Lecture 03: Functions, interpolation, integration, differentiation

Sergei V. Kalinin

Reminder

- All homeworks, midterms, and finals will be Colabs
- Please submit to sergei2vk@gmail.com
- You will have a week for each
- Late submissions are generally not a problem, but avoid accumulation
- Semester deadlines are final
- Don’t hesitate to ask for help and additional explanations!

This and that

- Any comments re first homework?

- Final project parameters:
  - Can be alternative to the final
  - Should meaningfully use optimization, Gaussian processes, or causal methods
  - Be connected to your own research
  - Ideally will become publication

- Some possibilities:
  - Explore molecular space towards candidates with specific functionality
  - Building workflow for materials optimization in automated synthesis
  - Causal analysis of the perovskite data base
  - ....

Functions and spaces

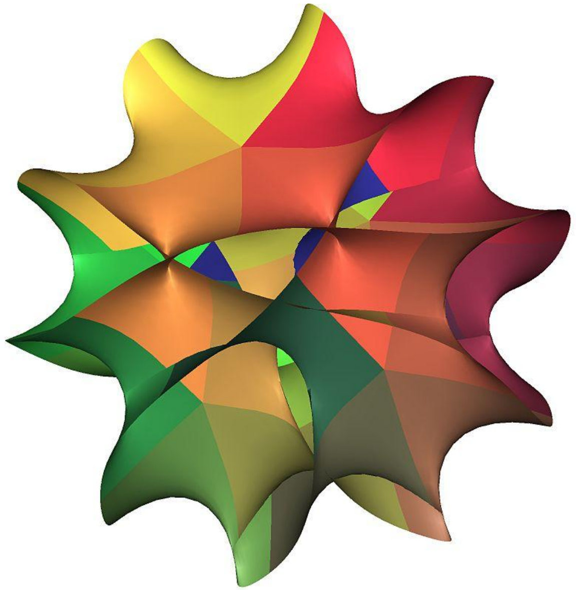
- Our biggest strength – and weakness – is the intuition developed for the 3D Euclidean spaces

- Mathematicians has created large set of alternative spaces based on possible functional relationships and properties of objects
  - All real number
  - All integral numbers
  - All functions over unit interval
  - … and many more

- In physical sciences, only a small subset of these is practically useful
- … but these general principles are important to know!

What is space?

A space is a set with some added structure. It consists of selected mathematical objects that are treated as points, and selected relationships between these points.



The nature of the points can vary:

- elements of a set,
- functions on another space, or
- subspaces of another space.

The relationships between objects define the nature of the space.

Isomorphic spaces are considered identical, where an isomorphism between two spaces is a one-to-one correspondence between their points that preserves the relationships.

https://en.wikipedia.org/wiki/Space_(mathematics)

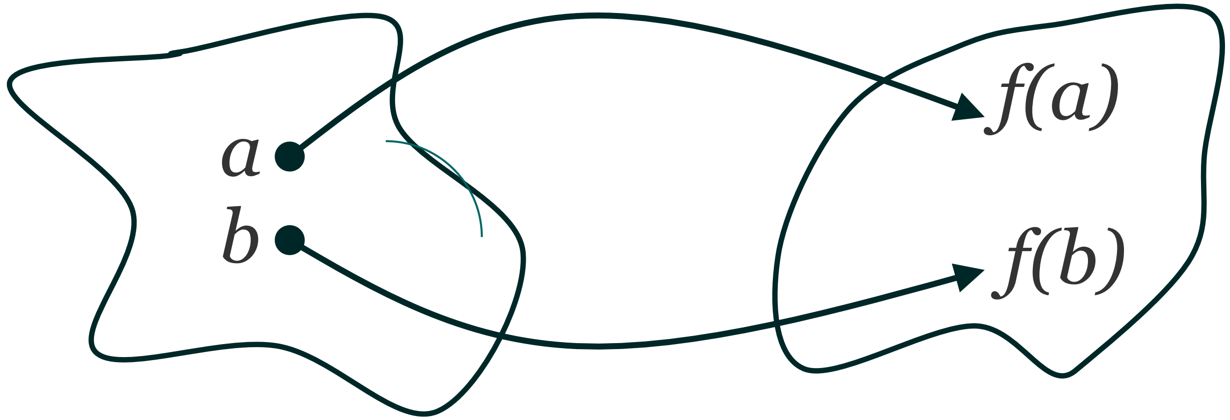
https://en.wikipedia.org/wiki/Calabi%E2%80%93Yau_manifold

What is function?

In mathematics, a function from a set X to a set Y assigns to each element of X exactly one element of Y. The set X is called the domain of the function, and the set Y is called the codomain of the function.

We can define function by:

- Listing values
- Algebraic formulae
- Specified algorithm.
- ... and so on



Some function definitions:

- f is injective (or one-to-one) if $f(a) \neq f(b)$ for any two different elements a and b of X
- f is surjective if its range X equals its codomain Y , i.e. for each element y of the codomain, there exists some element x of the domain such that $f(x)=y$
- f is bijective (or a one-to-one correspondence) if it is both injective and surjective. That is, f is bijective if, for any y ∈ Y, the preimage $f^{-1}(y)$ contains exactly one element.

https://en.wikipedia.org/wiki/Function_(mathematics)

What should we look at practically?

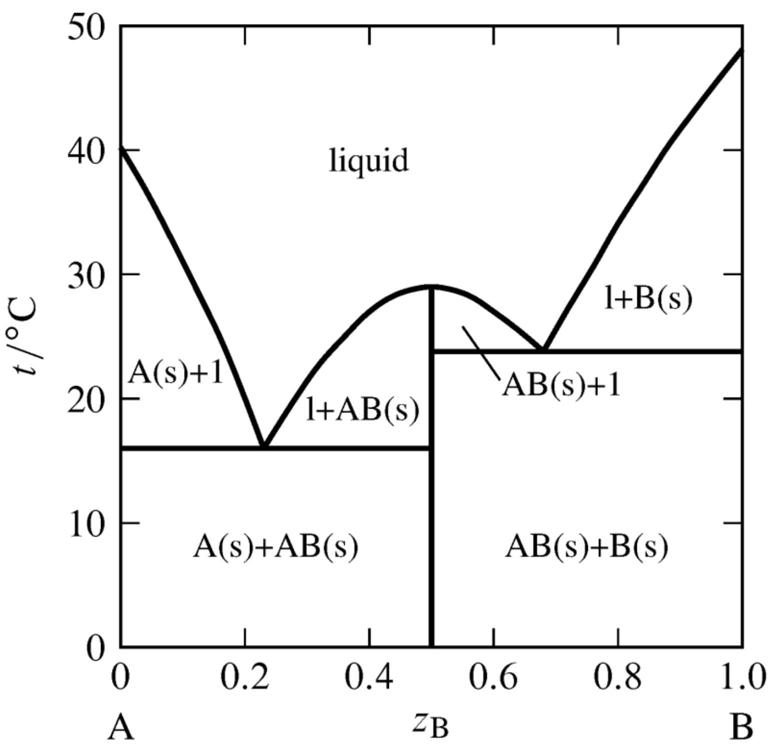
These are important:

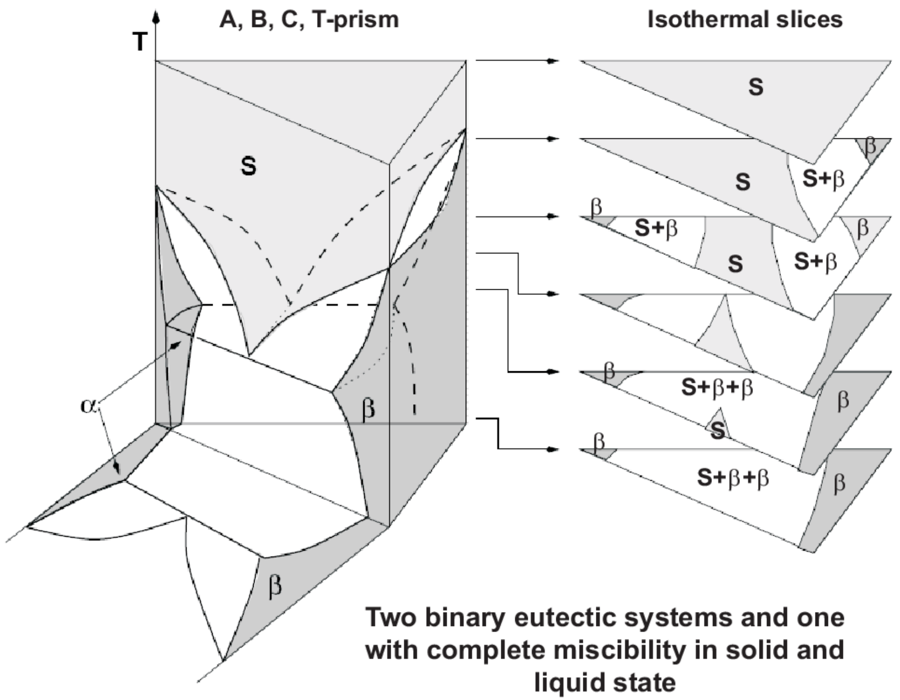
- Dimensionality
- Metrics (Distance Function)
- Continuity and Differentiability

These may be important:

- Symmetry
- Topological Properties
- Inner Product (Hilbert Spaces)
- Manifold Structure

Phase diagrams





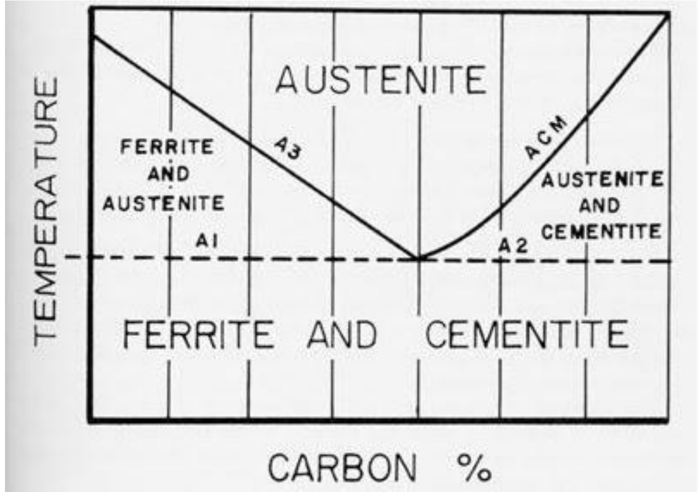
Two binary eutectic systems and one with complete miscibility in solid and liquid state

- What is the dimensionality of the parameter space?
- Is it differentiable?
- What is the function?
- Is it continuous?
- Is it differentiable

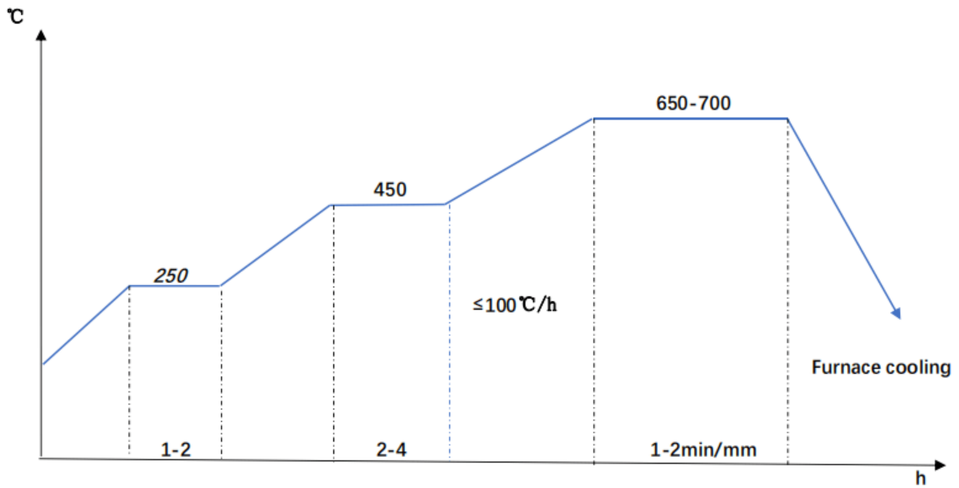
[13.2: Phase Diagrams- Binary Systems - Chemistry LibreTexts]

https://www.tf.uni-kiel.de/matwis/amat/td_kin_ii/kap_1/backbone/r_se17.html

Processing



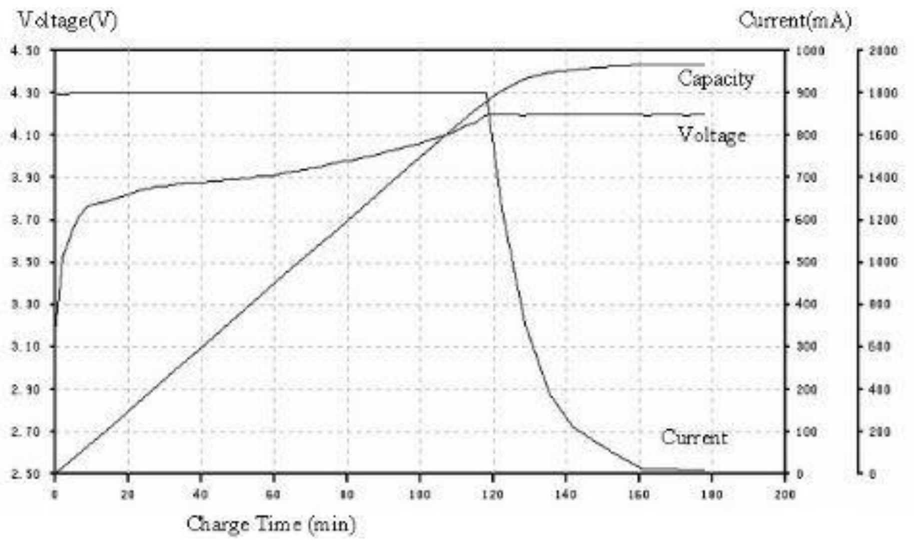
AUSTENITE
FERRITE AND AUSTENITE
A3
ACM
AUSTENITE AND CEMENTITE
A1
A2
FERRITE AND CEMENTITE
TEMPERATURE
CARBON %

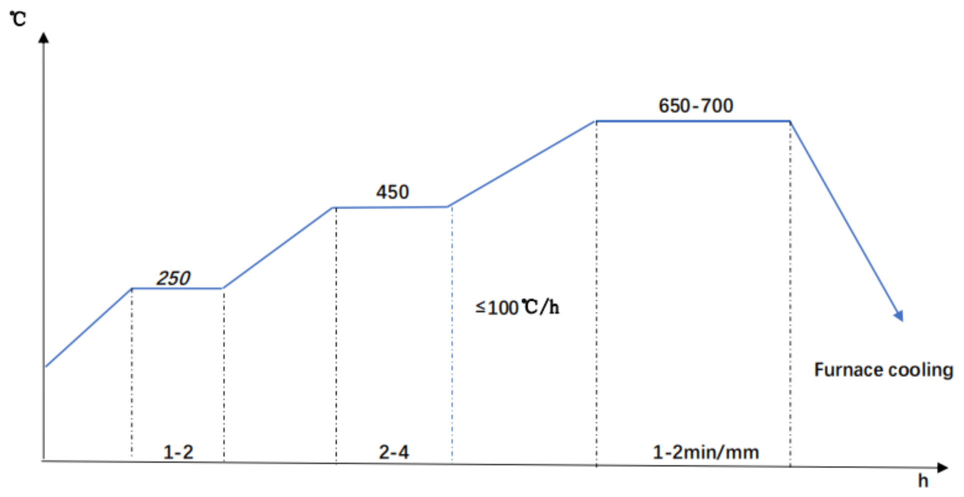


℃
250
450
650-700
≤100℃/h
Furnace cooling
1-2
2-4
1-2min/mm
h

Processing

- Making steel – can be complicated and took a lot of time optimize
- Battery charging – fairly simple now, but obvious economic impact
- Annealing hybrid perovskite thin films
- Poling ferroelectric materials
- … and so on

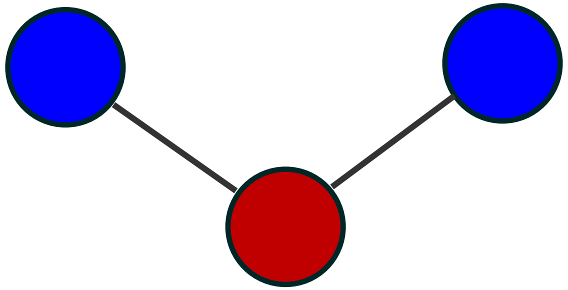




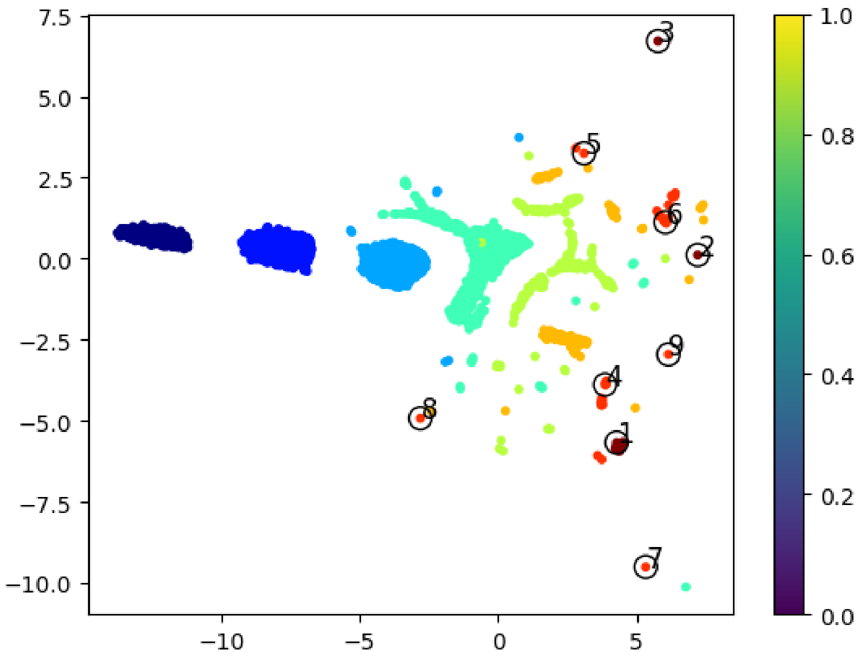
- What is the dimensionality of the parameter space?
- Is it differentiable?
- What is the function?
- Is it continuous?
- Is it differentiable

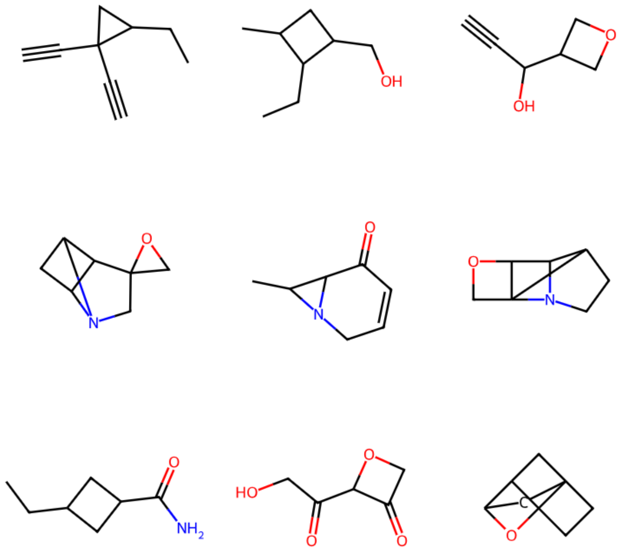
Molecular spaces

Geometric space



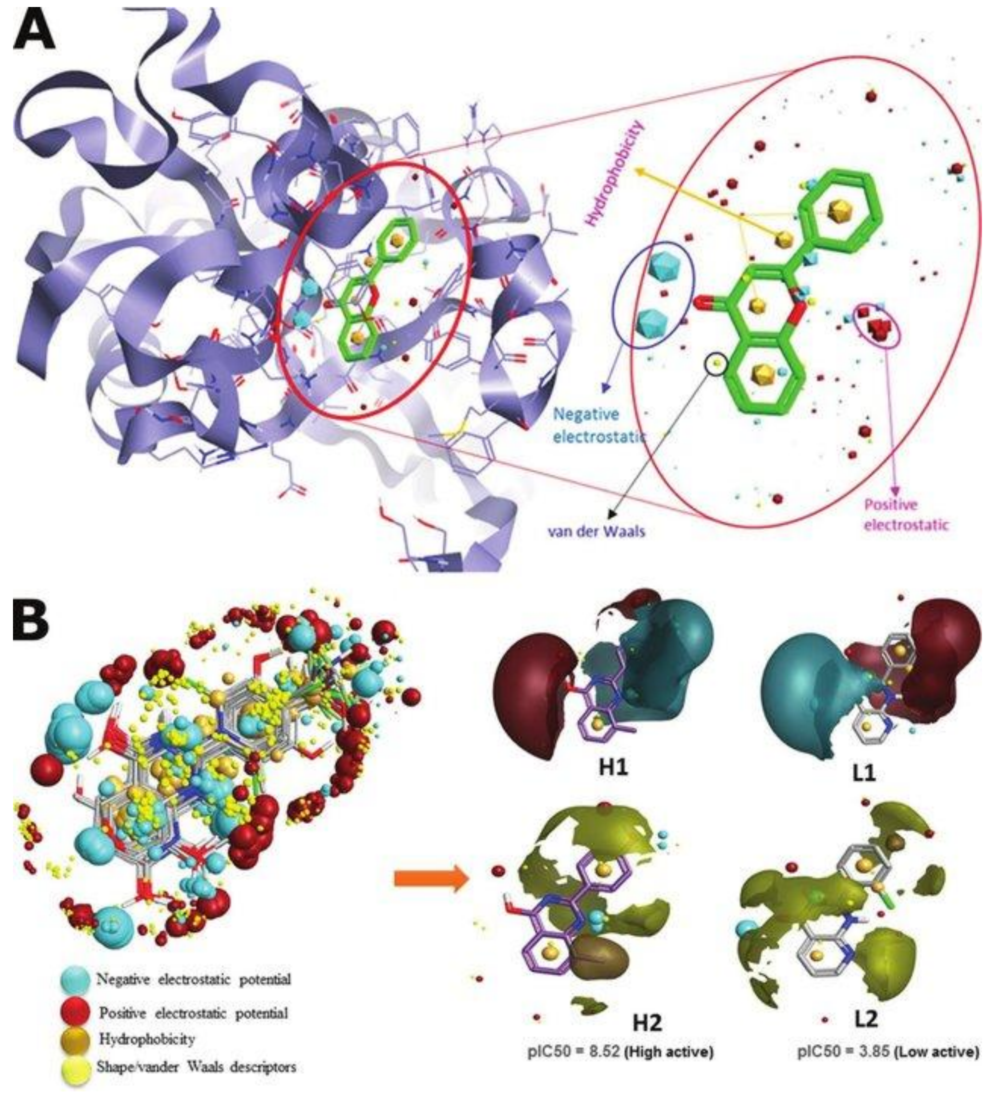
Chemical space





- What is the dimensionality of the parameter space?
- Is it differentiable?
- What is the function?
- Is it continuous?
- Is it differentiable

Structure–property relationships

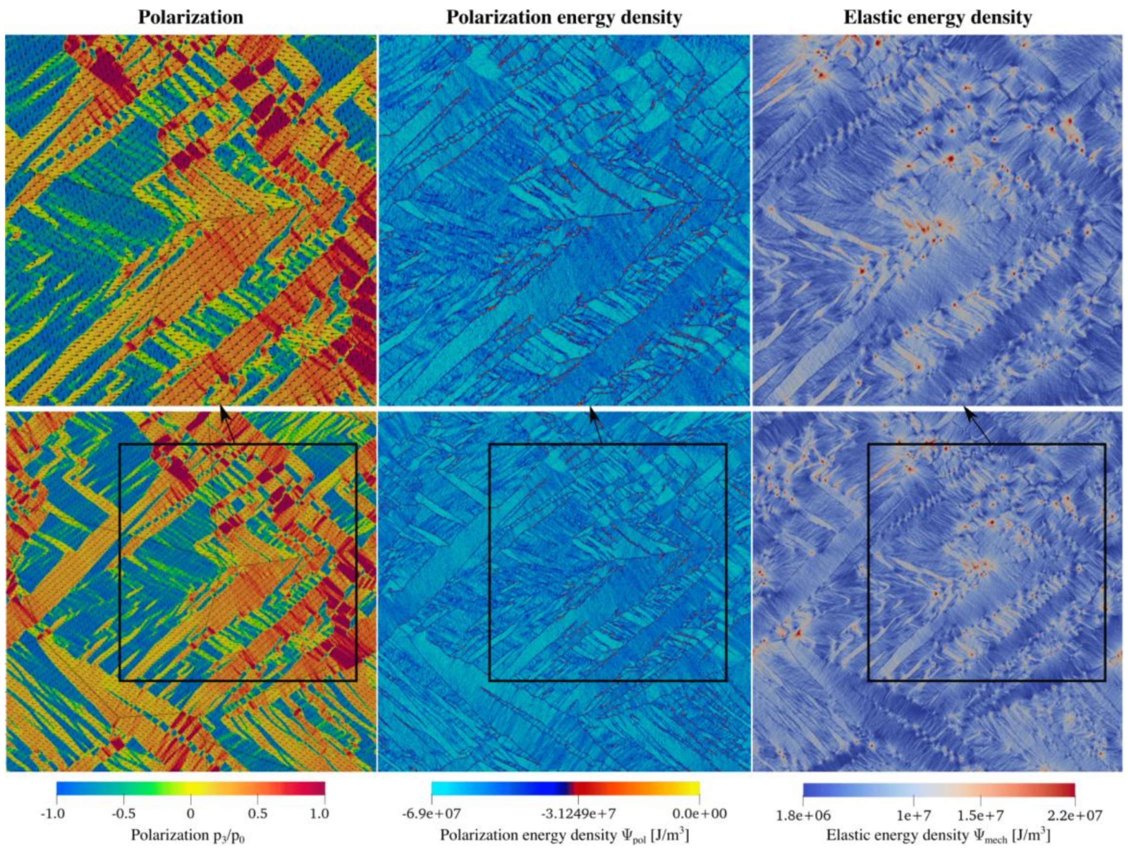


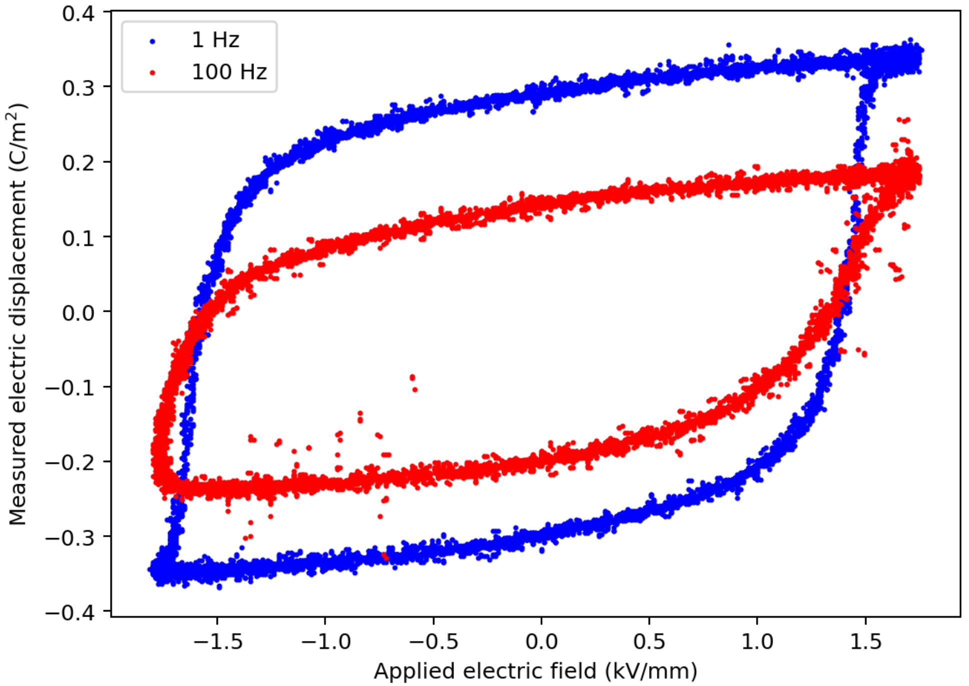
- What is the dimensionality of the parameter space?
- Is it differentiable?
- What is the function?
- Is it continuous?
- Is it differentiable

Hint: activity cliffs in quantitative structure-activity relations (QSAR)

https://www.researchgate.net/publication/342718908_Artificial_Intelligence_and_Machine_Learning_in_Computational_Nanotoxicology_Unlocking_and_Empowering_Nanomedicine/figures?lo=1

Structure–property relationships





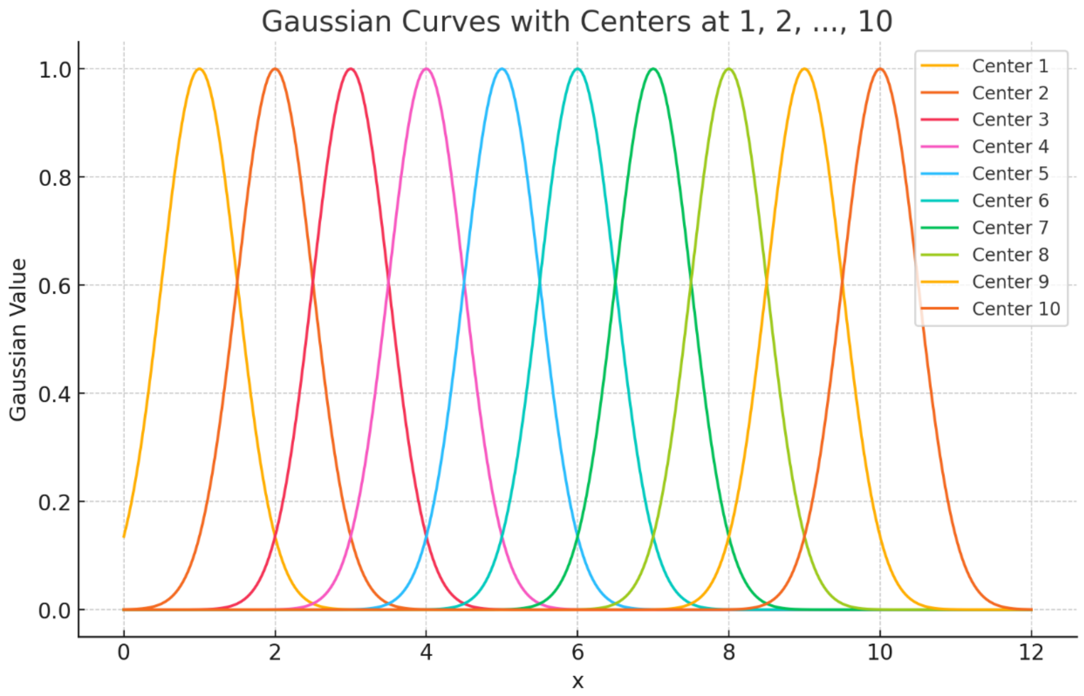
- What is the dimensionality of the parameter space?
- Is it differentiable?
- What is the function?
- Is it continuous?
- Is it differentiable

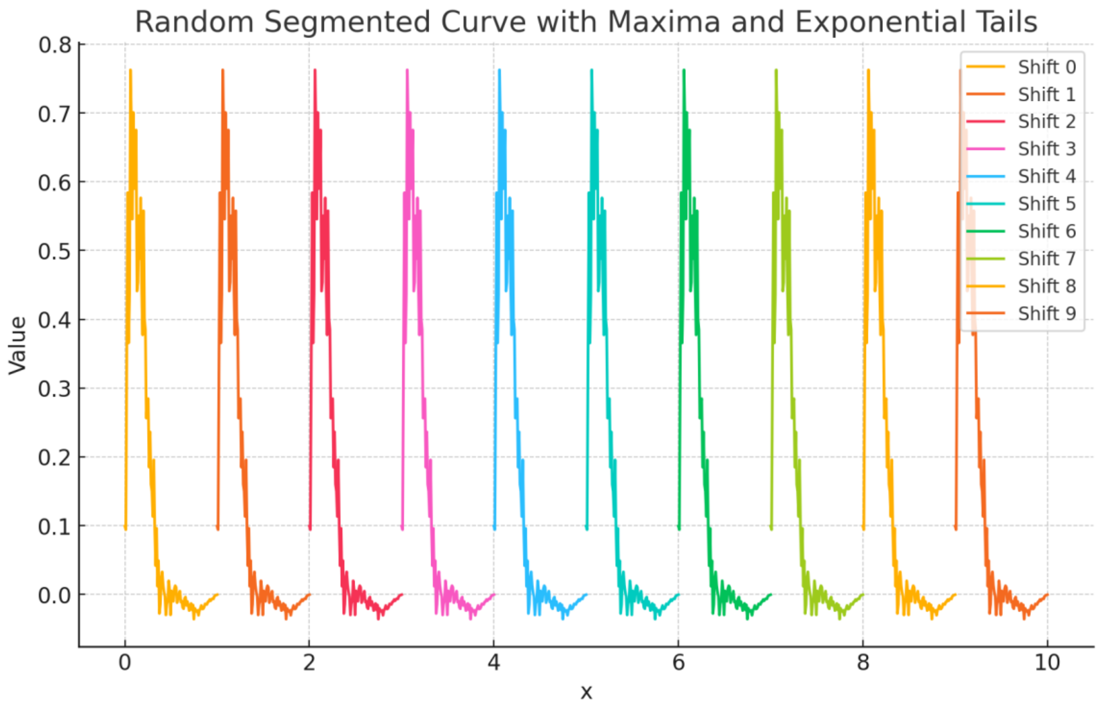
For each of these scenarios

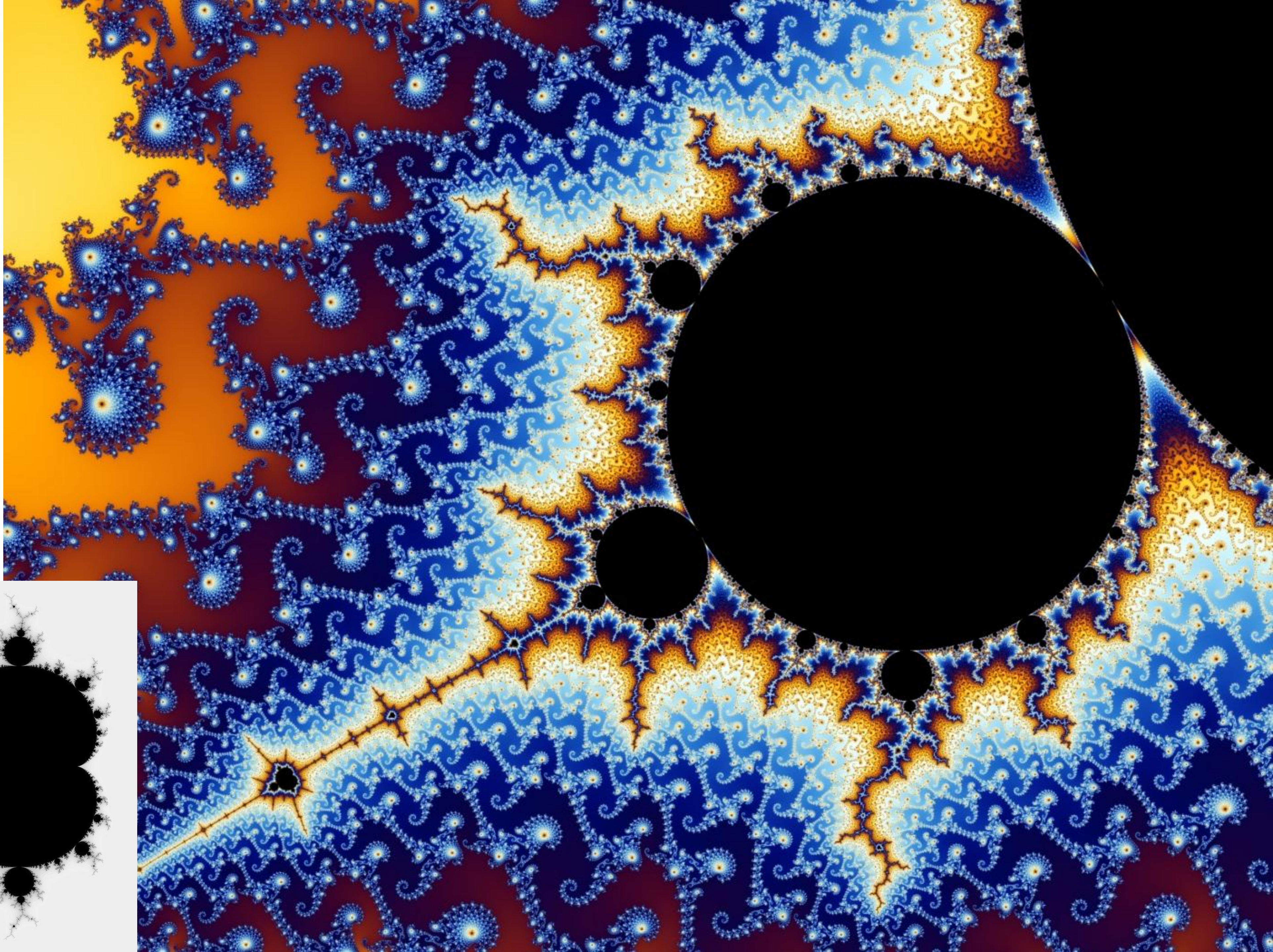
- Can we interpolate?
- Can we differentiate?
- Can we integrate?

  - Phase diagram
  - Process trajectories
  - QASR
  - Hysteresis loops

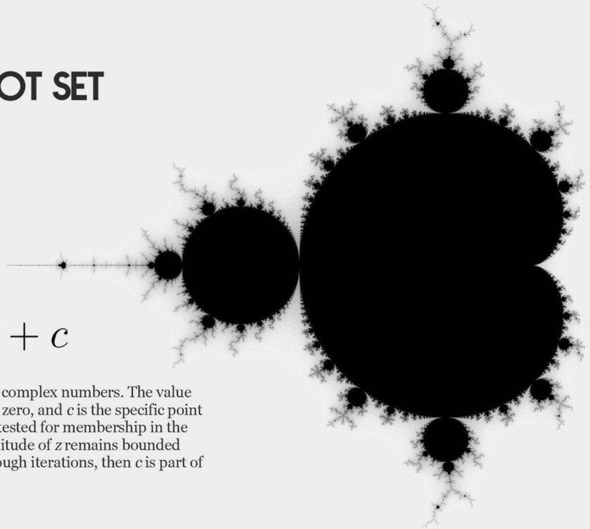
Complexity - trivial







MANDELBROT SET



$$f_c(z) = z^2 + c$$

In this equation, z and c are complex numbers. The value of z is iterated starting from zero, and c is the specific point in the complex plane being tested for membership in the Mandelbrot set. If the magnitude of z remains bounded *(does not go to infinity)* through iterations, then c is part of the Mandelbrot set.

Off to Colab!

Interpolation

Sometimes we know the value of some function f(x) at a discrete set of points x_0, $x_{1,}$ …, x_N, but we do not know how to (easily) calculate its value at arbitrary x

Examples:

- Physical measurements
- Long numerical calculations

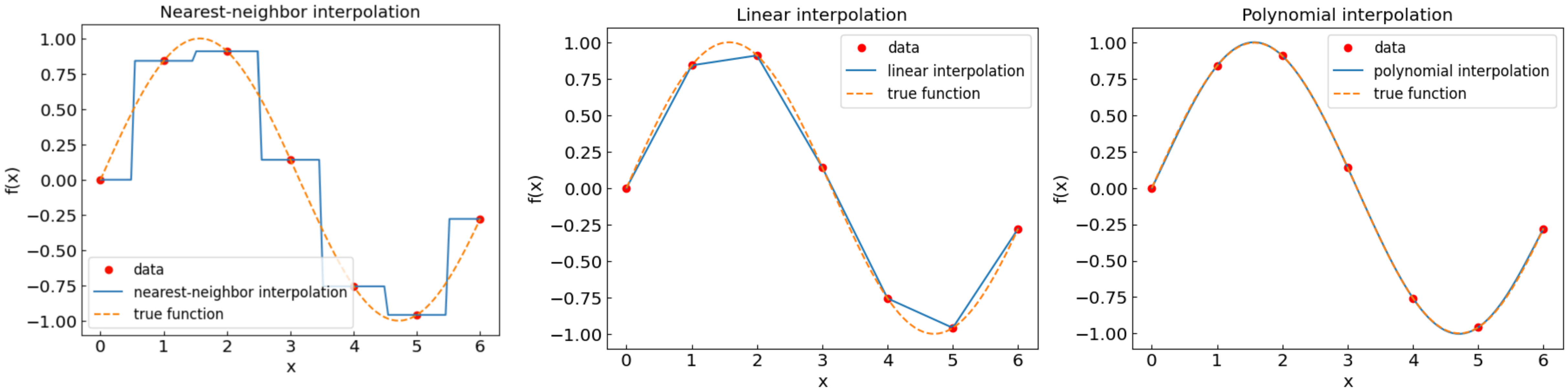
Interpolation is a method to generate new data points from existing data points consisting of two steps:

1. Fitting the interpolating function to data points
2. Evaluating the interpolating function at a target point x

References: Chapter 3 of *Numerical Recipes Third Edition* by W.H. Press et al.

From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Interpolation methods



From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation (Lagrange form)

Theorem: There exists a *unique* polynomial of order *n* that interpolates through n+1 data points (x_0,y_0), (x_1,y_1), …, (x_n,y_n)

How to build such a polynomial?

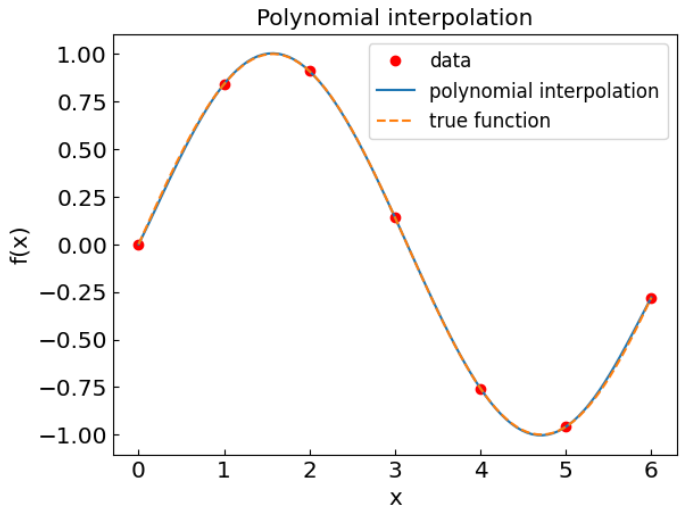
Consider *Lagrange basis functions*:

$$L_{n,j}(x) = \prod_{k \neq j} \frac{x - x_k}{x_j - x_k}.$$

Easy to see that for $x=x_k$ one has $L_{n,j}(x_k) = \delta_{kj}.$

Therefore:

$$f(x) \approx p(x) = \sum_{j=0}^{n} y_j L_{n,j}(x)$$

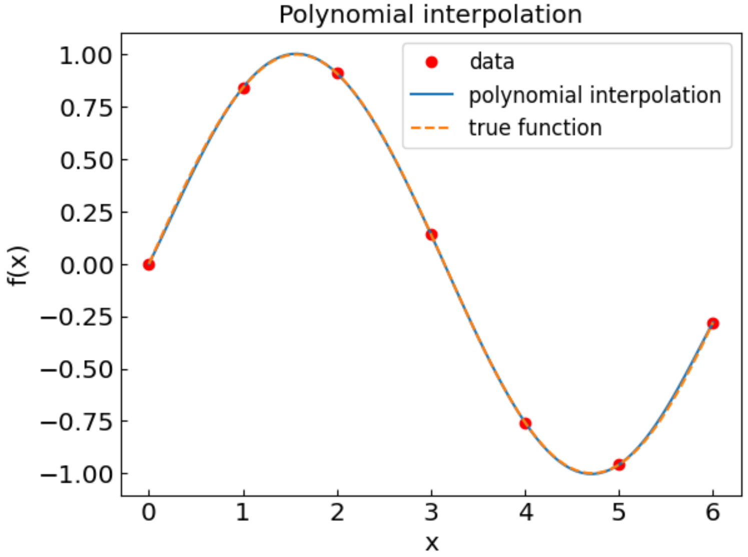


From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation

For our example f(x) = sin(x)

x	sin(x)
0	0.
1	0.841471
2	0.9092974
3	0.14112
4	-0.7568025
5	-0.9589243
6	-0.2794155



one obtains

$$p(x) = -0.0001521x^6 - 0.003130x^5 + 0.07321x^4 - 0.3577x^3 + 0.2255x^2 + 0.9038x.$$

In practice, the Lagrange form is more stable with respect to round-off errors

From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

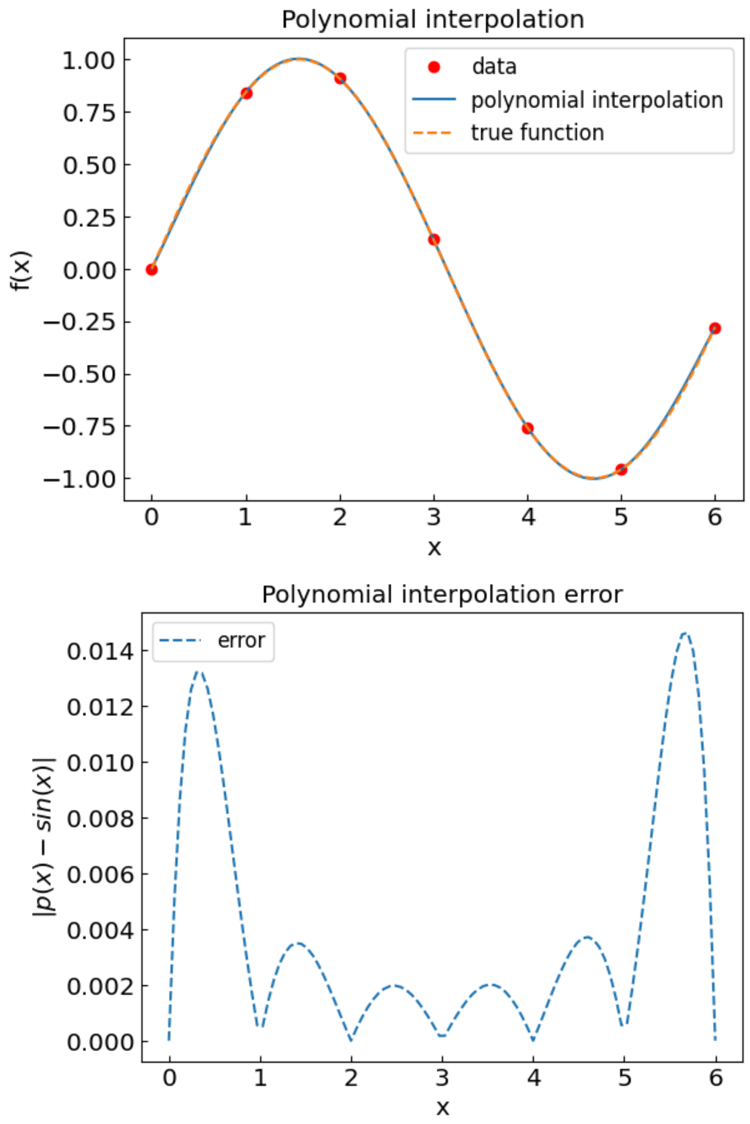
Polynomial interpolation

In Python:

```
def Lnj(x,n,j,xdata):
    """Lagrange basis function."""
    ret = 1.
    for k in range(0, len(xdata)):
        if (k != j):
            ret *= (x - xdata[k]) / (xdata[j] - xdata[k])
    return ret


def f_poly_int(x, xdata, fdata):
    """Returns the polynomial interpolation of a function at point x.
    xdata and ydata are the data points used in interpolation."""
    ret = 0.
    n = len(xdata) - 1
    for j in range(0, n+1):
        ret += fdata[j] * Lnj(x,n,j,xdata)
    return ret

xpoly = np.linspace(0,6,100)
fpoly = [f_poly_int(xin,xdat,fdat) for xin in xpoly]
```



From the course by Volodymyr Vovchenko, https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Errors and artefacts

- Truncation errors

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{n+1}\prod_{i=0}^{n}(x - x_i)$$

- Round-off errors
  - Especially for high-order polynomials

Truncation errors can be a problem if

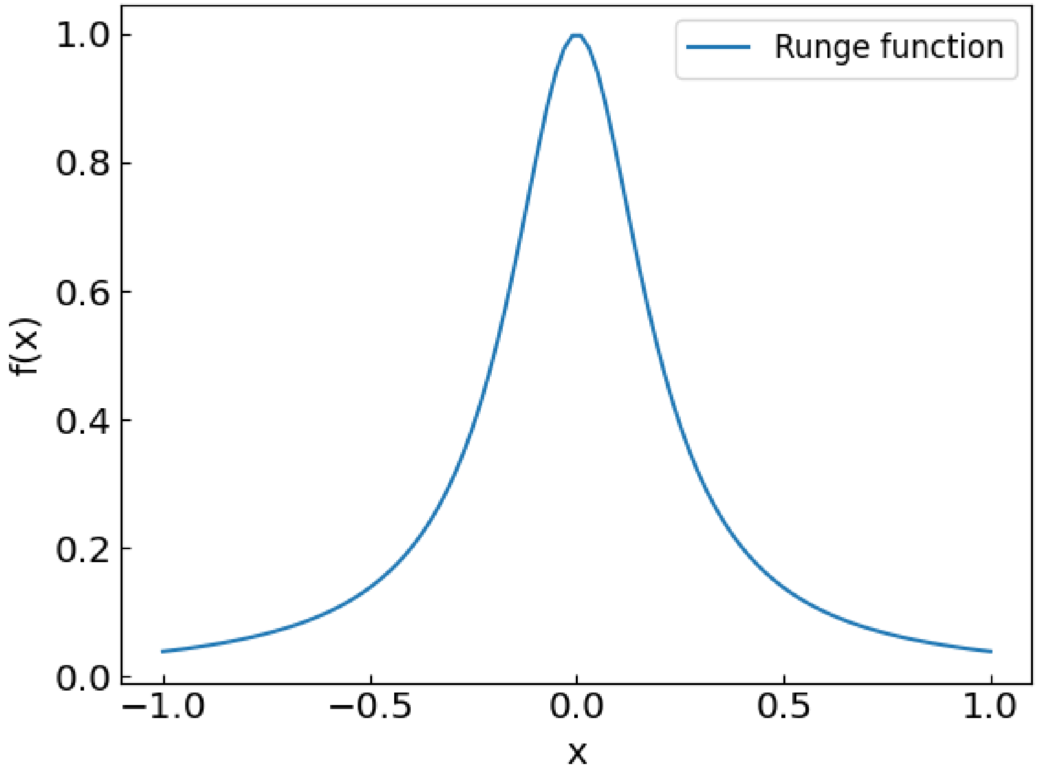
- High-order derivatives $f^{(n+1)}(x)$ of the function are significant
- The choice of nodes leads to a large value of the product factor

Runge phenomenon: Oscillation at the edges of the interval which gets *worse* as the interpolation order is increased

From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Runge phenomenon

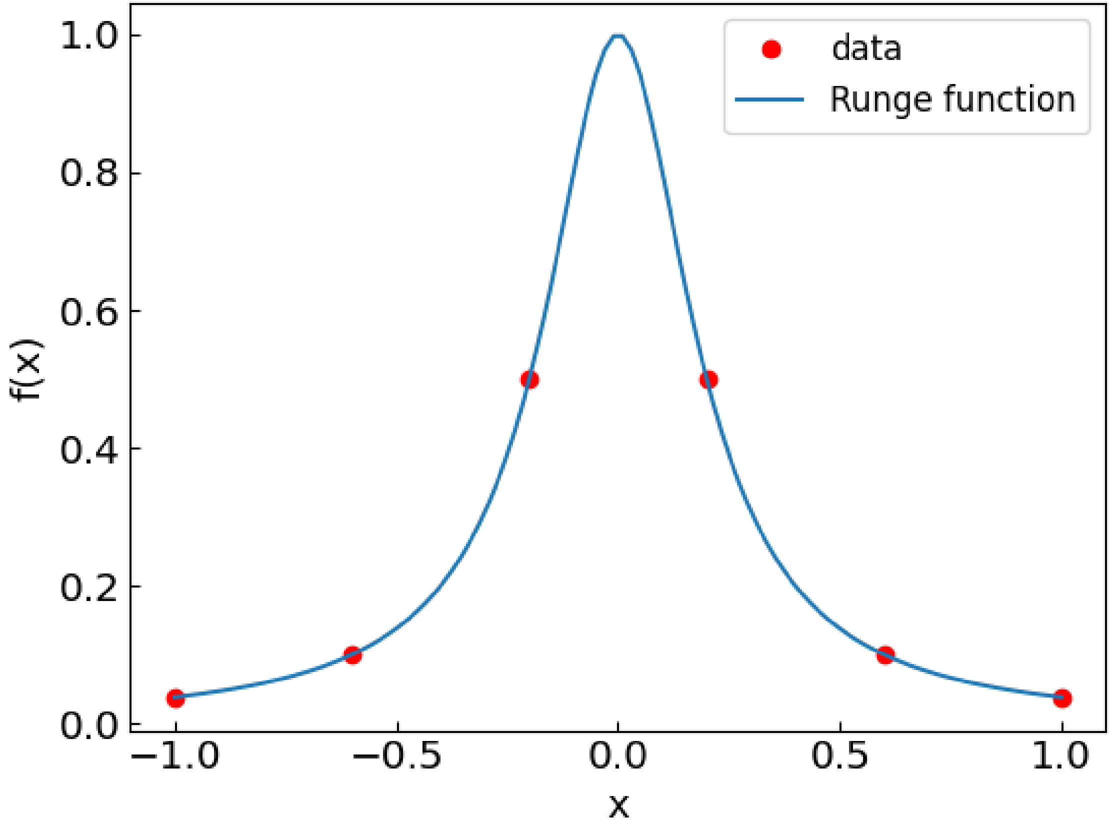
Consider the Runge function: $$f(x) = \frac{1}{1+25x^2}$$



Let us do polynomial interpolation using equidistant nodes

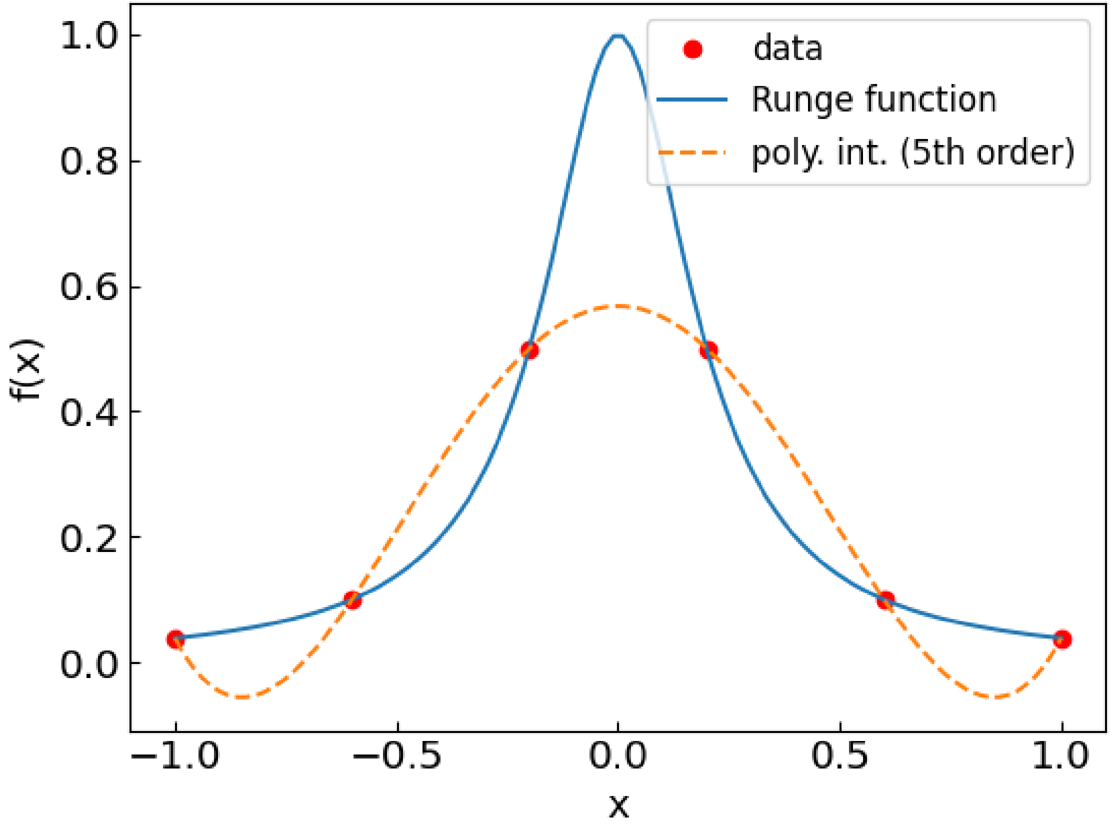
From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Runge phenomenon



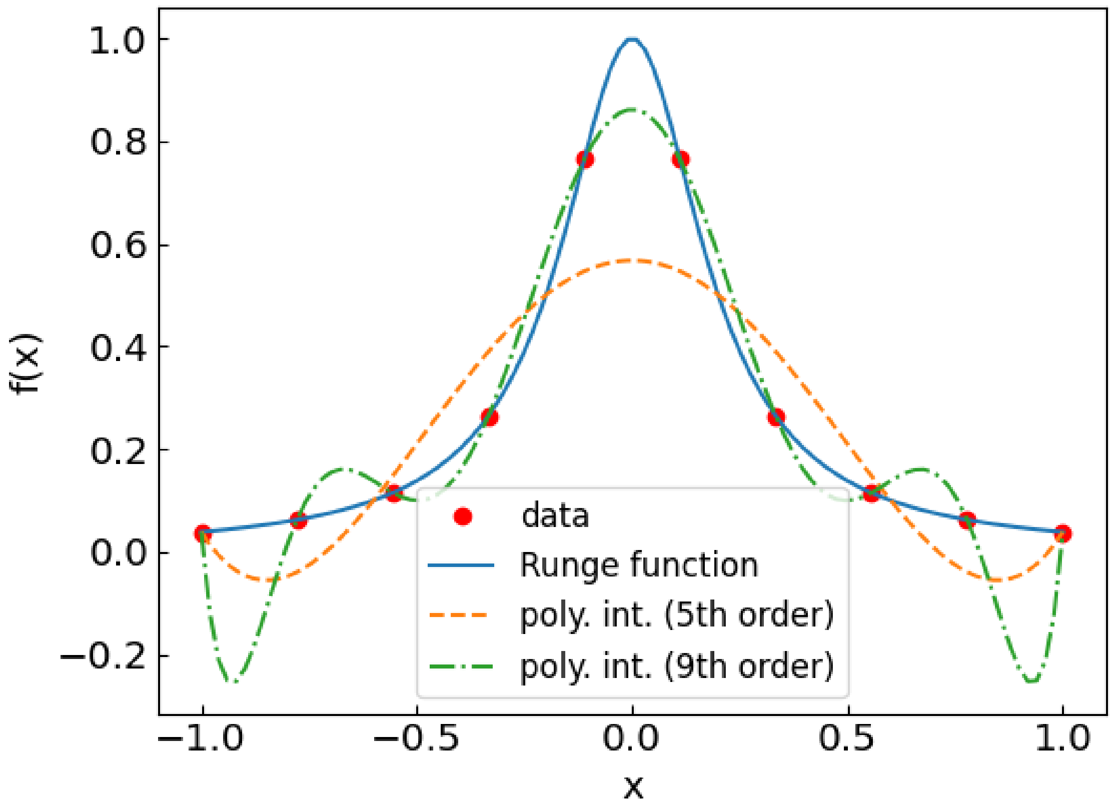
From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Runge phenomenon



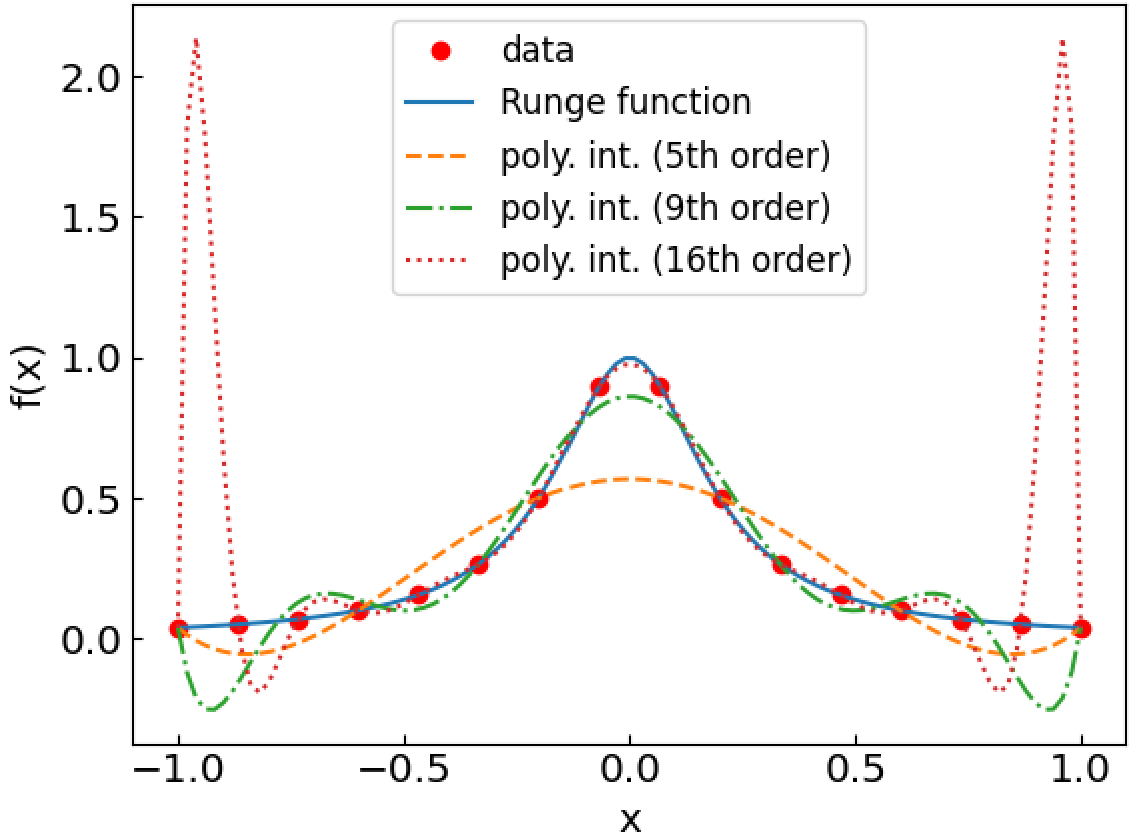
From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Runge phenomenon



From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Runge phenomenon



We have a real problem at the edges!

From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Chebyshev nodes

Recall the truncation error

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{n+1} \prod_{i=0}^{n} (x - x_i)$$

So far, we used the equidistant nodes:

$$x_k = a + hk, \qquad k = 0, \dots, n, \qquad h = (b-a)/n$$

Can we choose the nodes x_i differently to minimize the product factor? *Yes!*

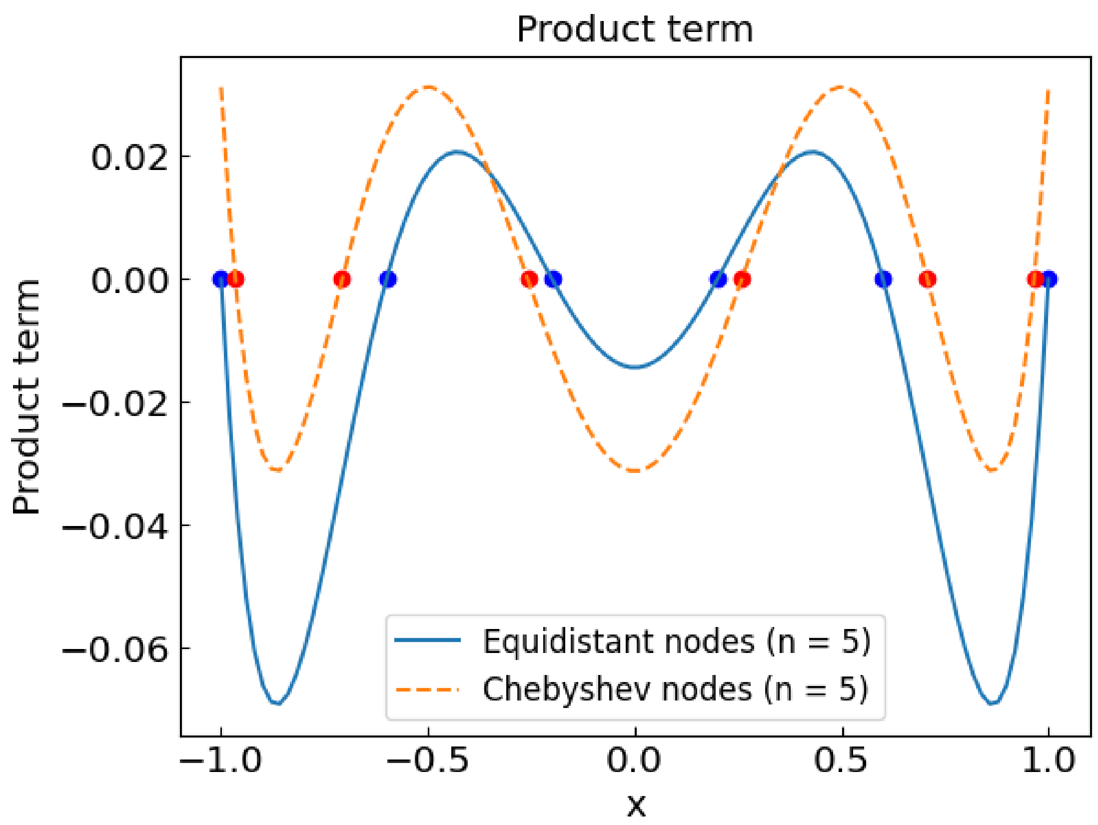
Chebyshev nodes:

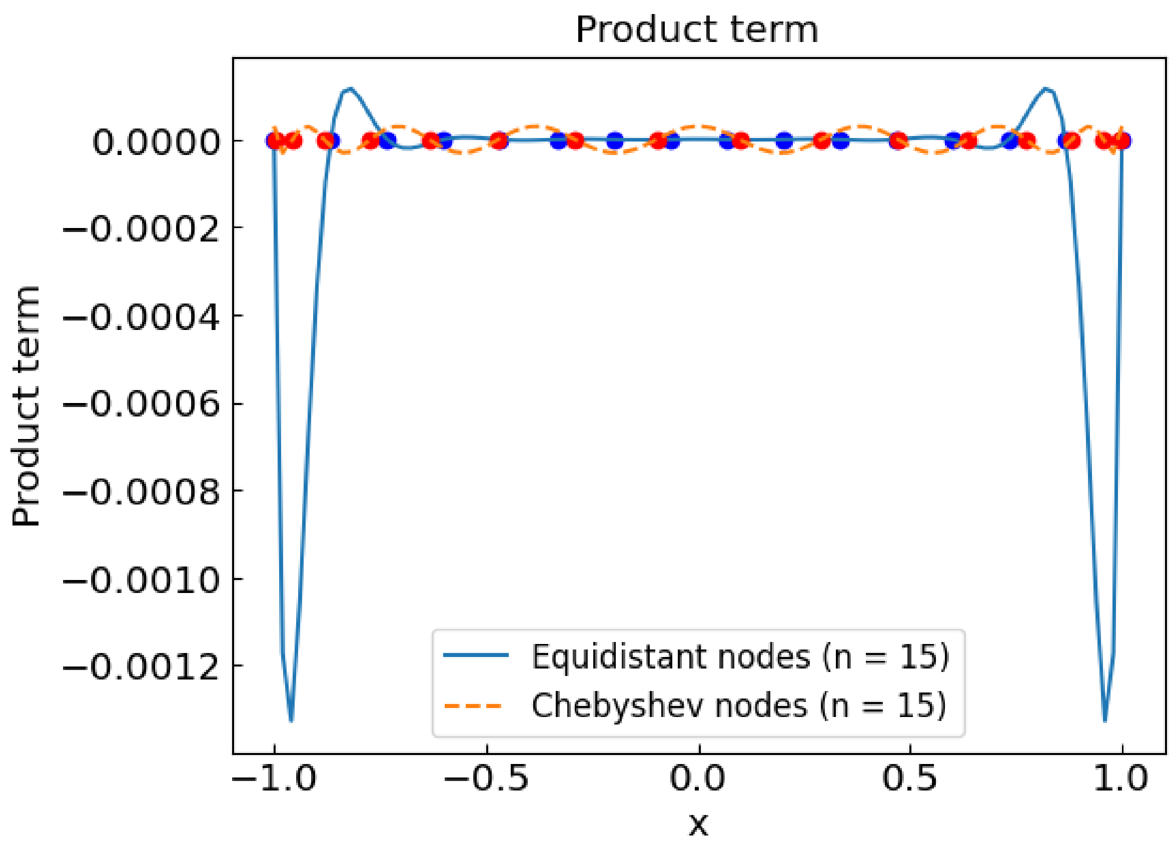
$$x_k = \frac{a+b}{2} + \frac{b-a}{2} \cos\left(\frac{2k+1}{2n+2}\pi\right), \qquad k = 0, \dots, n,$$

From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Equidistant vs Chebyshev nodes

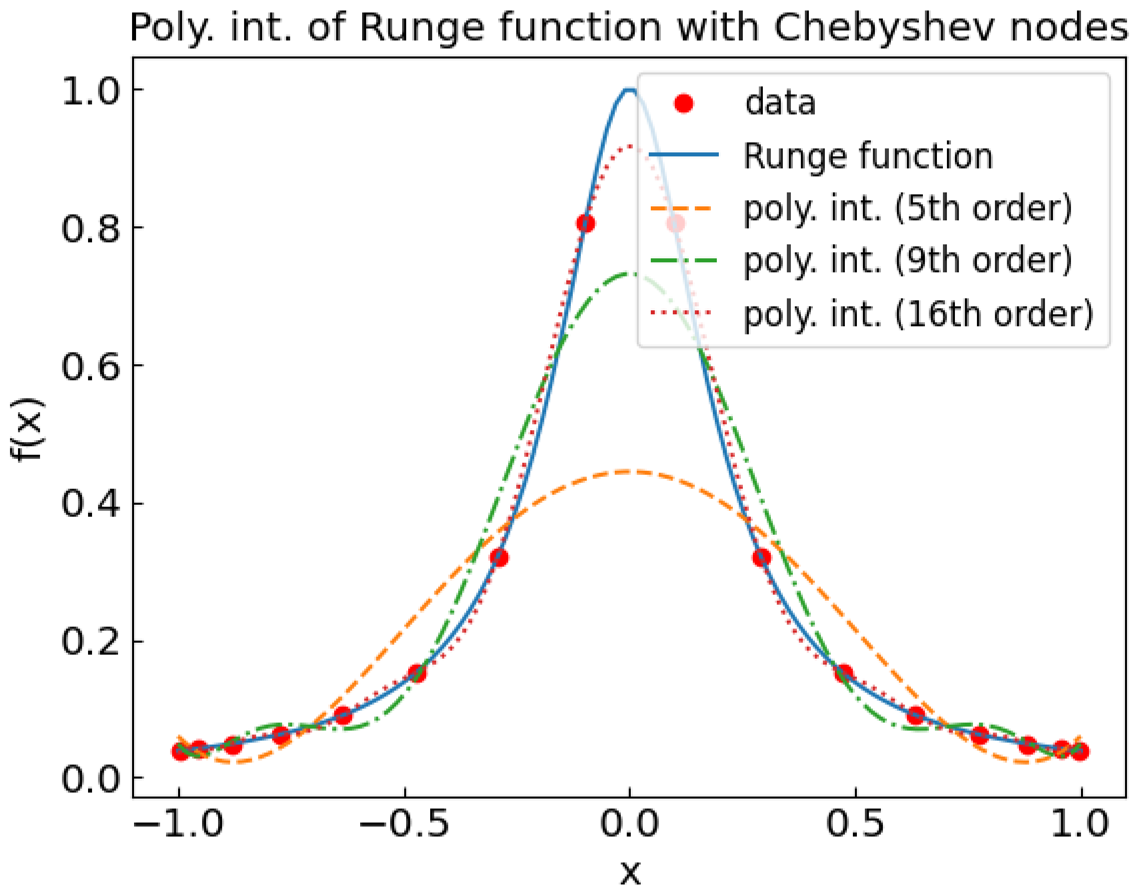
Plot $\prod_{i=0}^{n}(x - x_i)$ as a function of x for different number of nodes n on a (-1,1) interval





From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Back to the Runge function: Chebyshev nodes



From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

Polynomial interpolation: Summary

Advantages:

- Generally more accurate than the linear interpolation
- Derivatives are continuous
- Can be used for numerical integration and differential equations

Disadvantages:

- Implementation not so simple
- Artefacts possible (such as large oscillations between nodes)
- Polynomials of large order susceptible to round-off errors
- Not easily generalized to multiple dimensions

From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

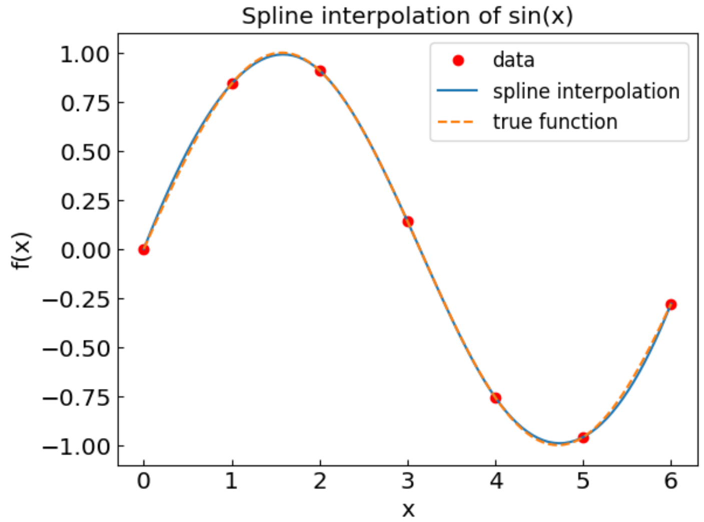
Spline interpolation

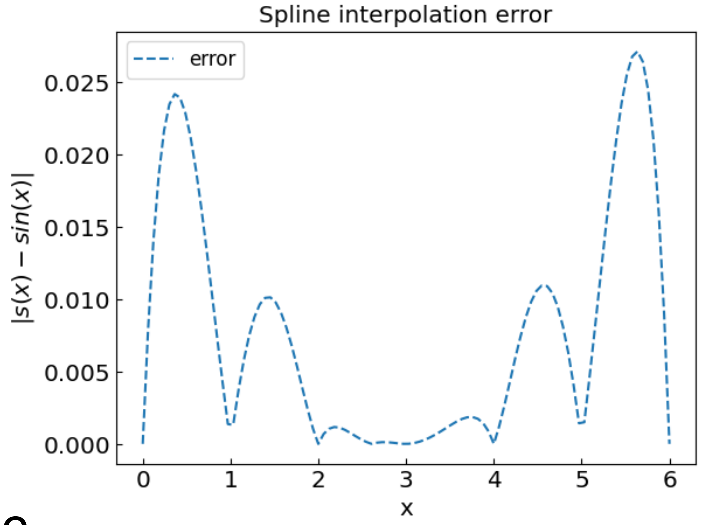
Connect each pair of nodes by a cubic polynomial

$$q_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i, \qquad x \in (x_i, x_{i+1})$$

4n coefficients a_i, b_i, c_i, d_i determined from

- n+1 data points
- continuity of first and second derivatives at nodes
- Boundary conditions for first derivative





Advantages:

- More accurate than linear interpolation
- Derivatives are continuous
- Avoids issues with polynomials of high degree

Disadvantages:

- Implementation not so simple
- Artefacts like large oscillations between nodes are possible

From the course by Volodymyr Vovchenko, https://github.com/vlvovch/PHYS6350-ComputationalPhysics

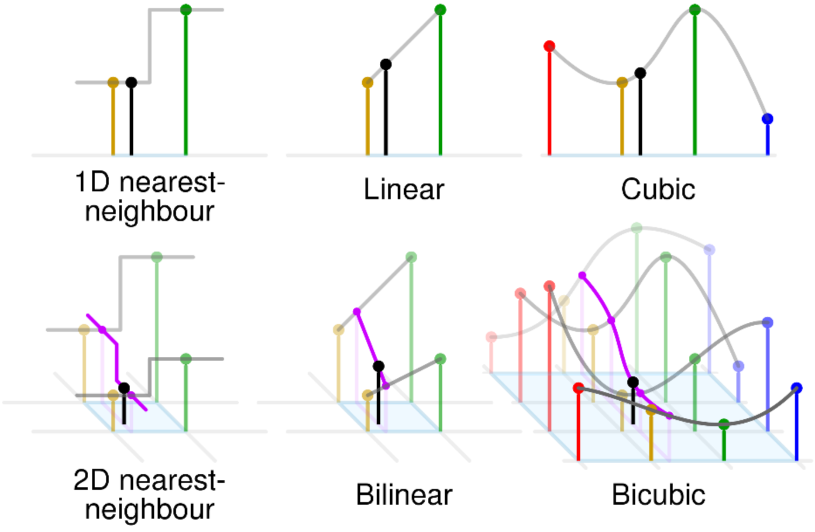
Multiple dimensions

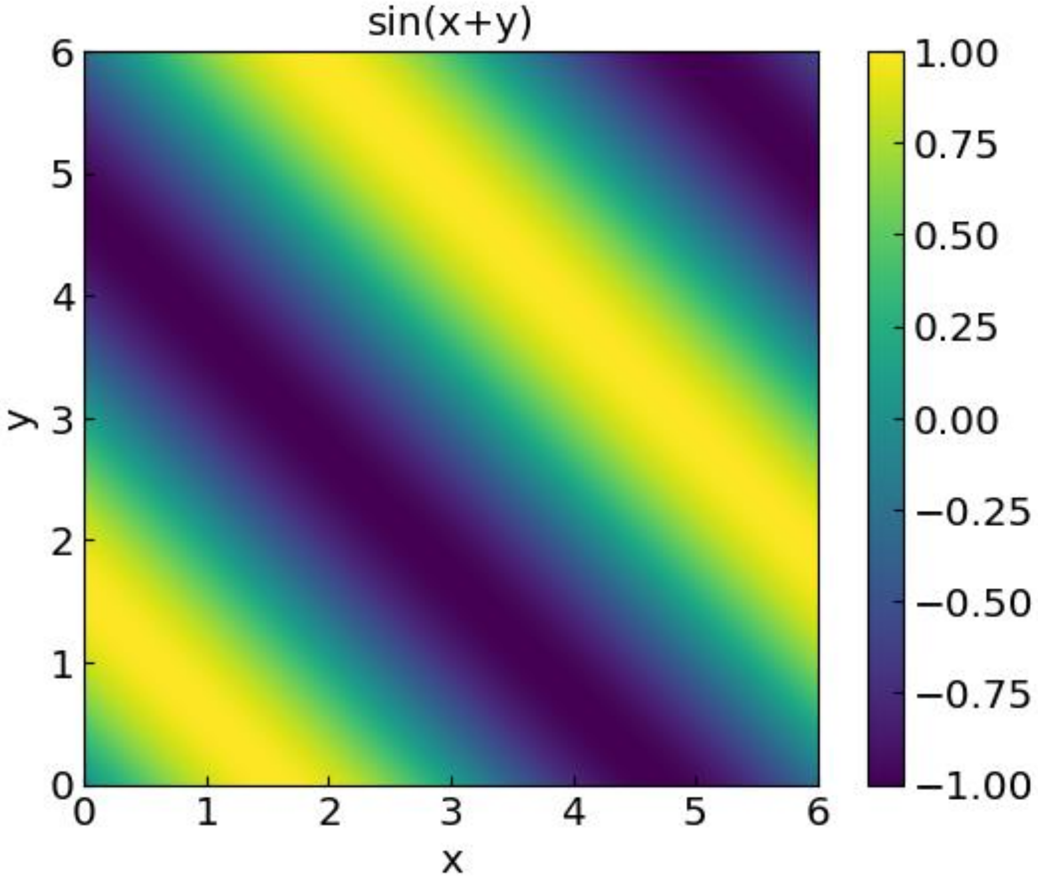
Functions of more than one variable, e.g. f(x,y) = sin(x+y)

Data points: (x_i, y_i, f_i)

Main methods:

- Nearest-neighbor
- Successive 1D interpolations





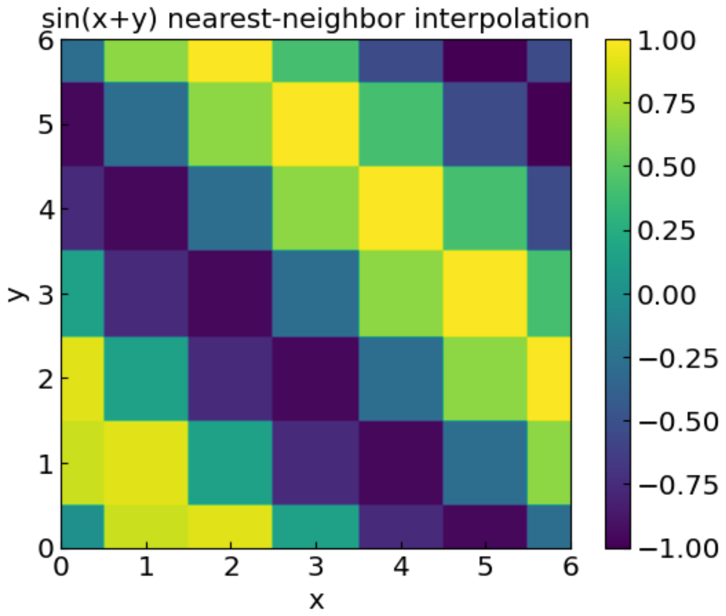
From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

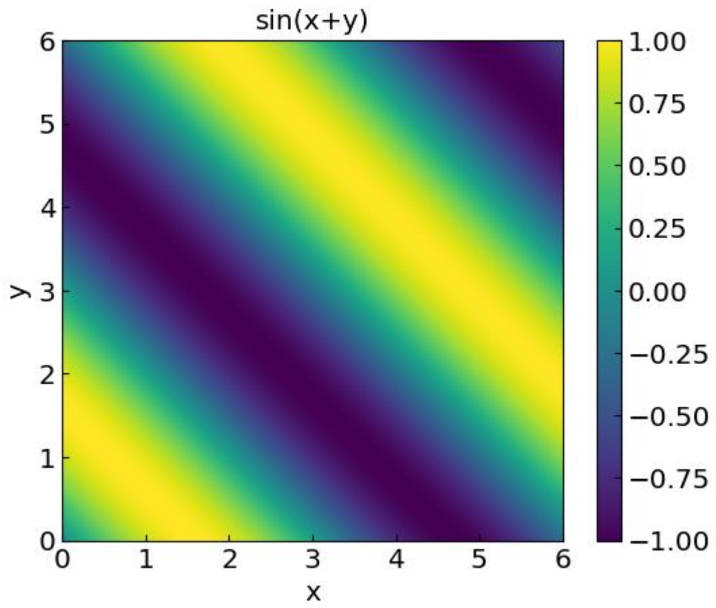
2D nearest-neighbor

2D nearest-neighbor:
Simply assign the value of the closest data point to *(x,y)* in the plane

Consider f(x,y) = sin(x+y)

Data points at integer values x,y=0,1,…6 (*regular grid*)



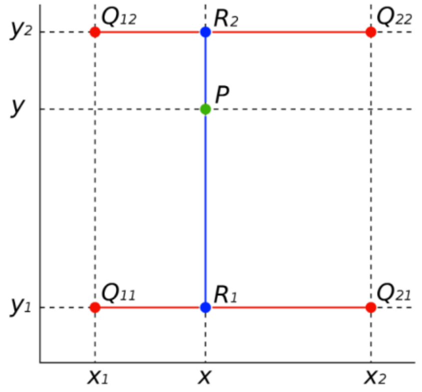


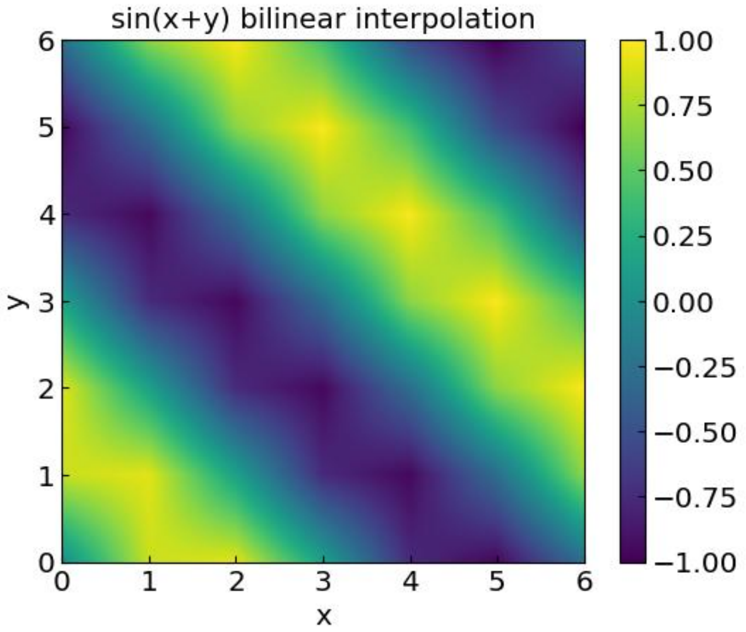
From the course by Volodymyr Vovchenko,
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

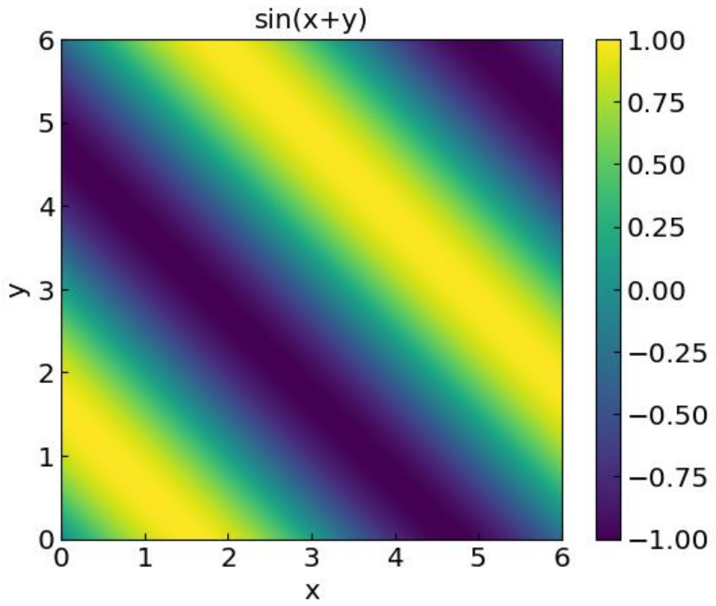
Bilinear interpolation

Bilinear interpolation: apply linear interpolation twice

1. Find (x_1, x_2) and (y_1, y_2) such that $x \in (x_1, x_2)$ and $y \in (y_1, y_2)$
2. Calculate R_1 and R_2 for $y = y_1$ and $y = y_2$, respectively, by applying linear interpolation in x
3. Calculate the interpolated function value at (x, y) by performing linear interpolation in y using the computed values of R_1 and R_2







From the course by Volodymyr Vovchenko, https://github.com/vlvovch/PHYS6350-ComputationalPhysics