

# Méthodes algorithmiques II

Floriana Gargiulo - GEMASS  
floriana.gargiulo@cnrs.fr

# Programme du cours

- S1 : Introduction au langage Python
- S2 : Variables et types complexes
- S3 : Manipulation de bases de données avec Pandas
- S4 : Visualisation des données avec Matplotlib et Seaborn
- S5 : Manipulation du texte
- S6 : Extraction des données

# Programme du cours

- S7 : Web scraping
- S8 : Analyse des réseaux
- S9 : Natural Language processing
- S10 : Preparation des posters
- S11 : Exposition des posters

# Evaluations

- PRESENTATION DES POSTERS sur le cas d'étude : 15 minutes par groupe (**Séance 11**)
- "Fiches techniques" pour la reproductibilité des résultats du poster (**dernière semaine d'examen**) - sous forme d'un Git :
  - description des données et sources
  - descriptions des méthodes
  - codes commentés

# R, c'est bien... mais pourquoi aussi Python ?



- Plus simple
- Beaucoup de bibliothèque de visualisation
  - Données de plusieurs type: csv, json, sql,...
- Plus generalist (c'est un vrai language de programmation)
- Plus performant pour le data scraping et l'IA (y compris le traitement du texte)



- Plus adapté à la statistique
- Beaucoup de bibliothèque de visualisation
  - Spécialisé pour traiter des tableaux (csv)
- Plus de bibliotheques spécifiques pour la recherche

# R, c'est bien... mais pourquoi aussi Python ?



## **Méthodes digitales ou méthodes quali/quant**

- Faire du data scarping
- Analyse du text (aussi des entretiens)
- Traiter des données complexes (json, sql)
- Utiliser des techniques IA
- Modeles multi-agents



## **Méthodes statistiques (quantitatifs)**

- Traitement de données d'enquête
- Statistiques et régressions.
- Données en format tableau.
- Plus petits jeux de données

# L'écosystème Python

## Scripts and Notebooks

### 1. Python scripts (.py)

- Fichiers de code exécutés d'un seul coup
- Utiles pour l'automatisation et les pipelines reproductibles
- Idéals pour des tâches longues ou répétitives

Modélisation, extraction de données, traitements longues et lourds

### 2. Jupyter Notebooks (.ipynb)

- Mélangent code, texte et graphiques dans un même document
- Exécution cellule par cellule
- Parfaits pour apprendre, explorer des données et documenter une analyse

Exploration, visualisation, communication des résultats

### Ce que nous utiliserons dans ce cours:

Nous travaillerons principalement avec **Jupyter Notebooks**

- Pour apprendre progressivement
- Pour visualiser immédiatement les résultats
- Pour commenter et interpréter les analyses

# From RStudio to Jupyter

Syntaxe différente, logique très proche

## **1. Même logique générale**

- RStudio et Jupyter servent à écrire, exécuter et organiser du code
- On travaille par blocs de code et on visualise les résultats
- On combine analyse et documentation

## **2. Équivalences clés**

- Script R (.R) → Script Python (.py)
- R Markdown (.Rmd) → Jupyter Notebook (.ipynb)
- Console R → Cellules Jupyter
- Environnement R → Kernel Python



# Comment installer Jupyter

## 1. Télécharger et installer Anaconda

- **Télécharger Anaconda** : Va sur le site officiel d'Anaconda à [anaconda.com](https://www.anaconda.com) et télécharge le programme d'installation correspondant à ton système d'exploitation (Windows, macOS ou Linux).

<https://www.anaconda.com>

- **Installer Anaconda** : Suis les instructions du programme d'installation. Cela installera Anaconda ainsi que Python, Jupyter Notebook, et d'autres outils scientifiques comme pandas, NumPy, et matplotlib.

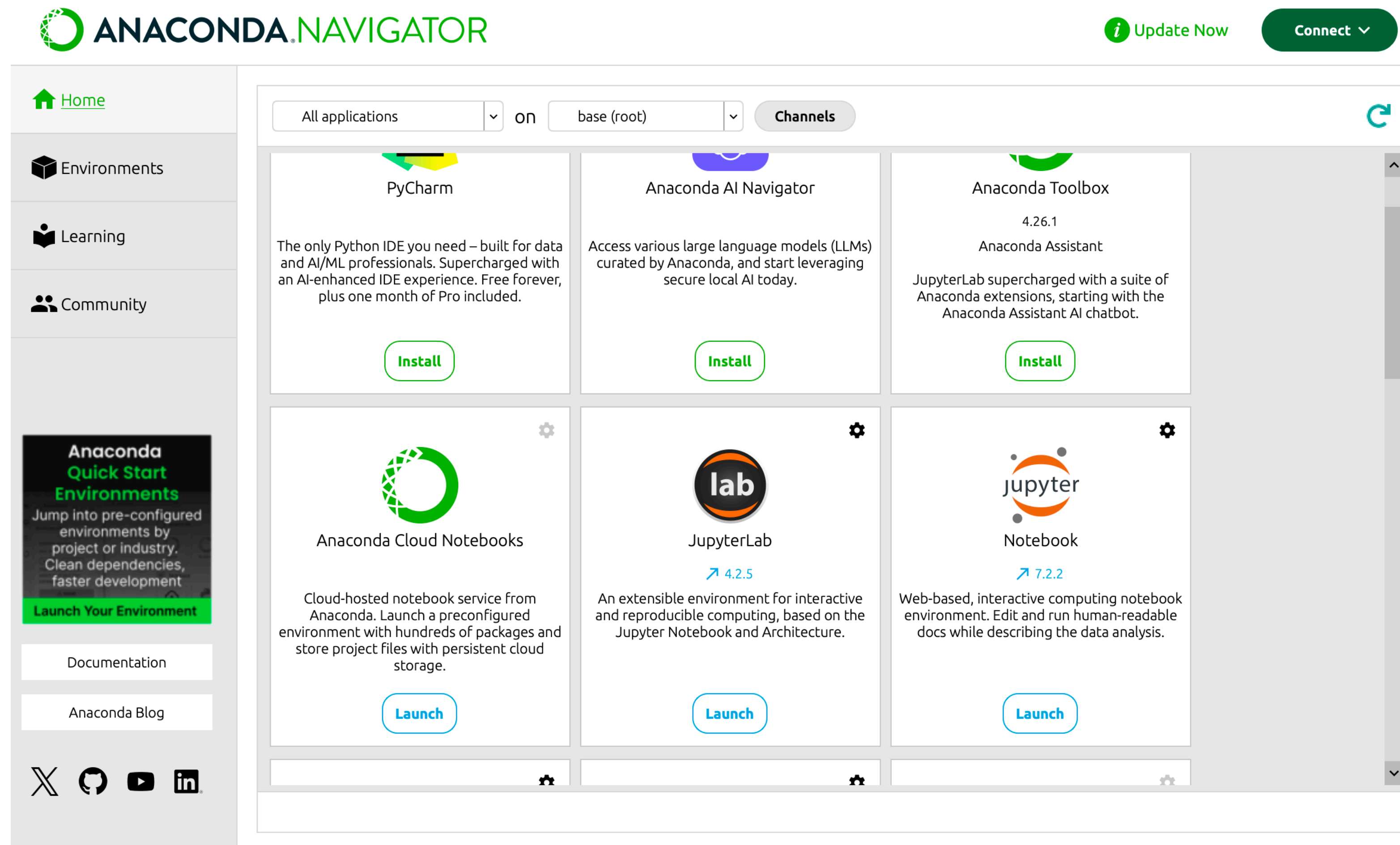
## 2. Vérifier l'installation d'Anaconda

- Après l'installation, ouvre une fenêtre de terminal (ça dépend de votre système d'exploitation)
- Tape la commande suivante pour vérifier que conda (le gestionnaire de packages d'Anaconda) est bien installé :

```
conda --version
```

- Si la commande renvoie une version de conda, c'est que l'installation s'est bien déroulée.

# Anaconda navigator



1. Démarrer Anaconda Navigator

2. Démarrer Jupyter Notebook

Option alternative:

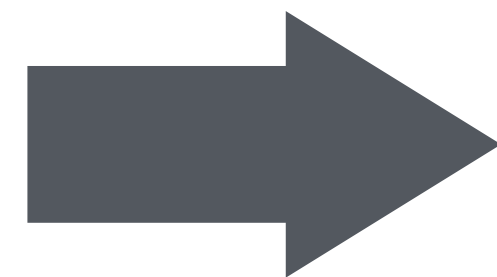
Installer et configurer **Visual Studio Code**

(plus difficile au début mais beaucoup plus simple à gérer sur le long terme)

# Le matériel du cours

**<https://github.com/FlorianaGargiulo/Python4SHS>**

Le Git sera mis à jour pendant le cours.



Exécution du premier script:  
1\_intro2Jupyter.ipynb

# Le cas d'étude

## Cartographie de la sous-culture Incel



- Les Incels (« involuntary celibates ») : terme apparu dans les années 1990 pour désigner des personnes confrontées à l'impossibilité d'établir des relations amoureuses.
- Diffusion au grand public renforcée par les médias, notamment la série Netflix *Adolescence*, qui a exposé certains codes et discours de cette sous-culture.
- Les Incels font partie de la **manosphere**, un ensemble de communautés en ligne centrées sur les questions de masculinité et de relations de genre.
- Forte présence dans le web : forums spécialisés, blogs personnels, communautés Reddit.
- Objectif de l'étude : analyser réseaux et discours pour cartographier la structure et la diffusion de cette sous-culture.

# Données web

- **Téléchargement de dumps** : fichiers massifs déjà disponibles (ex. archives Reddit, forums publics).
- **Web scraping** : extraction automatique de contenus depuis des sites web (ex. blogs, forums).
- **APIs publiques** : accès direct à des bases de données structurées (ex. Reddit API, Twitter API).
- **Flux RSS / newsletters** : suivi automatisé des publications régulières.
- **Crowdsourcing / sondages en ligne / data donation** : données collectées directement auprès des utilisateurs.

Pendant la première partie du cours

On fera des exemples pour enrichir les données de dumps

Kmetty, Zoltán, et al. "Determinants of willingness to donate data from social media platforms in the US and Hungary." (2023)