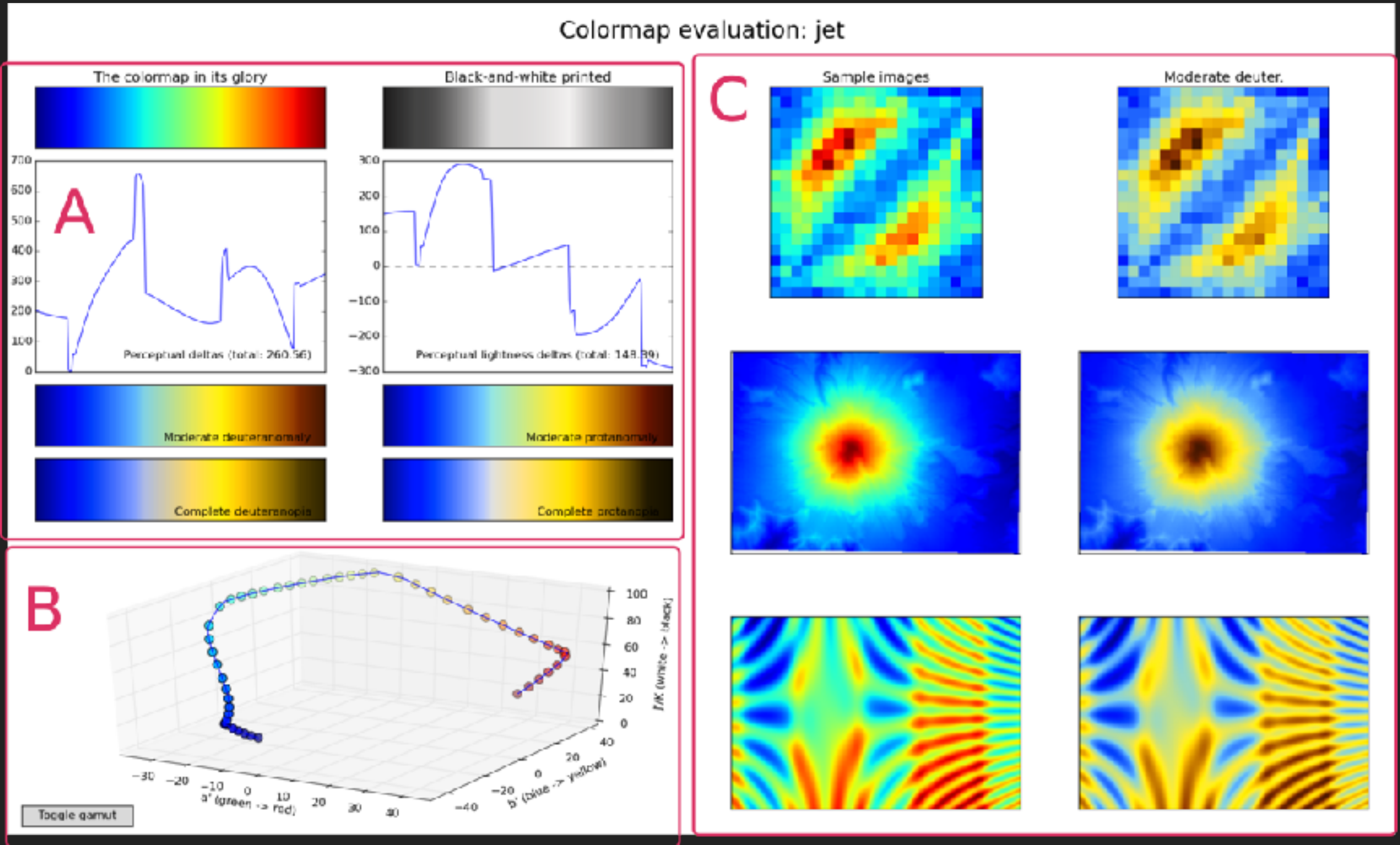
- Root finding is a very large topic.
- We just scratched the surface.
- Our methods work well for 1D
- High dimensional problems are MUCH harder

PSCB57 - PROF. HANNO REIN

PLOTTING

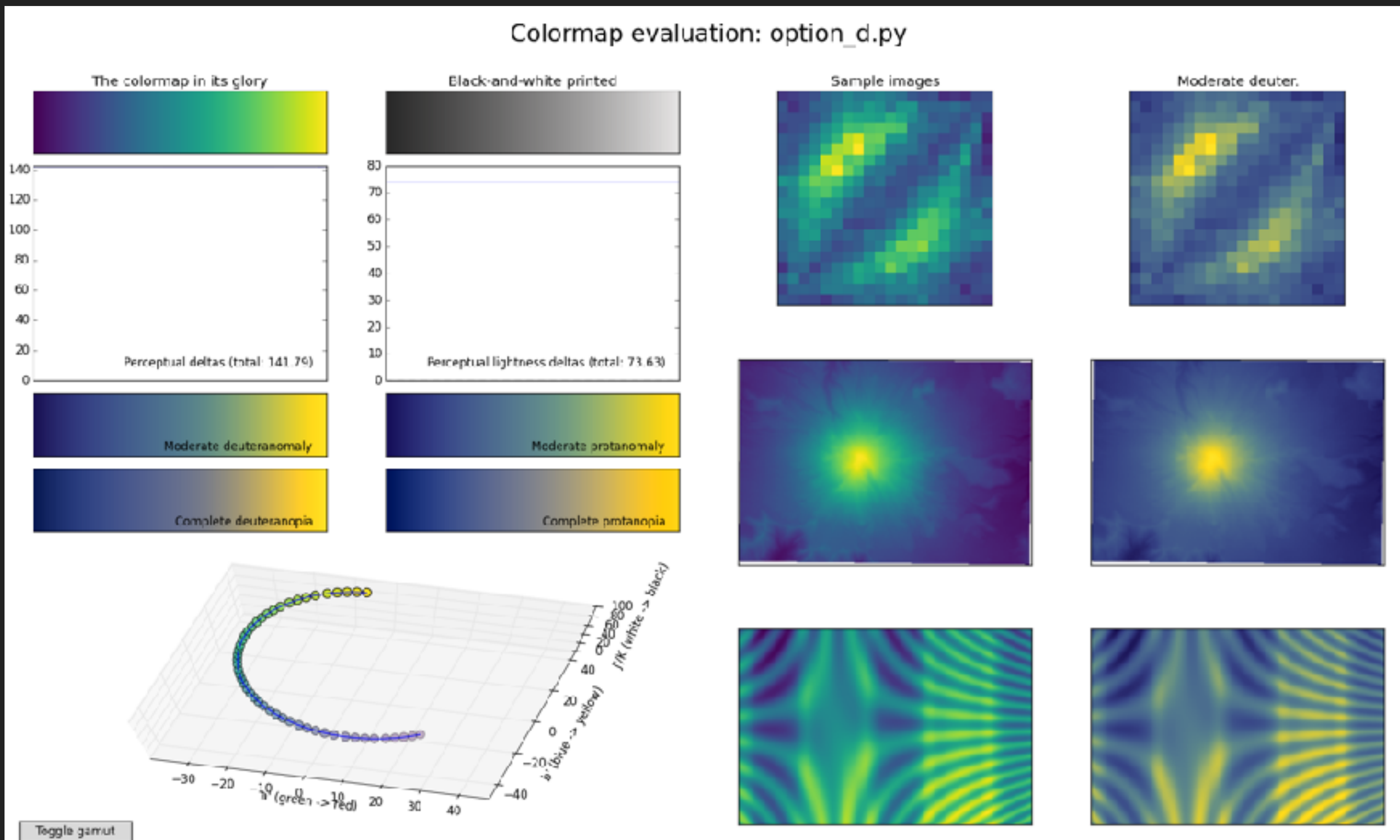
PLOTTING

- Non-perceptually uniform colour map



PLOTTING

- Perceptually uniform colour map



PSCB57 - PROF. HANNO REIN

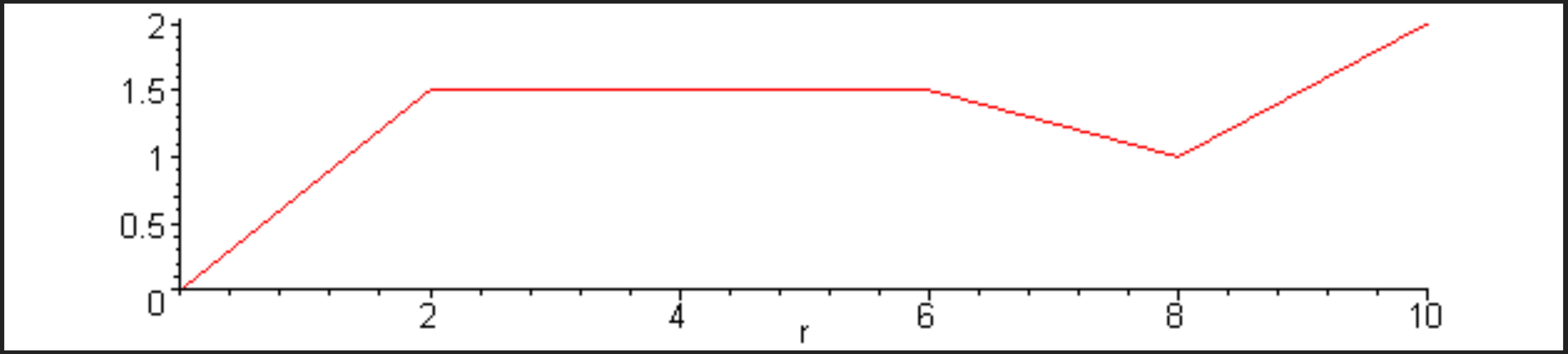
INTERPOLATION METHODS

INTERPOLATION METHODS

- Difference between interpolation and fit
- Interpolation goes through all data points, independent of any model
- Note that plotting data points and connecting them by lines is already an interpolation

INTERPOLATION METHODS

- Nearest neighbour / constant interpolation / Voronoi mesh
- Piece-wise linear interpolation



INTERPOLATION METHODS

- Lagrange interpolation

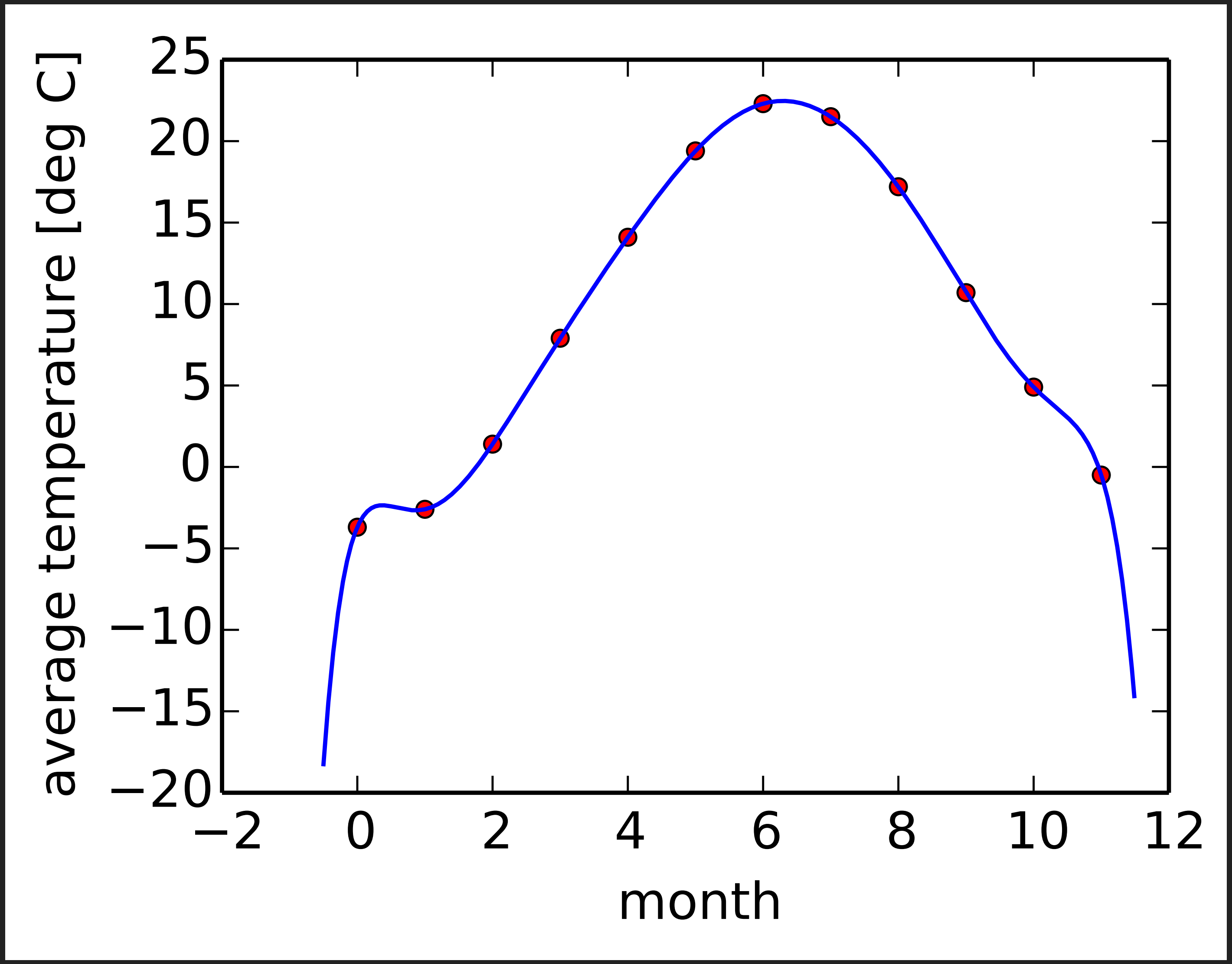
$$L(x) = \sum_{j=0}^{N-1} y_j \ell_j(x)$$

- With basis polynomials

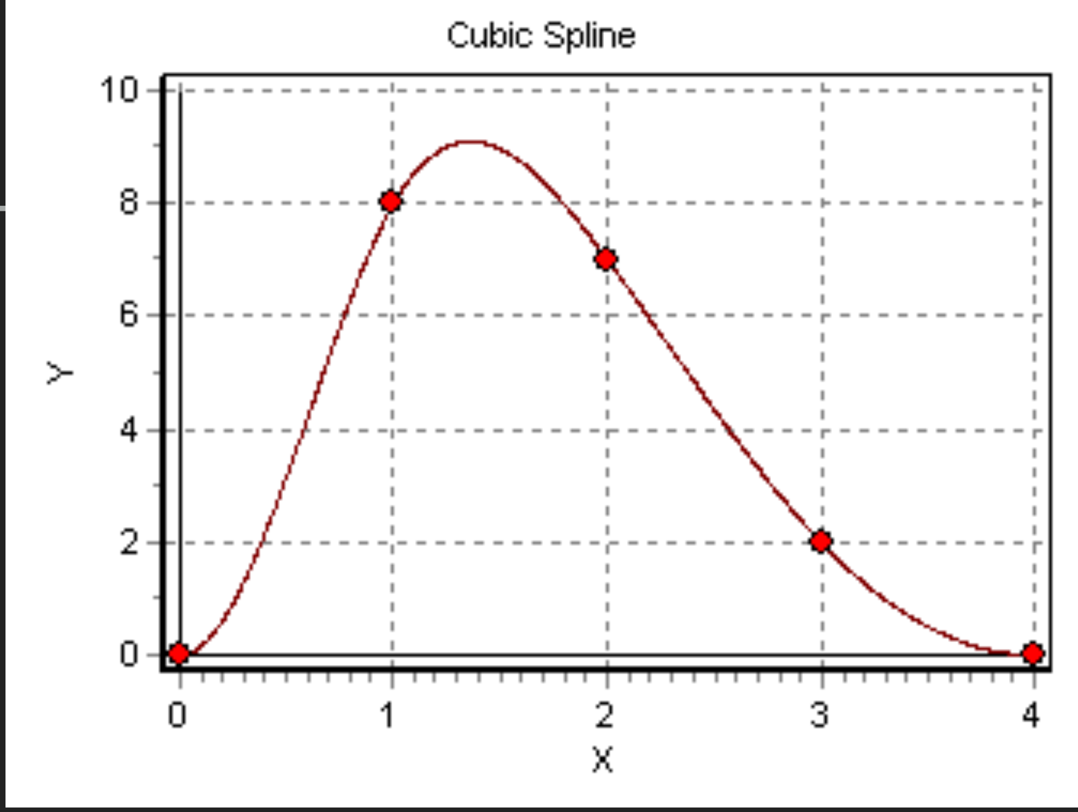
$$\ell_j(x) = \prod_{\substack{0 \le m \le N-1 \\ m \ne j}} \frac{x - x_m}{x_j - x_m} = \frac{(x - x_0)}{(x_j - x_0)} \cdots \frac{(x - x_{j-1})}{(x_j - x_{j-1})} \frac{(x - x_{j+1})}{(x_j - x_{j+1})} \cdots \frac{(x - x_{N-1})}{(x_j - x_{N-1})}$$

INTERPOLATION METHODS

- Problems with Lagrange interpolation



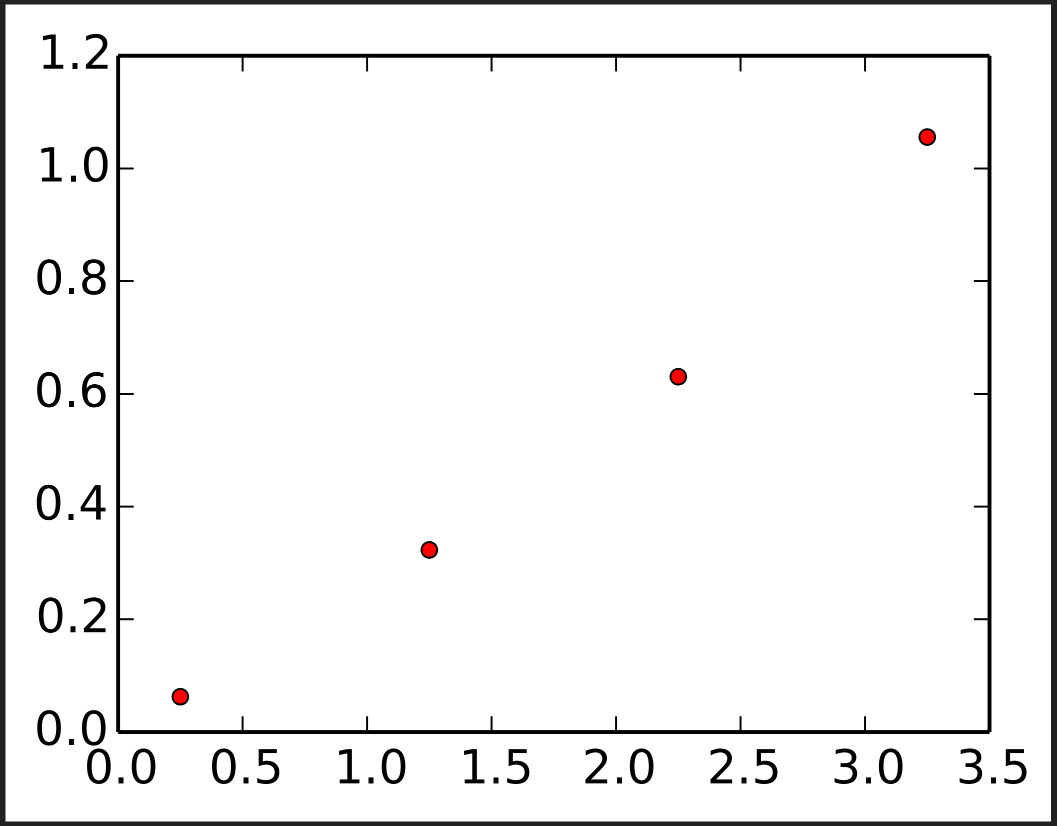
INTERPOLATION METHODS



- Cubic Spline
- Know the definition: a piecewise cubic polynomial that goes through all datapoints, matched derivatives at datapoints to make it smooth
- You do not need to know: how to derive matrix and how to solve it.

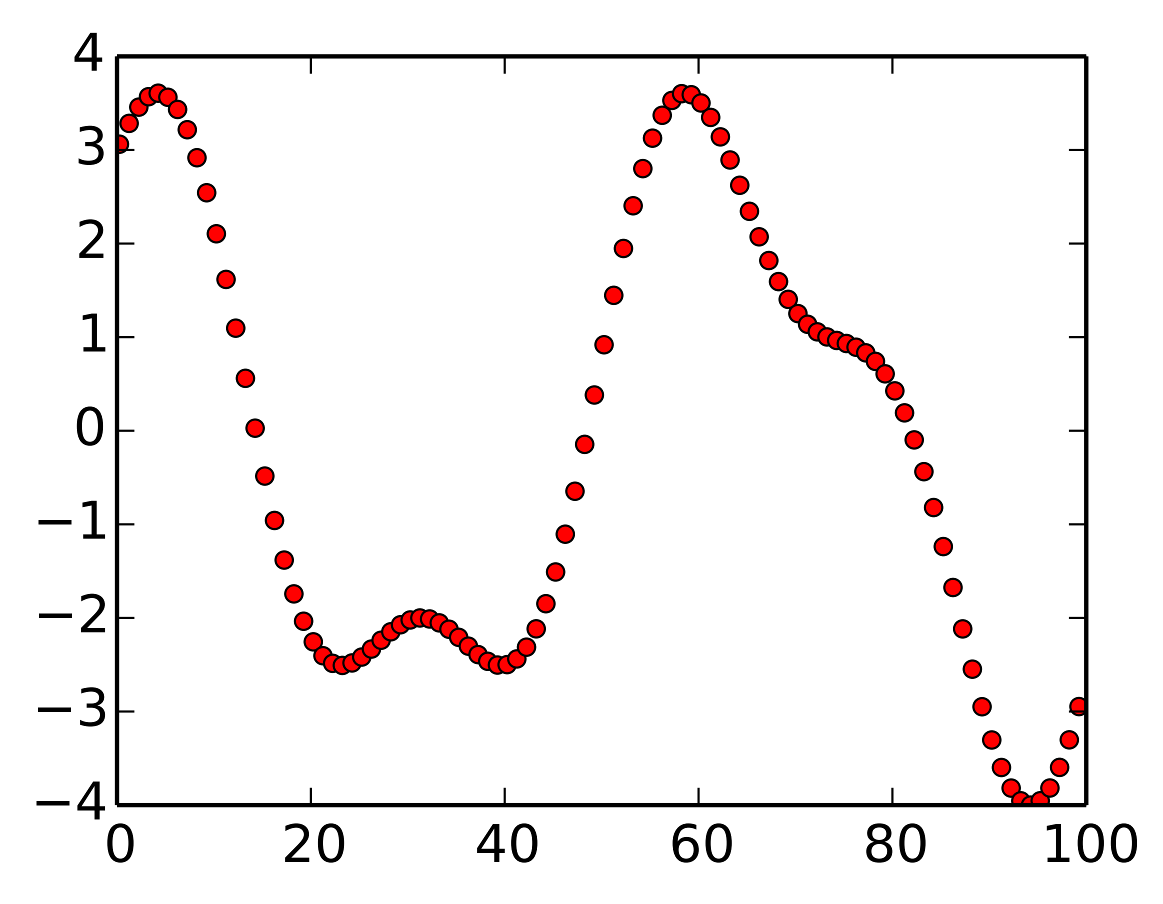
INTERPOLATION METHODS

- You should be able to choose the appropriate interpolation method!



INTERPOLATION METHODS

- You should be able to choose the appropriate interpolation method!



PSCB57 - PROF. HANNO REIN

DIFFERENTIAL EQUATIONS

DIFFERENTIAL EQUATIONS

- Definition: A set of equations where the solution is a function.
- We talk about ordinary differential equations in this course.
- They have an order, determined by the highest derivative.

DIFFERENTIAL EQUATIONS

- Some differential equations depend explicitly on time, others do not (autonomous)
- In general we write a first order ordinary differential equations in the form

$$y'(t) = F(y, t)$$

DIFFERENTIAL EQUATIONS

- Note that the names of the variables might differ depending on the problem at hand.
- You need to identify which is the time variable, which is the right hand side, etc

$$y'(t) = F(y,t)$$

DIFFERENTIAL EQUATIONS

- You can rewrite any high order differential equation as a set of first order differential equations.
- This is important because almost all the methods we talked about are for first order differential equations.
- Practice how to do that!

DIFFERENTIAL EQUATIONS

- Every differential equations needs initial conditions!
- First order -> 1 initial condition
- Second order -> 2 initial conditions
- Etc

DIFFERENTIAL EQUATIONS

- We talked about multiple numerical methods to solve differential equations.
- All work by splitting the time into very small timesteps dt
- The smaller the timestep, the more accurate, but also the more expensive the method

DIFFERENTIAL EQUATIONS

- Explicit Euler method
- Simplest method possible
- 1st order

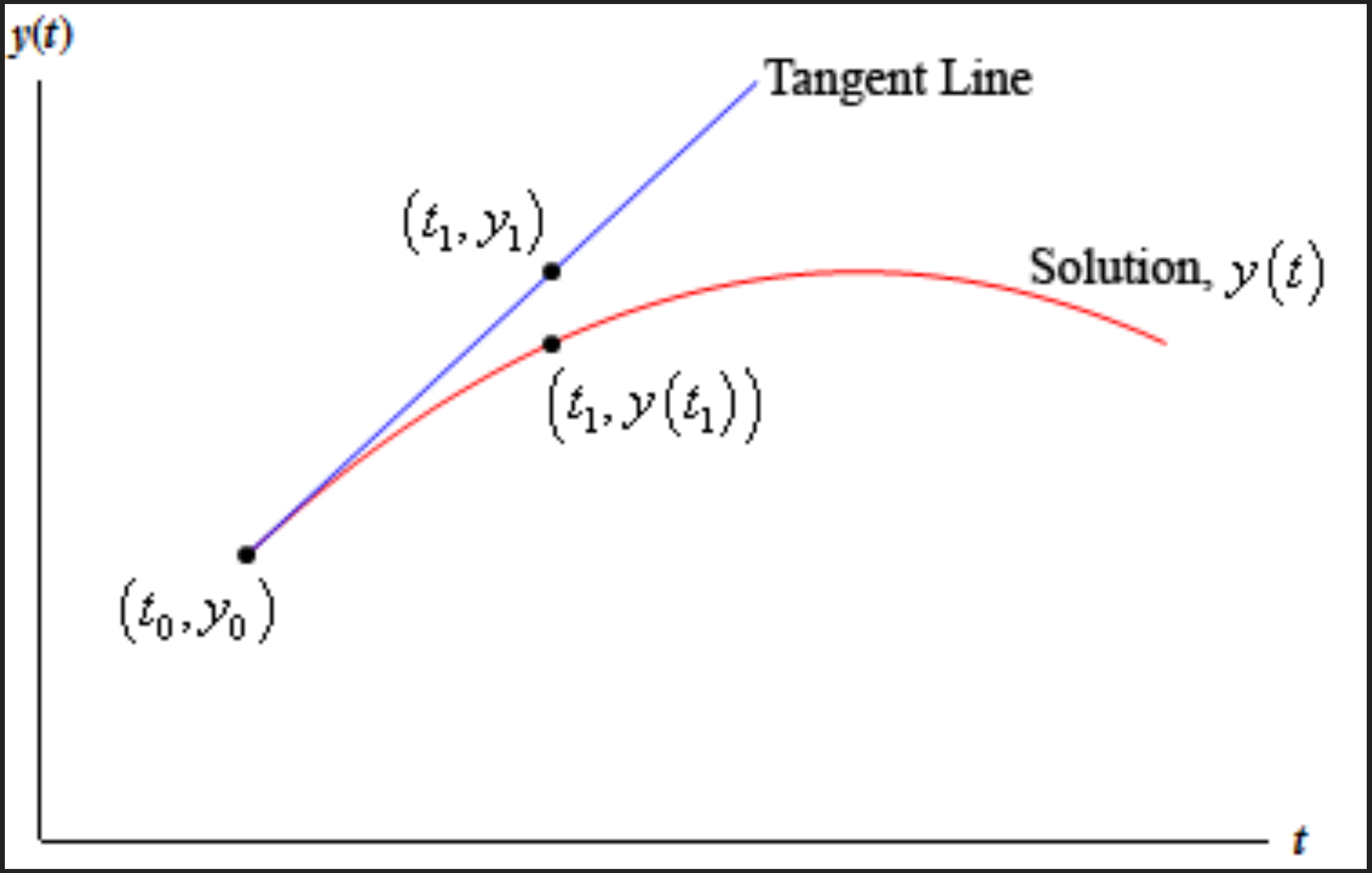
$$E \sim \frac{1}{2} dt^2 \cdot \frac{\partial^2 y}{\partial t^2}$$

DIFFERENTIAL EQUATIONS

- Explicit Euler method
- Calculate derivative (right hand side) at beginning of time step, multiply with dt, then add to value at beginning.

DIFFERENTIAL EQUATIONS

- Graphical representation of explicit Euler method

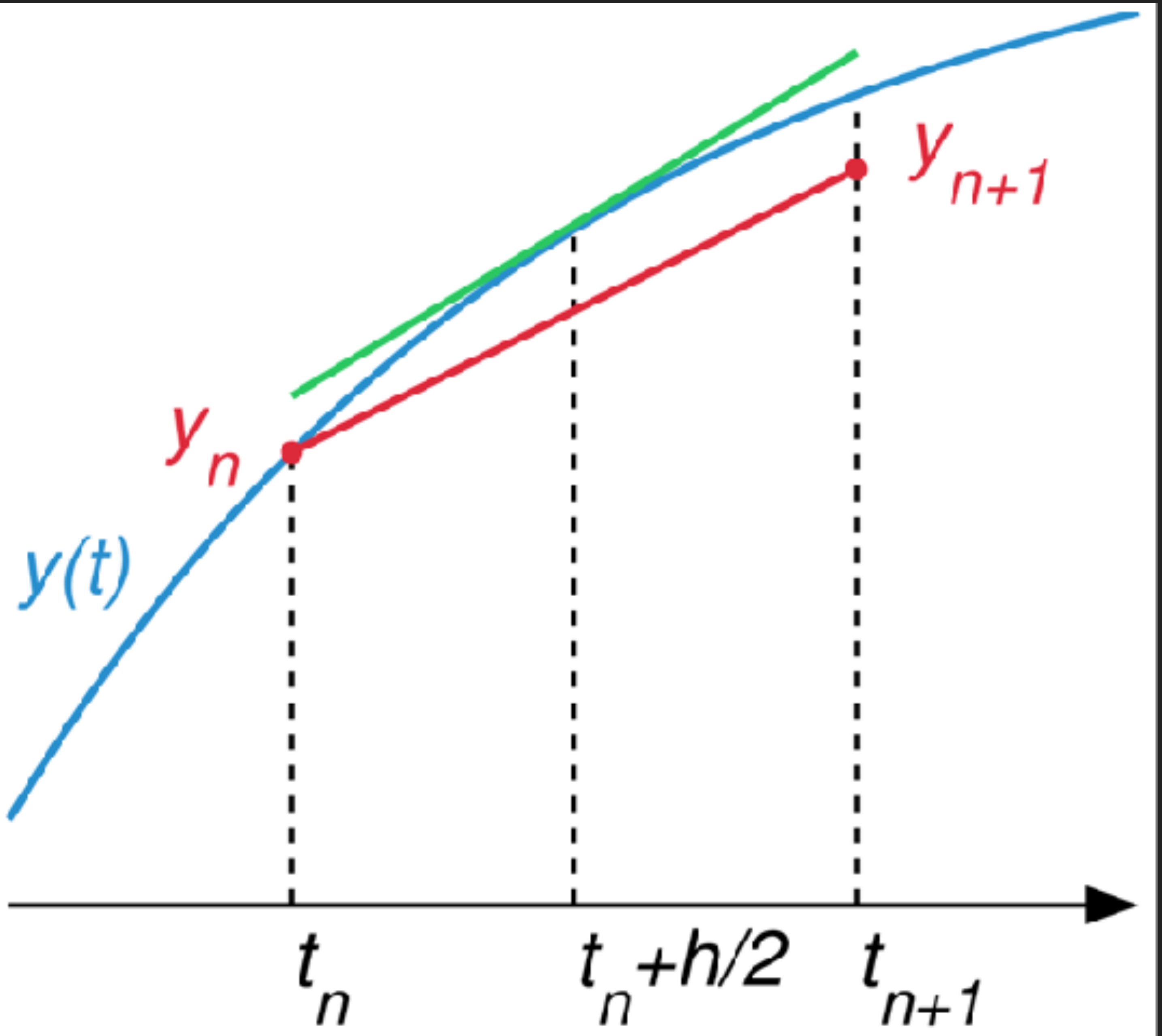


DIFFERENTIAL EQUATIONS

- Explicit Euler method is rarely ever used.
- This is because of the low order.
- Midpoint method is second order.
- Uses a sub-step, effectively combining two Euler steps

DIFFERENTIAL EQUATIONS

- Graphical representation of the midpoint method



DIFFERENTIAL EQUATIONS

- Higher order methods can be constructed.
- Often used: 4th or 5th order Runge Kutta

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	1/3	1/3	1/6

DIFFERENTIAL EQUATIONS

- N-body simulations are simulations of N interacting gravitational bodies
- Need to solve a 6*N dimensional coupled differential equation
- Difficult because we need very high precision over long timescales

DIFFERENTIAL EQUATIONS

- N-body simulations often use advanced integration methods
- Either very high order
- Or geometric/symplectic integrators which preserve some of the underlying symmetries of the problem.

PSCB57 - PROF. HANNO REIN

MONTE CARLO METHODS

MONTE CARLO METHODS

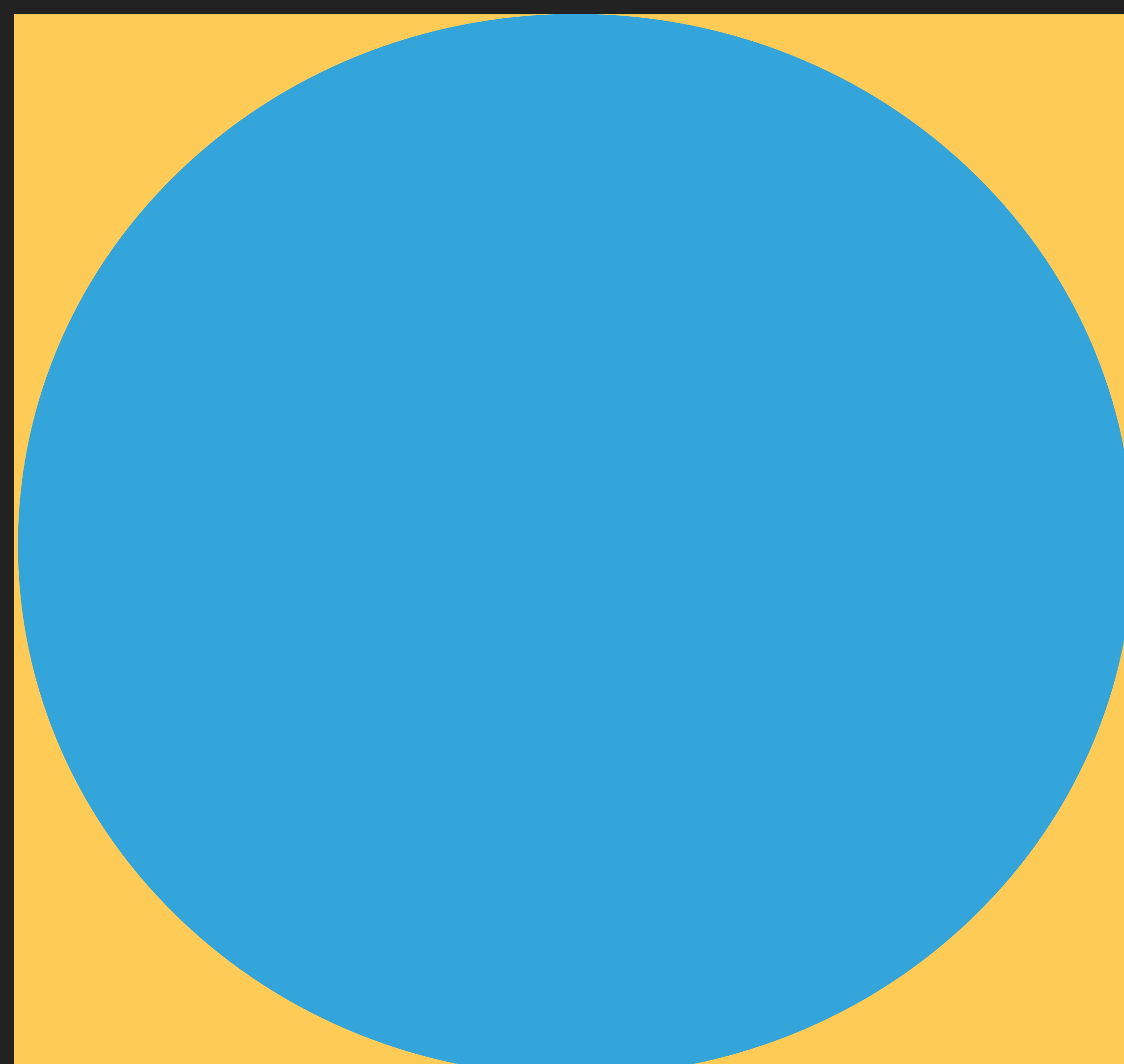
- A random number generator outputs *pseudo* random numbers on a computer
- Randomness is hard for the computer
- A good random number generator outputs uncorrelated, uniformly distributed random numbers that are hard to predict.

MONTE CARLO METHODS

- Random numbers are used in cryptography
- We use them to simplify numerical calculations!

MONTE CARLO METHODS

- Calculate pi using random numbers:



MONTE CARLO METHODS

- In general: use random numbers to calculate an integral:

PSCB57 - PROF. HANNO REIN

BAYES' THEOREM

BAYES' THEOREM

- Very important statical tool.
- Derivation is very simple!

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

BAYES' THEOREM

- Makes use of conditional probability.
- The syntax P(A|B) means the probability that event A is true given event B is true.
- Can use Bayes' theorem to inver the equation to get P(B|A)

BAYES' THEOREM

- Can apply this to simple statical problems such as the *Cookie problem* or the *Monty Hall* problem.
- Make sure you know how we did those calculations!

BAYES' THEOREM

- Diachronic interpretation
- Used in relationship to testing a hypothesis in science using data
- Terms in Bayes' Theorem have names. Know them and understand their meaning!

BAYES' THEOREM

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- $P(A|B)$ = Posterior
- $P(B|A)$ = Likelihood
- $P(A)$ = Prior
- $P(B)$ = normalization constant

BAYES' THEOREM

- Using Bayes' theorem is related to solving a high dimensional integral.
- Can use Monte Carlo Methods to do that.
- We are randomly *sampling the posterior.*

PSCB57 - PROF. HANNO REIN

COMING UP…

TUTORIAL TOMORROW

- Come to get help with the project.
- Run the presentation by me, if you want.
- Also can ask questions about any other material from the course.

PROJECT REPORT

- Due on December 4th
- Can hand it in in paper form or submit it online.

PROJECT PRESENTATIONS

- Will happen on December 4th
- All project members need to be present, but not all need to take part in the presentation
- Make sure your computer works if you plan to use the projector
- Make sure you do not run over