# Week 1 — Introduction to Machine Learning

## From Artificial Intelligence to Function Approximation

UdeM AI — Université de Montréal

# What is Artificial Intelligence?

**Artificial Intelligence (AI)** studies systems capable of:

- perceiving their environment,
- processing information,
- making decisions toward a goal.

A compact view:

$$\text{AI} = \text{Perception} + \text{Decision} + \text{Reasoning}(+\ \text{Learning})$$

Learning is not always required, but it is now central to most modern systems.

# Where does Machine Learning fit?

**Machine Learning (ML)** is a subfield of AI.

Key idea: instead of explicitly programming a solution, we design a system that learns from data.

A useful hierarchy:

$$\text{AI} \supset \text{Machine Learning} \supset \text{Neural Networks}$$

The core of ML is not the algorithm itself, but representation and learning from examples.

# Machine Learning — a minimal framework

We aim to learn an unknown relationship:

$$x \mapsto y$$

We introduce a parameterized model:

$$\hat{y} = f_\theta(x)$$

The parameters $\theta$ are learned from data:

$$\theta^* = \arg\min_\theta \frac{1}{N} \sum_{i=1}^{N} \ell\big(f_\theta(x_i), y_i\big)$$

Goal: perform well on new data, not only on observed samples.

# Learning is function approximation

We assume the existence of an unknown target function:

$$f : \mathbb{R}^d \to \mathbb{R}$$

Nature provides observations:

$$(x_i, y_i), \qquad y_i \approx f(x_i)$$

**Central problem:**

$$\text{find } f_\theta \text{ such that } f_\theta(x) \approx f(x)$$

Machine learning is fundamentally a problem of **function approximation from data**.

# Concretely: “approximation from data”

Saying *“approximate a function from data”* means:

- the true rule $f$ is unknown,
- we only observe samples $(x_i, y_i)$,
- we construct a rule $f_\theta$ that best reproduces these examples.

But the goal is not memorization:

we want a rule that works on new $x$.

So the problem is both:

- **mathematical** (approximation),
- **statistical** (generalization).

# Choosing a representation for $f_\theta$

The key point is not only optimization, but the **choice of the form of $f_\theta$**.

Possible representations include:

- polynomials,
- Fourier series,
- superpositions of nonlinear functions,
- grids (fixed or adaptive),
- ansatz guided by problem structure.

This choice encodes assumptions about the shape of the function to be learned.

# What does “choosing the form of $f_\theta$” mean?

Choosing the form of $f_\theta$ means deciding **how** the model can represent a function.

Examples (intuition):

- **Polynomials**: assume a globally smooth function.
- **Fourier**: assume the function is well described by oscillations.
- **Nonlinear superpositions**: combine flexible nonlinear building blocks.
- **Adaptive grids**: allocate more resolution where variation is high.
- **Ansatz**: impose structure inspired by the domain (physics, geometry, etc.).

Two models may be optimized identically, but **only one** may have the right structure.

# Classical bases: polynomials and Fourier

**Polynomials:**

$$f_\theta(x) = \sum_{n=0}^{N} \theta_n x^n$$

**Fourier series (finite interval):**

$$f_\theta(x) = \sum_{|k|\leq K} \theta_k e^{ikx}$$

These bases are:

- universal under certain assumptions,
- analytically simple,
- sometimes inefficient for local structures.

# Expressivity vs efficiency (parameters & data)

Two fundamental criteria:

**Expressivity** Can the model approximate a wide class of functions?

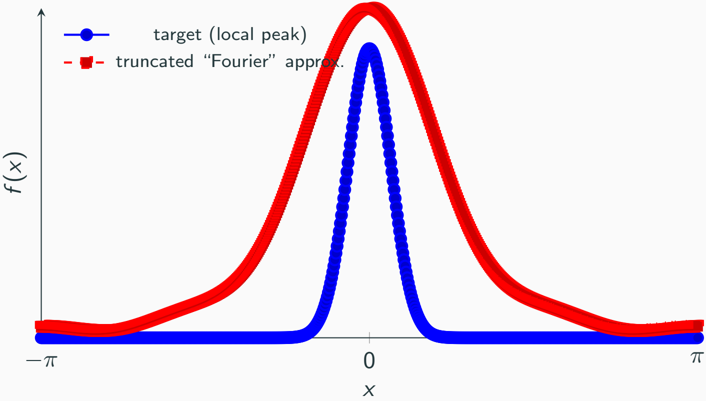
**Efficiency** How many **parameters** and **data points** are needed?

Important point:

High expressivity does not guarantee good generalization.

(Being able to represent “everything” can also make learning unstable or noise-sensitive.)

# Visual example: a local peak and Fourier approximation



Idea: a global basis (like Fourier) may require many terms to capture highly local structure.

# What does “local structure” mean?

A function has **local structure** when:

- it varies sharply in some regions,
- and very little elsewhere.

Intuitive examples:

- a narrow peak,
- a sharp transition between regimes,
- a small “interesting” region in an otherwise simple signal.

In such cases, using the same resolution everywhere is inefficient.

# Why does this matter in practice?

In practice, real-world data are often:

- heterogeneous,
- noisy,
- concentrated in specific regions of the input space.

We therefore want to:

- allocate capacity where it matters,
- avoid wasting it where the function is simple.

This is exactly what good representations aim to do: **adapt to the structure of the data** (and thus generalize better).

# Key message

Machine learning is not magic.

It is a framework to:

- approximate unknown functions,
- from data,
- using effective representations.

In practice, performance often depends more on the model's implicit assumptions than on the optimization algorithm itself.

# References

- F. Marquardt, *Advanced Machine Learning for Physics, Science, and Artificial Scientific Discovery*, YouTube lecture series.
- A. Ng, *Machine Learning*, Coursera.