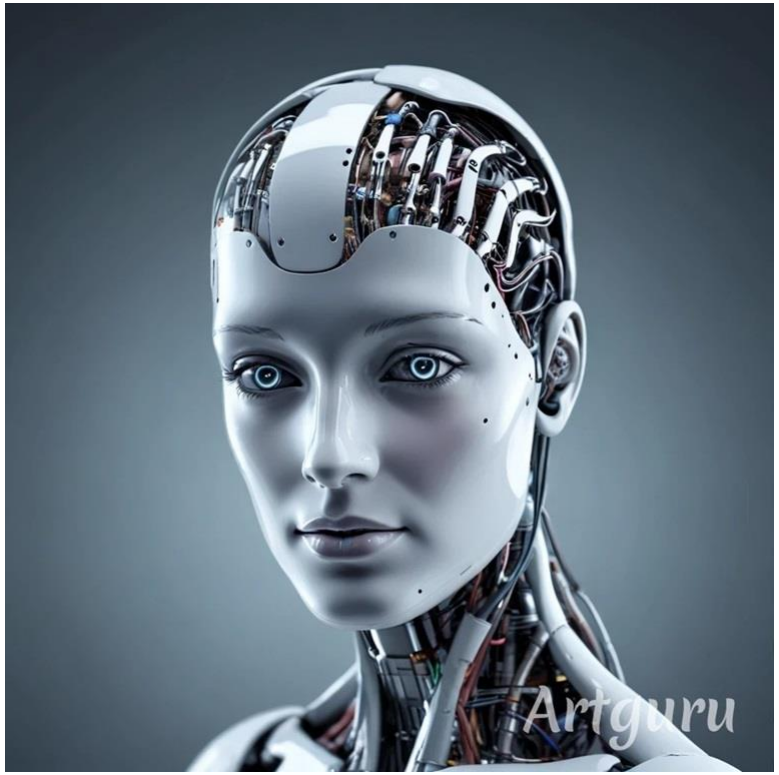
Intelligence Artificielle
et
Analyse de données

Johan Peralez

1. Définir l’Intelligence Artificielle (IA)



- L’IA est une notion floue et qui évolue rapidement => sa définition dépend:
  - du domaine (traitement d’image, contrôle, etc.)

    e.g. distinguer des visages vs faire marcher un robot
  - de l’époque (depuis le milieu du XXe siècle)

    e.g. Deep Blue vs Alpha Go

- Exemples:
  - « L'**automatisation** d'activités que nous **associons à la pensée humaine**, comme la prise de décision, la résolution de problème ou l'apprentissage. » (BELLMAN 1978)

    subjectif (nous ?)
  - « L'étude de comment **programmer les ordinateurs** pour qu'ils réalisent des tâches pour lesquelles les êtres humains sont **actuellement meilleurs**. » (RICH & KNIGHT 1991)

    paradoxal (jeu d’échec vs football ?)
  - « Ensemble de **théories et de techniques** mises en œuvre en vue de réaliser des machines capables de **simuler l'intelligence humaine**. » (Larousse 2024)

    vague (définir intelligence ?)

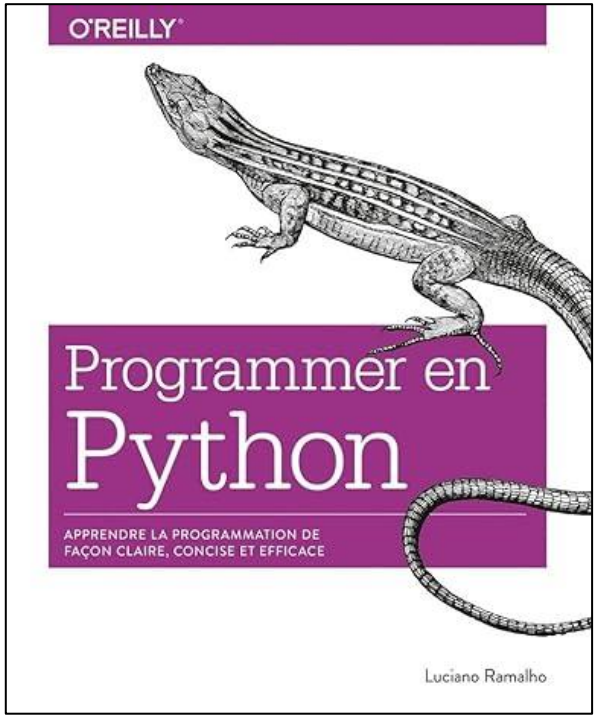
- Difficulté à définir l’IA = difficulté à définir l’Intelligence.

  Selon [Larousse 2024], l’intelligence =

  1. « Ensemble des fonctions mentales ayant pour objet la connaissance conceptuelle et rationnelle »
  2. « Aptitude d'un être humain à s'adapter à une situation, à choisir des moyens d'action en fonction des circonstances »

- Approche ‘’pragmatique’’:
  - IA = ensemble des méthodes qui sont habituellement classées dans l’IA par les gens de son domaine.
  - Traitement de l’image : réseaux de neurones convolutifs (CNN), algorithmes de segmentation, etc.
  - En contrôle : réseaux de neurones récurrents (RNN), algorithmes d’apprentissage par renforcement (RL), etc.

2. Prérequis (programmation-mathématiques)



- L'IA et l'analyse de données nécessitent des compétences :
  - en **programmation**

    Python (langage le + utilisé en IA)
  - en **mathématiques**

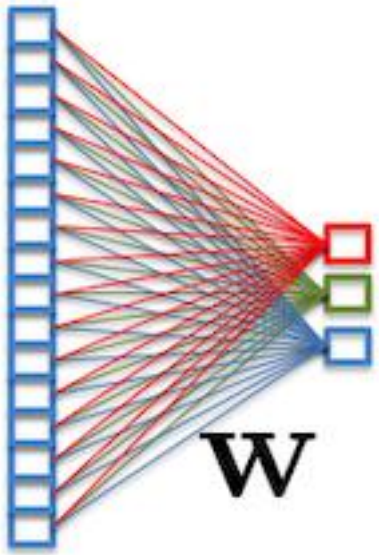
    Algèbre linéaire, calcul différentiel, probabilités, statistiques

- Ressources Python en ligne:
  1. https://www.learnpython.org/ (exos interactifs)

     i) Learn the Basics, ii) Advanced Tutorials
  2. *https://www.france-ioi.org/algo/chapters.php* (exos interactifs)

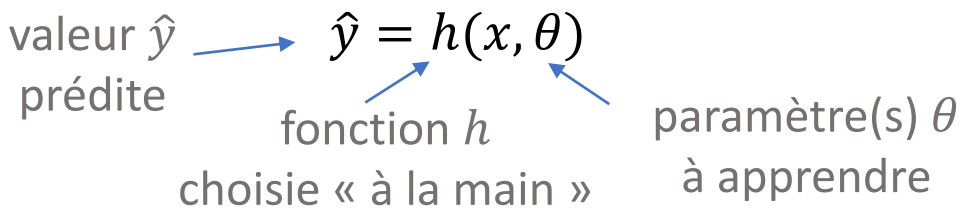
     Niveau 1 à 6
  3. Librairie NumPy : https://courspython.com/apprendre-numpy.html (cours)

3. Introduction aux réseaux de neurones

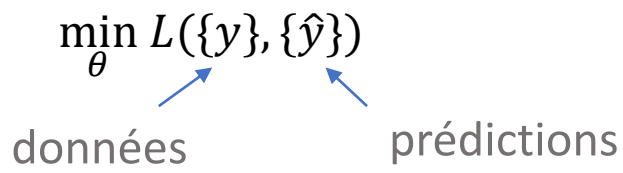


3.1 Problème de régression

Formulation : **apprendre à prédire** une valeur de sortie y à partir d'une donnée d'entrée x



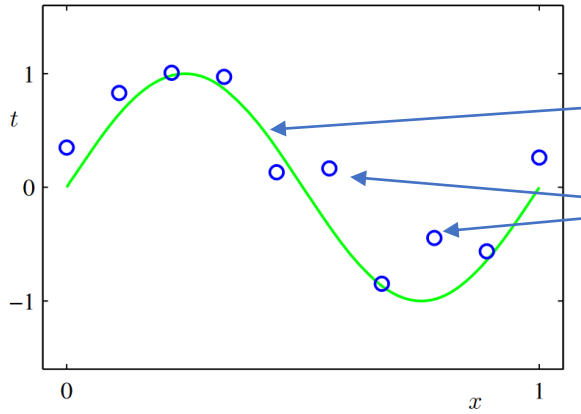
afin de minimiser une fonction **coût** L :



=> **problème d'optimisation**

Exemple :

Données :



$\sin(2\pi x)$

Sorties t générées avec cette fonction + du « bruit »

[C. Bishop, Pattern recognition and Machine learning, 2006]

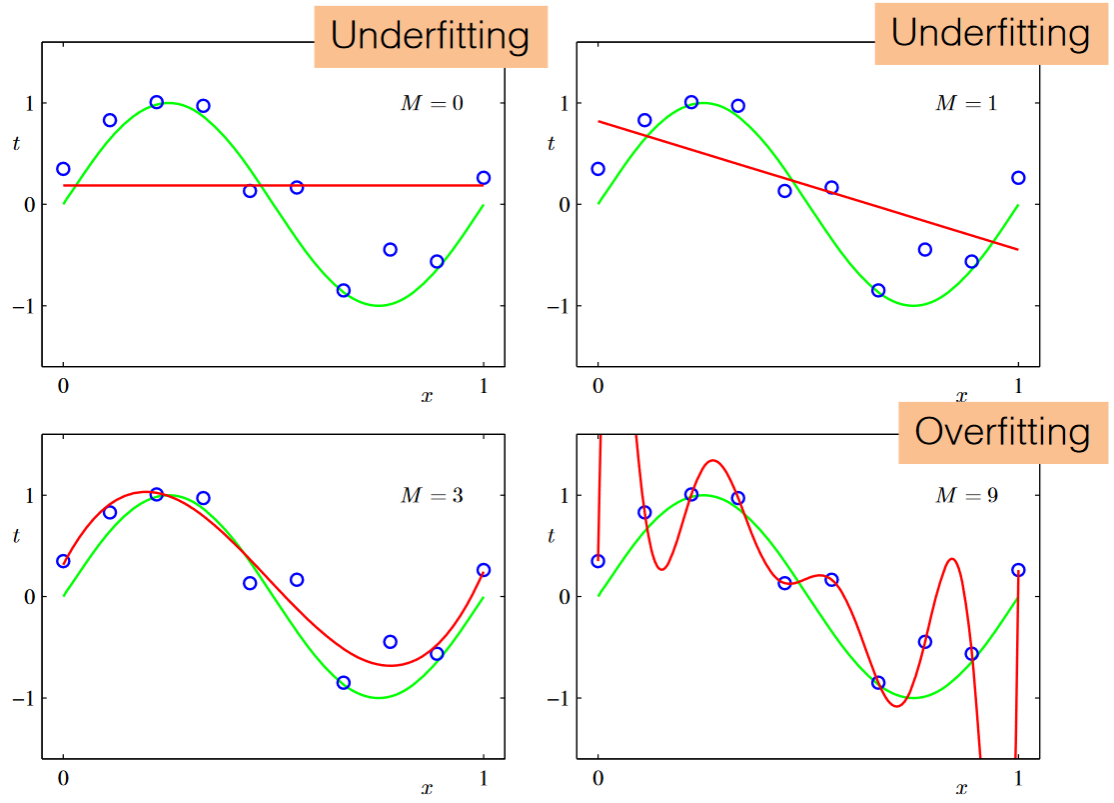
On choisit de prédire avec un polynôme d'ordre M :

$$\hat{t} = h(x, \theta) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_M x^M$$

Et une fonction de coût « moindre carrés »:

$$L = \frac{1}{2}\sum(t - \hat{t})^2$$

Quel ordre M choisir pour le polynôme ?



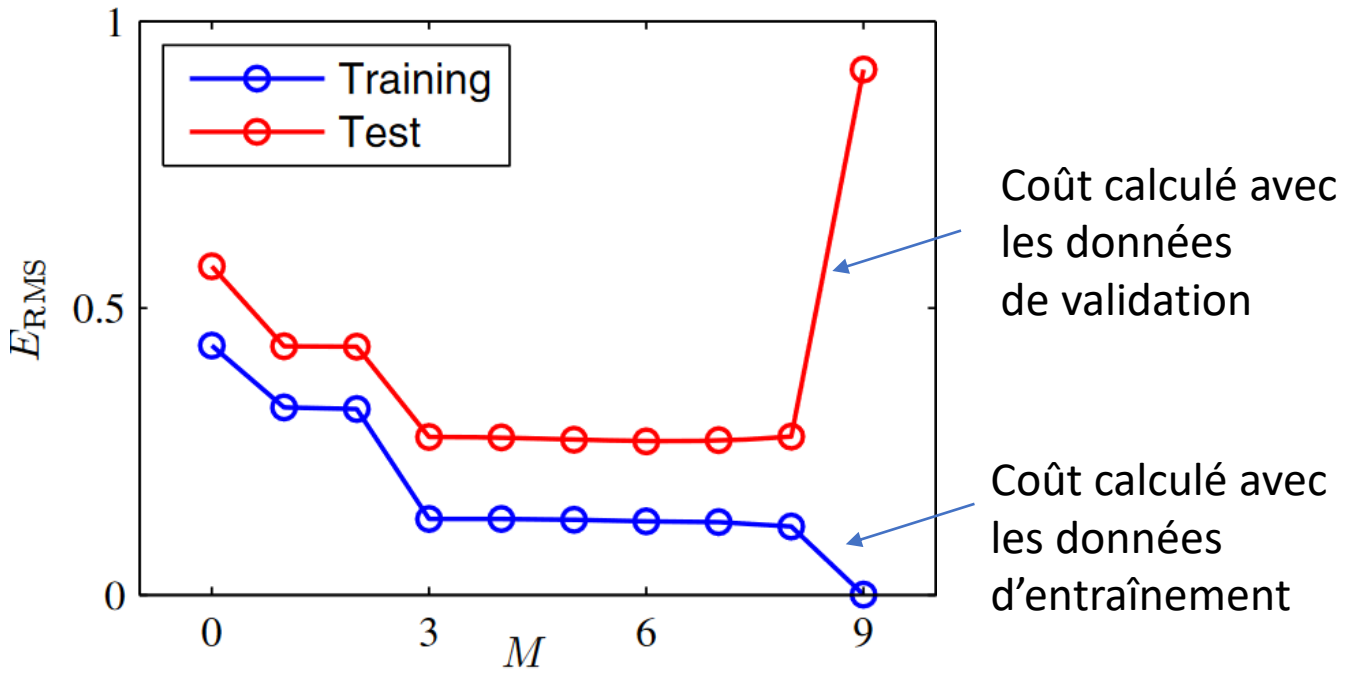
[C. Bishop, Pattern recognition and Machine learning, 2006]

- Pour M trop petit : problème de sous-apprentissage (***underfitting***).
- Pour M trop grand : problème de sur-apprentissage (***overfitting***).

Remarque (**polynômes de Lagrange**) : pour n données distinctes il existe un (unique) polynôme d'ordre n-1 qui passe exactement par chaque donnée.

Séparation des données en, au moins, 2 ensembles :

- Données d'entraînement (*training set*).
  Pour l'apprentissage de θ, i.e. la résolution du problème d'optimisation.
- Données de validation (*validation set*).
  Permet de détecter le surapprentissage :



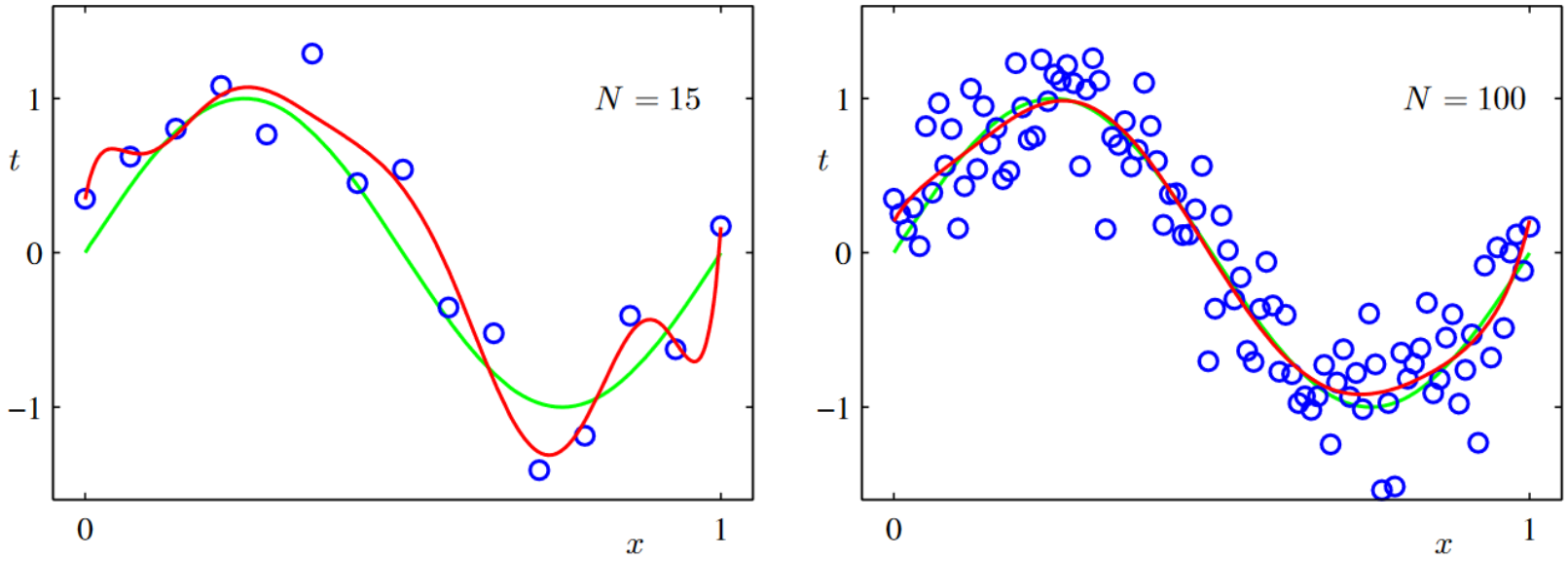
Coût calculé avec les données de validation

Coût calculé avec les données d'entraînement

[C. Bishop, Pattern recognition and Machine learning, 2006]

Big Data ?

Augmenter (fortement) le nombre de données d'apprentissage permet de réduire le surapprentissage :



M=9

[C. Bishop, Pattern recognition and Machine learning, 2006]

3 grandes difficultés des problèmes d'apprentissage :

1. Expressivité

   Mon modèle, i.e. ma fonction h, peut-elle apprendre des phénomènes complexes ?

2. Difficulté à entraîner

   Le problème d'optimisation, i.e. minimiser la différence entre prédictions et données, est-il difficile ?

   e.g. si on résout par descente de gradient, la fonction coût est-elle dérivable ? lisse ?

3. Généralisation

   Comment mon modèle se comporte sur des données qui ne sont pas dans le *training set* ?

   Capacité à interpoler / extrapoler ?

   e.g. le surapprentissage implique une mauvaise généralisation.

Exercice (Python, NumPy):

Générer et visualiser un jeu de données similaire à l'exemple ci-dessus.

1. Importer les librairies *numpy* et *matplotlib*

```
import numpy as np
from matplotlib import pyplot as plt
```

2. *Paramètres*

```
N = 30 # nombre de points
```

3. *Générer (aléatoirement entre 0 et 1) les données d'entrées $\{x\}$, de taille N.*

   Utiliser la fonction *np.random.uniform(…)* pour utiliser une distribution uniforme.

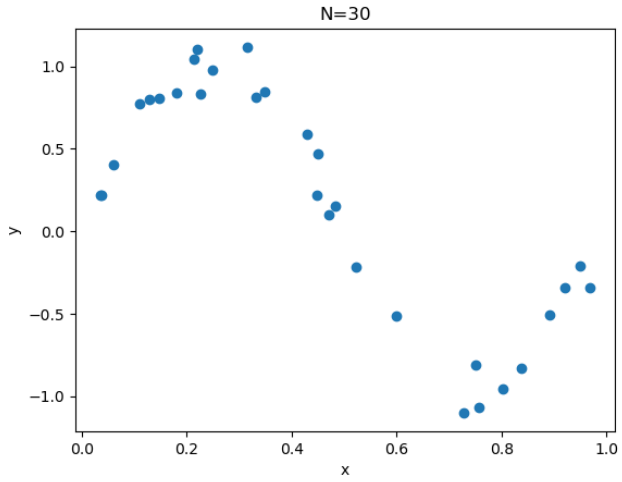
   Ordonner les données par ordre croissant.

4. *Générer les données de sorties $\{y\}$, telles que $y = \sin(2\pi x) + w$.*

   Utiliser la fonction *np.random.normal(…)* pour générer w à partir d'une distribution normale centrée d'écart type (*standard deviation*) de 0.1

5. *Visualiser le jeu de données*

   Utiliser plt.plot, plt.legend, plt.xlabel, etc.



Algorithme de descente de gradient

Calcul du gradient

- Approx (différence finie)
- Exacte (auto-différentiation)

3.2 Un réseau de neurones simple : le **MLP** (multi-layers perceptron)

```
import numpy as np
from matplotlib import pyplot as plt
import paraview.simple as simple
import vtk
from vtk.util.numpy_support import vtk_to_numpy
import os

PATH_READ = "C:/Users/peralezADM/Documents/temp"
FILE_READ = "Volume.1.20001e+01.vti" # format="appended"
PATH_WRITE = "C:/Users/peralezADM/Documents/temp"
FILE_WRITE = 'test_binary.vti' # format="binary"


#%% read .vti file (format appended) with paraview
```

4. Les réseaux de neurones convolutifs

5. Les réseaux de neurones récurrents