PHOT 411: Numerical Methods for Photonics

LECTURE 01

Michaël Barbier, Fall semester (2024-2025)

ERRORS

WHERE DO ERRORS ORIGINATE FROM?

1. **Significant figures**: Measurement errors, Imperfect input data

2. Machine errors or **round-off Errors**: floating point representation

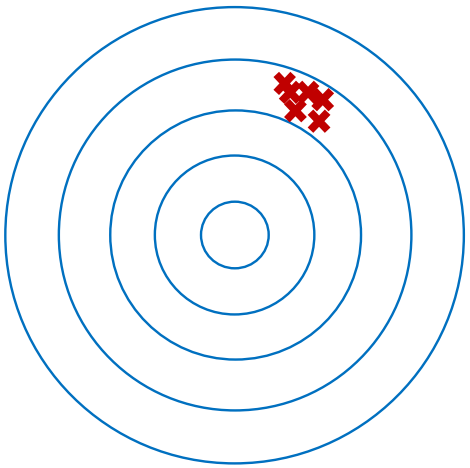
3. Truncation errors from **Numerical approximations**

Difference between type of errors:

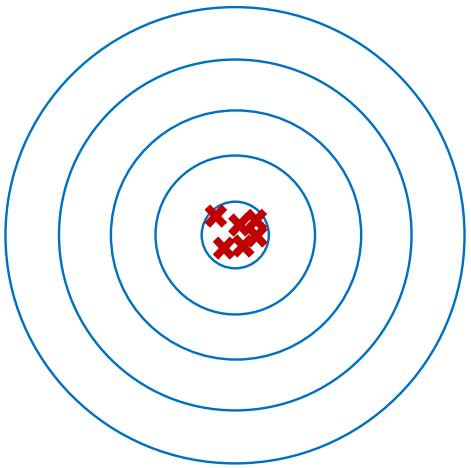
- Precision $\longleftrightarrow$ uncertainty
- accuracy $\longleftrightarrow$ bias, systematic error

WHERE DO ERRORS ORIGINATE FROM?

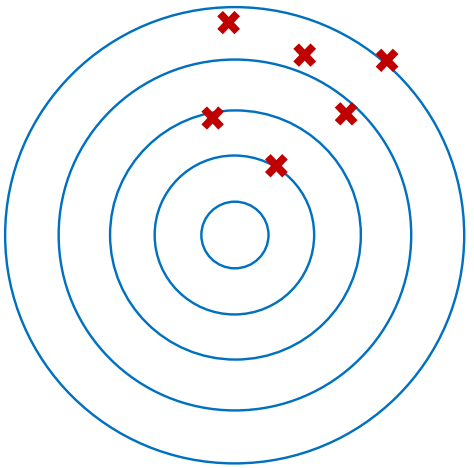
1. **Significant figures**: Measurement errors, Imperfect input data
2. Machine errors or **round-off Errors**: floating point representation
3. Truncation errors from **Numerical approximations**



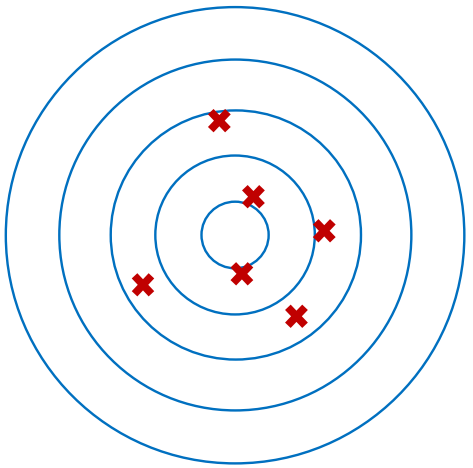
inaccurate
precise



accurate
precise



inaccurate
imprecise



accurate
imprecise

SIGNIFICANT FIGURES

- We only know some number up to certain accuracy
- Scientific notation indicates accuracy

Scientific notation:

$$1.50 \times 10^2 \ \mathrm{km/h} \qquad (3 \text{ significant digits})$$
$$7.85356 \times 10^8 \ \mathrm{km/h} \qquad (6 \text{ significant digits})$$

- More significant numbers is more accurate

RELATIVE ERROR

- We only know some number up to certain accuracy
- Compare the error to the actual value
- Relative errors are important!
- How much accuracy do you need?

$$\text{true error } E_t = \text{true value} - \text{approximated value}$$

$$\text{relative error } \varepsilon_t = \frac{\text{true error}}{\text{true value}}$$

Do you have a good estimate for ε_a?

APPROXIMATING ERRORS

$$\text{relative error } \varepsilon_t = \frac{\text{true error}}{\text{true value}}$$

- Often we don't know the true value
- So … estimate the true value: **approximate value** $\longrightarrow$ estimation of the error

$$\text{Approximate relative error } \varepsilon_a = \frac{\text{approximate error}}{\text{approximate value}}$$

How to obtain a good estimate for ε_a?

FLOATING POINT NUMBERS (SINGLE & DOUBLE)

S = sign-bit (1 bit), E = exponent (8 bits), M = mantissa (23 bits)

To calculate real number x:

$$x = (-1)^S \times M \times 2^{(E-127)}$$

FLOATING POINT NUMBERS (SINGLE & DOUBLE)

To calculate real number x (single precision):

$$x = (-1)^S \times M \times 2^{(E-127)}$$

- Real numbers are finite in computers
- Finite number of significant figures (binary)
- Maximum number?
- Minimum nonzero number?

```
1 0.3/0.1 - 3
```

```
ans =

4.4409e-16
```

FLOATING POINT NUMBERS (SINGLE & DOUBLE)

To calculate real number x (single precision):

$$x = (-1)^S \times M \times 2^{(E-127)}$$

type	minimum	maximum
single	$2^{-126} \approx 1.18 \times 10^{-38}$	2.85×10^{45}
double	$2^{-1022} \approx 2.23 \times 10^{-308}$	1.80×10^{308}

```
1  realmin("double")
```

```
ans =

2.2251e-308
```

CUMULATION OF ROUND-OFF ERRORS

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} 1/10 & -1/5 & 3/10 \\ 0 & 2 & 4 \\ -1 & 1 & 0 \end{pmatrix}$$

$$\begin{aligned} \det(A) &= a_{11} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12} \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13} \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} \\ &= 1 \end{aligned}$$

$$\Rightarrow \det(A^n) = \det(A \cdot A \cdot A \cdot A \ldots) = 1$$

CUMULATION OF MACHINE ERRORS

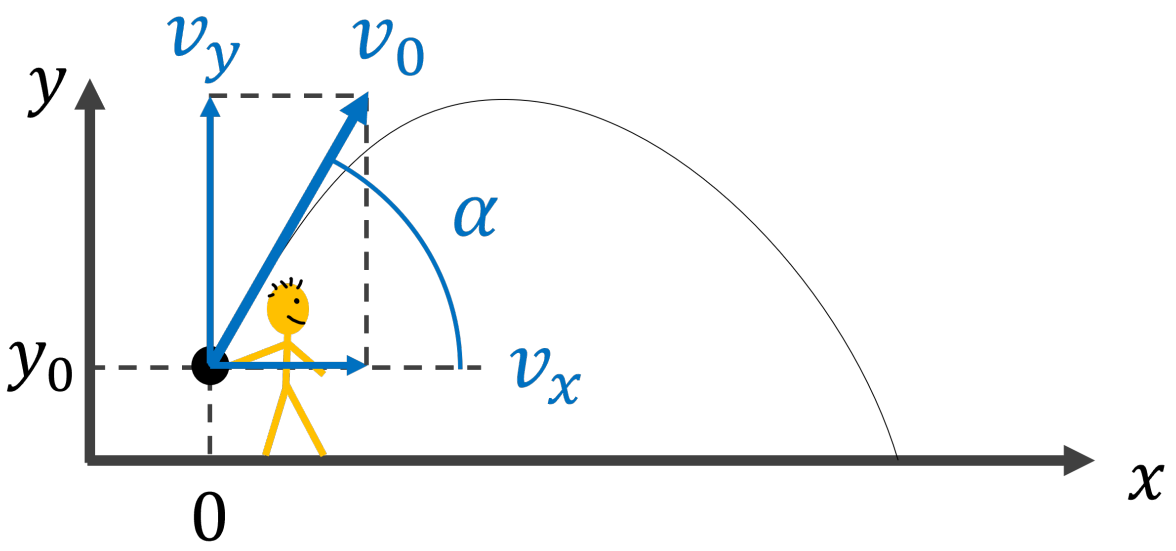
Numerical result for increasing power n:

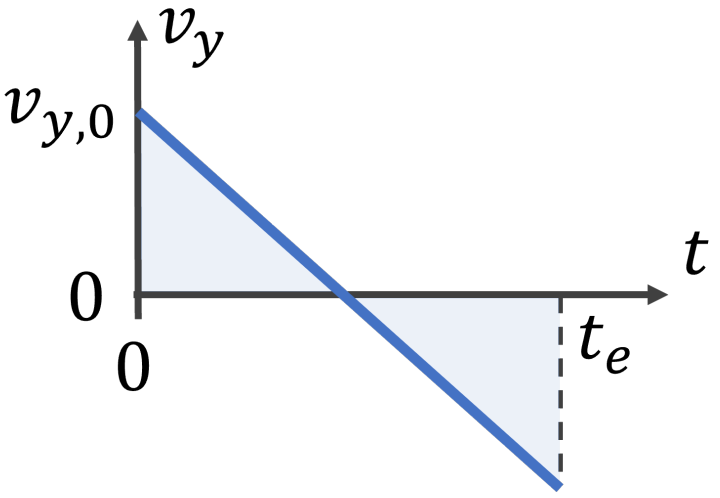
```
det(A^1) = 1.0
det(A^2) = 1.0000000000000009
det(A^3) = 0.9999999999999853
det(A^4) = 0.9999999999999742
det(A^5) = 0.9999999999996483
det(A^6) = 0.9999999999888063
det(A^7) = 0.9999999997708064
det(A^8) = 1.0000000024722748
det(A^9) = 1.0000000062737353
det(A^10) = 0.9999999184512143
det(A^11) = 0.9999991943727314
```

ERRORS DUE TO NUMERICAL APPROXIMATIONS

- Depend on numerical method and its implementation
- These errors are independent of round-off errors
- There are also physical model approximations: these are sometimes intertwined with the numerical method. Some Examples:
  - “Raycasting” (e.g. Zemax) uses geometric optics, very small optical systems cannot be modeled.
  - Finite difference methods can work in the time domain (FDTD) or frequency domain. Their errors depend on the problem AND the numerical approximations within the method.

THE TRAJECTORY OF A BALL





- Gravitation: $g = 9.81\ \mathrm{m/s^2}$
- Air resistance: ignore for a slow heavy ball
- Horizontal velocity is constant

TRAJECTORY OF A BALL: NUMERICALLY

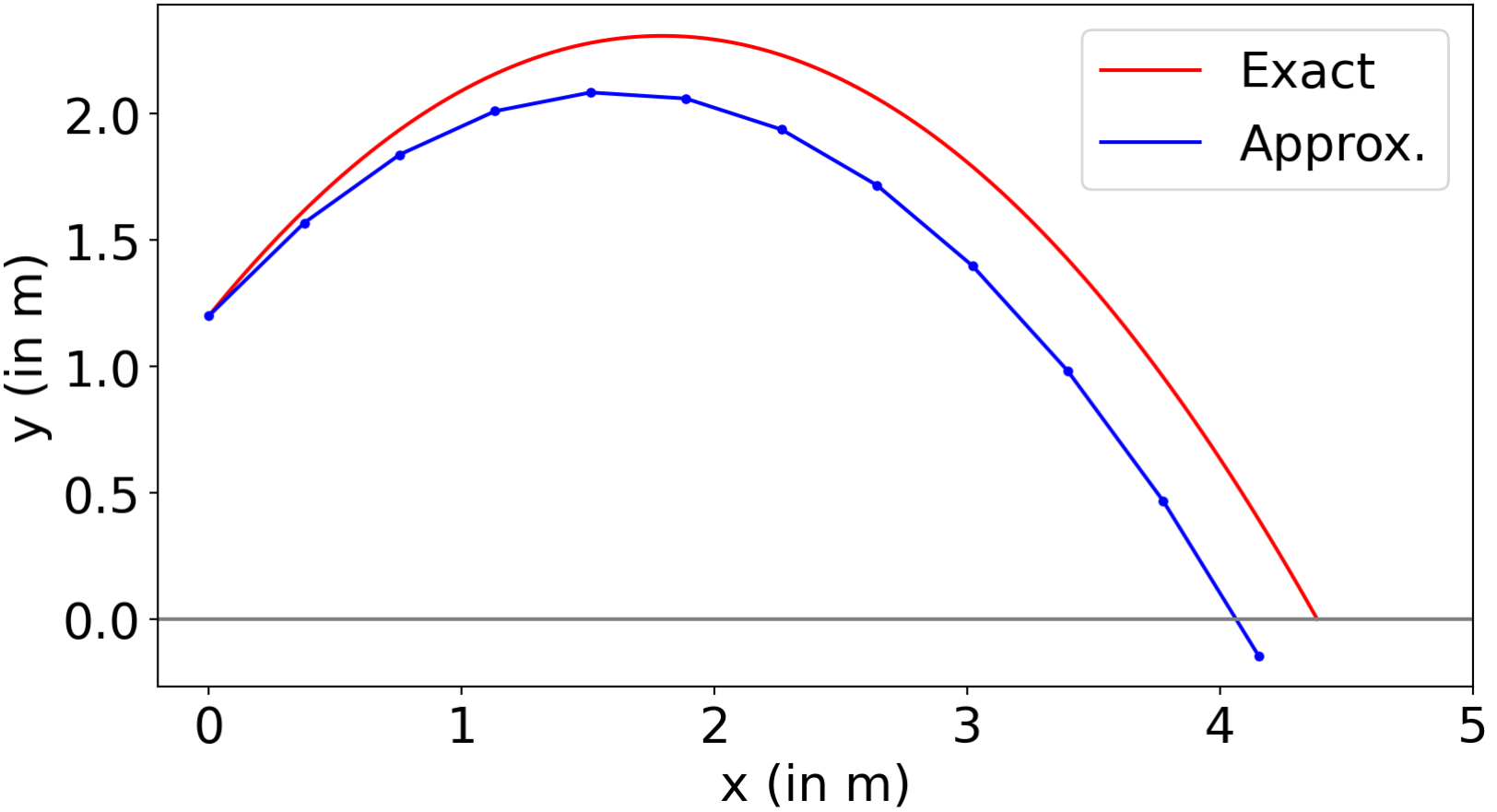
- Small time steps: $\delta t = 0.1$ s
- Constant velocity over δt
- Approximation true trajectory

Coordinates:

```
t = 0.10: (0.38,1.57)
t = 0.20: (0.76,1.84)
t = 0.30: (1.13,2.01)
t = 0.40: (1.51,2.08)
t = 0.50: (1.89,2.06)
t = 0.60: (2.27,1.94)
t = 0.70: (2.64,1.72)
t = 0.80: (3.02,1.40)
t = 0.90: (3.40,0.98)
t = 1.00: (3.78,0.47)
t = 1.10: (4.15,-0.15)
```

TRAJECTORY OF A BALL: ERRORS

Mismatch with exact curve due to approximation



PROPAGATING ERRORS

Performing mathematical operations changes the error

$$(1.31 \pm 0.04) + (2.5 \pm 0.1) = 3.81 \pm ?$$

Applying functions changes the error:

$$\ln(23 \pm 2) = \ln(23) \pm ?$$

How to propagate the errors of variables in any formula ?

PROPAGATING ERRORS: TAYLOR EXPANSIONS

Suppose a function $f(x)$:

- true value is x
- true error is Δx

Taylor expansion: function of a single variable

$$f(x + \Delta x) = f(x) + f'(x)\,\Delta x + \frac{f''(x)}{2!}\,\Delta x^2 + \frac{f^{(3)}(x)}{3!}\,\Delta x^3 + \ldots$$

Bundle higher orders together in a "rest" term R_n:

$$f(x + \Delta x) = f(x) + f'(x)\,\Delta x + R_3$$

PROPAGATING ERRORS: TAYLOR EXPANSIONS

$$f(x + \Delta x) = f(x) + f'(x)\,\Delta x + R_3$$

$$\Rightarrow f(x + \Delta x) - f(x) = f'(x)\,\Delta x + R_3$$

Error on function values:

$$\Rightarrow \Delta f = \frac{\partial f(x)}{\partial x}\,\Delta x$$

Extend this to $f(x, y, z, \ldots)$ with multiple variables:

$$\Delta f = \frac{\partial f}{\partial x}\,\Delta x + \frac{\partial f}{\partial y}\,\Delta y + \frac{\partial f}{\partial z}\,\Delta z + \ldots$$

PROPAGATING ERRORS: TAYLOR EXPANSIONS

Error propagation formula:

$$\Delta f = \frac{\partial f}{\partial x}\,\Delta x + \frac{\partial f}{\partial y}\,\Delta y + \frac{\partial f}{\partial z}\,\Delta z + \ldots$$

Apply this to the sum of two numbers:

$$f(x, y) = x + y$$

$$\Delta f = \frac{\partial (x+y)}{\partial x}\Delta x + \frac{\partial (x+y)}{\partial y}\Delta y = \Delta x + \Delta y$$

PROPAGATING UNCERTAINTY VS. TRUE ERRORS

- Often we don't know the true error
- We can use uncertainty of our value: estimated error σ

How do we propagate the uncertainty σ ?

- Uncertainty propagation for $f(x, y, z, \ldots)$
- Suppose $x \pm \sigma_x$, $y \pm \sigma_y$, $z \pm \sigma_z$, are known

[Math Processing Error]

PROPAGATING UNCERTAINTY

Propagation formula:

[Math Processing Error]

Example problem:

$$f(x, y) = 3\,x^2 + y \quad \text{with} \quad \sigma_x = 0.1 \text{ and } \sigma_y = 0.4$$

[Math Processing Error]

SOLVING SYSTEMS OF LINEAR EQUATIONS

LINEAR EQUATIONS

A system of linear equations with

- Unknown variables x_i
- Constant coefficients a_{ij}
- Constants b_i

[Math Processing Error]

SOLVING SMALL SYSTEMS

A system of linear equations with

- Unknown variables x_i
- Constant coefficients a_{ij}
- Constants b_i

[Math Processing Error]

MATRIX EQUATIONS

- Systems of linear equations in matrix-form

[Math Processing Error]

Which can be written more compact:

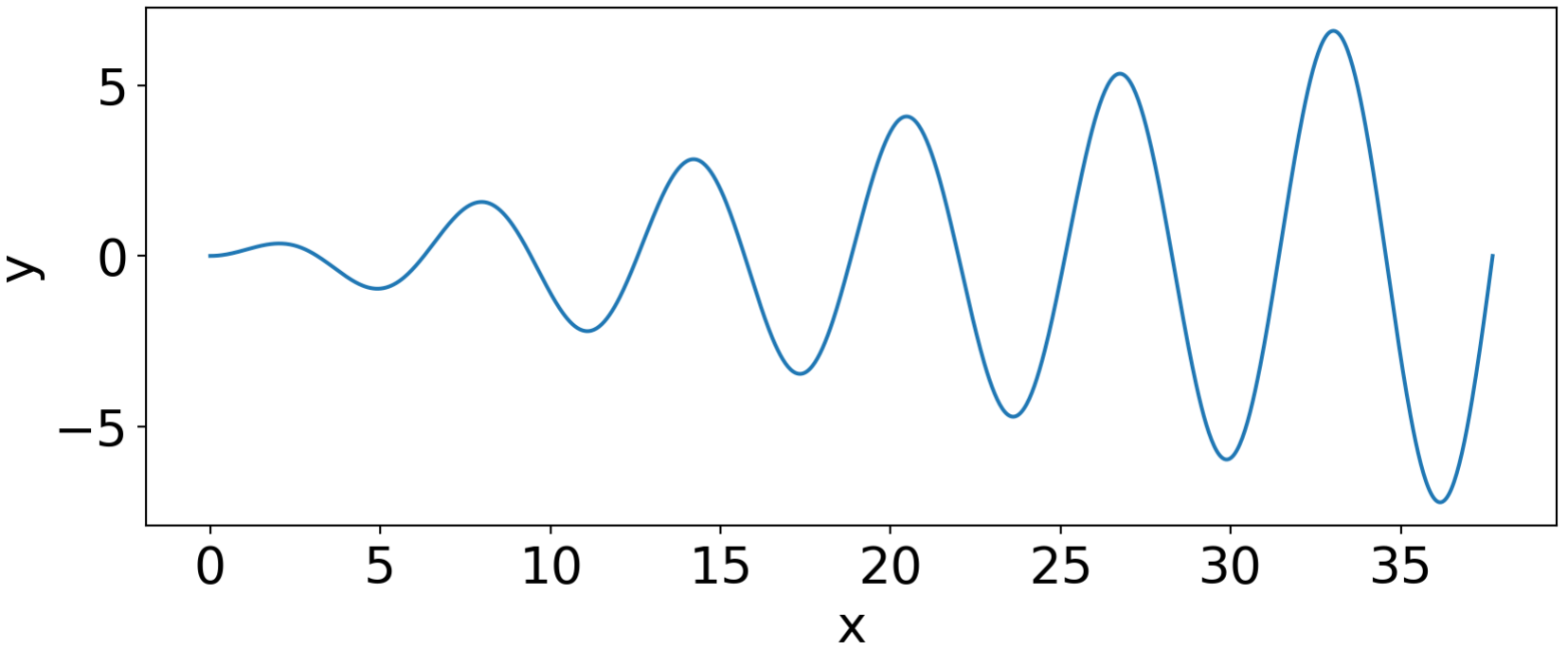
$$[A]\,[x] = [b] \qquad \text{or} \qquad A\,x = b \qquad \text{or} \qquad A\,\vec{x} = \vec{b}$$

APPROXIMATING FUNCTIONS

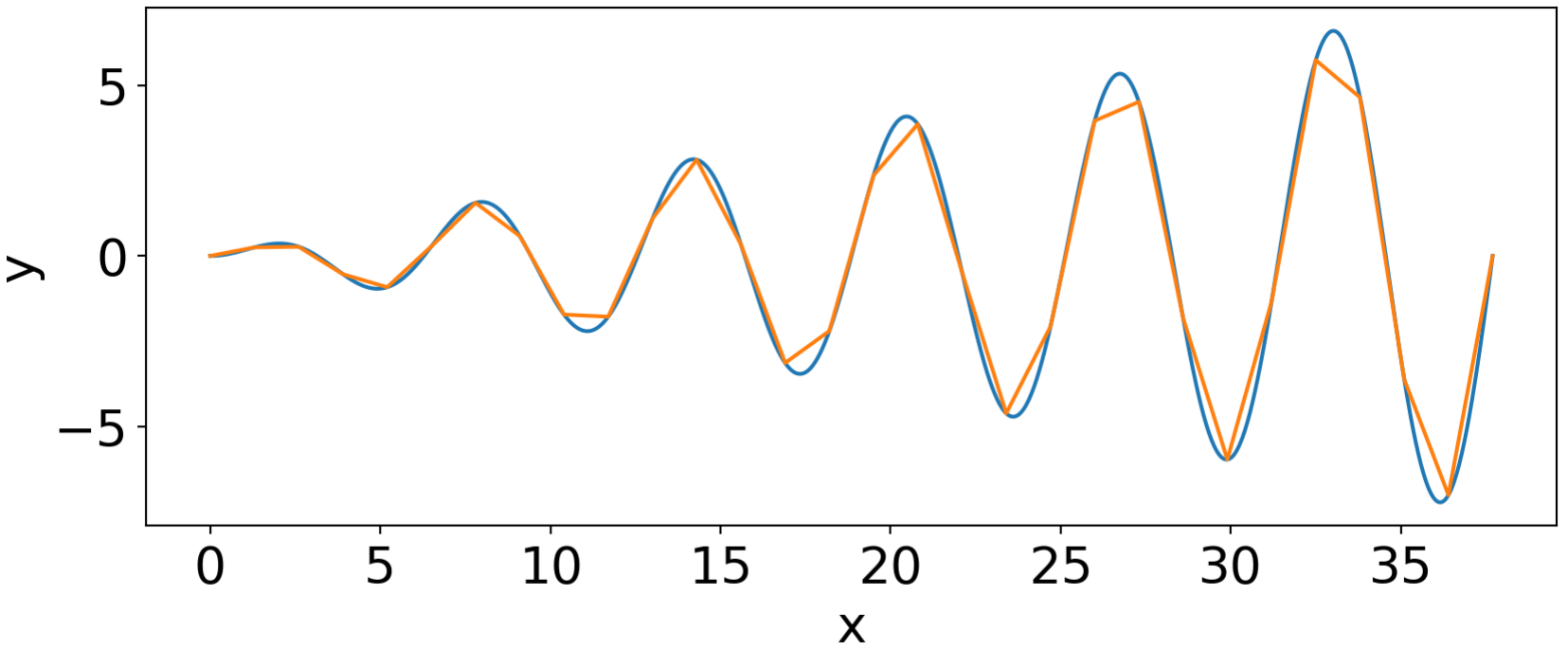
DISCRETIZING FUNCTIONS

- simplest: finite number of x_i and $y_i = f(x_i)$ (with $i = 1, \ldots, N$)
- Points x_i can be regularly spaced
- Accuracy depends on:
  - how smooth the real function is
  - how close the points are taken But more points $\longrightarrow$ **more data, slower**

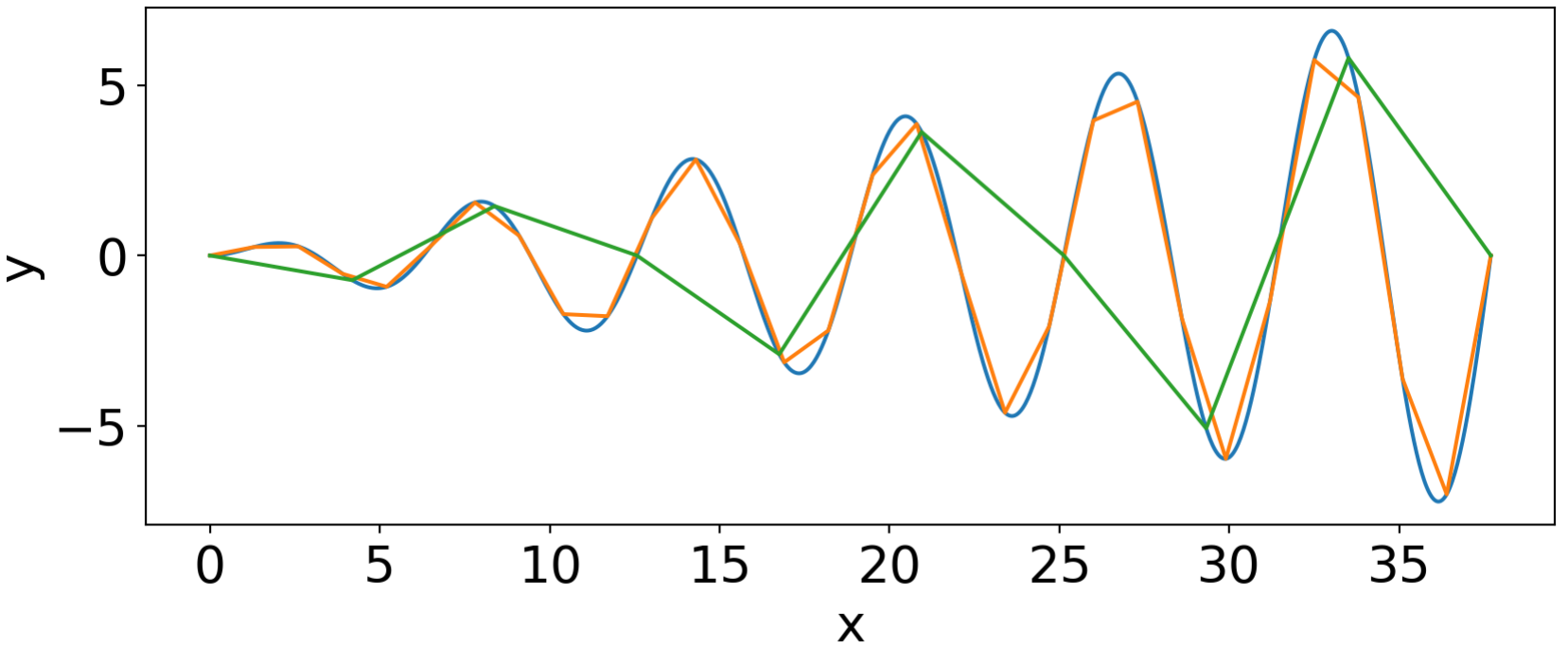
DISCRETIZING FUNCTIONS



DISCRETIZING FUNCTIONS



DISCRETIZING FUNCTIONS



How many points should be taken ?