

---

# PROGRAMMATION EN LANGAGES R ET PYTHON

# OBJECTIF D'APPRENTISSAGE GÉNÉRAL

Acquérir de l'**expérience pratique** et une compréhension préliminaire des éléments suivants du code **dans les langages R et Python** :

- Variables
- Structures de données
- Opérateurs
- Énoncés et expressions
- Blocs (et portée)
- Fonctions
- Flux logique (de commande)
- Bibliothèques/trousses/modules
- Données d'entrée/données de sortie
- Interpréteurs/compilateurs

# CONTENU

1. Langages R et Python : historique et comparaison
2. Ressources de programmation
3. Exercices d'apprentissage pratique
4. Exercices et lectures complémentaires

## OBJECTIFS D'APPRENTISSAGE PERSONNELS

Selon vos connaissances préalables, il peut être ambitieux de chercher à atteindre les objectifs d'apprentissage de la présente séance dans le temps prévu.

**Sélectionnez**, en fonction de votre point de départ, **un objectif d'apprentissage raisonnable que vous voulez atteindre au cours de la présente séance.**

Vous aurez le temps de poursuivre vos travaux sur les exercices proposés au cours des laboratoires du matin de la semaine prochaine. Vous pourrez aussi mettre en pratique vos compétences en programmation lorsque nous entreprendrons nos séances de laboratoire en après-midi en février.

# LANGAGES R ET PYTHON : HISTORIQUE ET COMPARAISON

PROGRAMMATION DANS LES LANGAGES R ET PYTHON

# UN PEU D'HISTOIRE

## Langage R :

- successeur du langage S
- langage de programmation statistique élaboré par des statisticiens
- structures de données et fonctions intégrées conçues pour faciliter le traitement des données
- s'est fait connaître comme solution de rechange gratuite à code ouvert aux dispendieux logiciels statistiques

## Python :

- créé dans les années 1990, mais popularisé dans les années 2000
- conçu pour faciliter la lecture, la compréhension et l'apprentissage, par rapport aux autres langages orientés objet
- comporte une importante bibliothèque de modules à code source ouvert

# COMPARAISON

## Langage R :

- est techniquement un langage orienté objet, mais l'utilisation démontre que cet aspect n'est pas très évident
- permet une création rapide de scripts et l'exploration de données
- comporte une notation spéciale intégrée pour les modèles statistiques
- comporte un type spécial de données – le cadre de données – qui permet la manipulation d'ensembles de données

## Python :

- est un langage orienté objet
- permet la rédaction d'un code informatique structuré préconçu
- se veut un langage de programmation général
- est conçu pour créer un code facile à lire

## REMARQUE : VECTORISATION DANS UN LANGAGE INTERPRÉTÉ

Les langages interprétés évolués sont plus lents que les langages compilés de bas niveau.

Pour contourner ce problème, ces langages transfèrent parfois (en arrière-plan) certains types d'opérations à des fonctions écrites dans un langage de bas niveau (comme le langage C).

Pour profiter de cela, les communautés des langages R et Python mettent l'accent sur une certaine stratégie de programmation pour l'utilisation de listes, de vecteurs et de tableaux.

Plus particulièrement, les langages évitent de parcourir chaque élément d'une liste et utilisent plutôt souvent des fonctions spéciales qui **établissent une correspondance** entre une fonction ou une opération et chaque élément d'une liste.

Cette mise en correspondance peut aller à contre-courant de l'expérience acquise en apprenant d'autres langages.



# DES TAS DE TROUSSES ET DE MODULES!

La puissance des langages R et Python repose sur un grand nombre de troussees et de modules techniques.

Ces troussees et ces modules permettent à un programmeur de créer des opérations très complexes à partir de quelques appels de fonctions.

Ouvrons les notebooks RPackagesDemo et PythonPackagesDemo pour découvrir ces troussees et ces modules à l'œuvre.

## Available CRAN Packages By Name

[A](#)[B](#)[C](#)[D](#)[E](#)[F](#)[G](#)[H](#)[I](#)[J](#)[K](#)[L](#)[M](#)[N](#)[O](#)[P](#)[Q](#)[R](#)[S](#)[T](#)[U](#)[V](#)[W](#)[X](#)[Y](#)[Z](#)

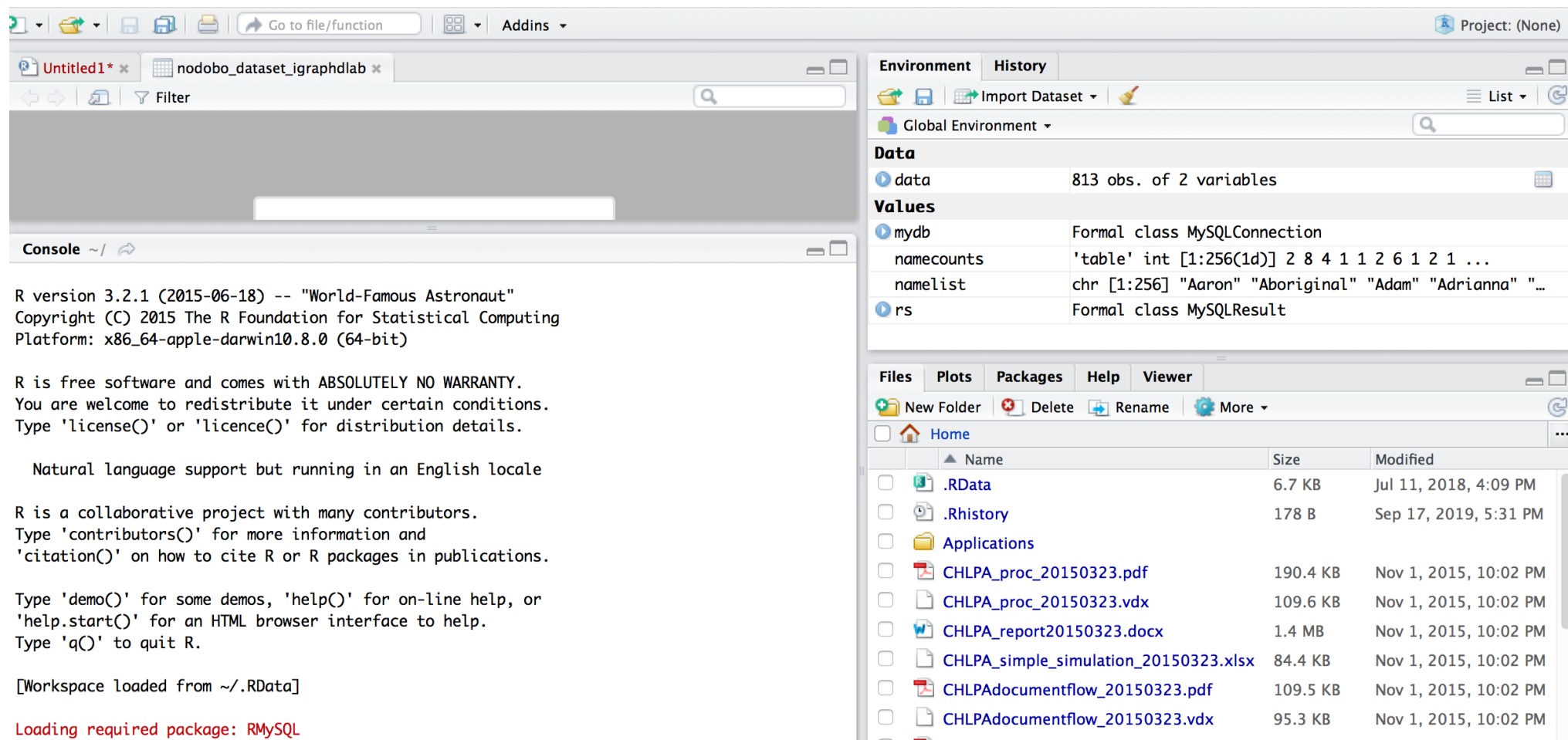
<a href="#">A3</a>	Accurate, Adaptable, and Accessible Error Metrics for Predictive Models
<a href="#">abbyyR</a>	Access to Abbyy Optical Character Recognition (OCR) API
<a href="#">abc</a>	Tools for Approximate Bayesian Computation (ABC)
<a href="#">abc.data</a>	Data Only: Tools for Approximate Bayesian Computation (ABC)
<a href="#">ABC.RAP</a>	Array Based CpG Region Analysis Pipeline
<a href="#">ABCanalysis</a>	Computed ABC Analysis
<a href="#">abcdeFBA</a>	ABCDE_FBA: A-Biologist-Can-Do-Everything of Flux Balance Analysis with this package
<a href="#">ABCOptim</a>	Implementation of Artificial Bee Colony (ABC) Optimization
<a href="#">ABCp2</a>	Approximate Bayesian Computational Model for Estimating P2
<a href="#">abcrf</a>	Approximate Bayesian Computation via Random Forests

---

# RESSOURCES DE PROGRAMMATION

PROGRAMMATION EN LANGAGES R ET PYTHON

# R STUDIO



The screenshot shows the R Studio interface. The top toolbar includes icons for file operations and a search bar. The main editor window displays a script titled 'Untitled1\*' and a package 'nodobo\_dataset\_igraphlab'. The console at the bottom shows the R version (3.2.1) and startup messages. The right sidebar contains the 'Environment' and 'History' tabs, showing a 'Global Environment' with a 'data' object (813 obs. of 2 variables) and a 'mydb' object (Formal class MySQLConnection). Below this is a 'Files' tab showing a file explorer view of the home directory, listing files like '.RData', '.Rhistory', and various PDF and VDX files.

R version 3.2.1 (2015-06-18) -- "World-Famous Astronaut"  
Copyright (C) 2015 The R Foundation for Statistical Computing  
Platform: x86\_64-apple-darwin10.8.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

Loading required package: RMySQL

**Environment** **History**

Global Environment

**Data**

data 813 obs. of 2 variables

**Values**

mydb Formal class MySQLConnection

namecounts 'table' int [1:256(1d)] 2 8 4 1 1 2 6 1 2 1 ...

namelist chr [1:256] "Aaron" "Aboriginal" "Adam" "Adrianna" "..."

rs Formal class MySQLResult

**Files** **Plots** **Packages** **Help** **Viewer**

New Folder Delete Rename More

Home

Name	Size	Modified
.RData	6.7 KB	Jul 11, 2018, 4:09 PM
.Rhistory	178 B	Sep 17, 2019, 5:31 PM
Applications		
CHLPA_proc_20150323.pdf	190.4 KB	Nov 1, 2015, 10:02 PM
CHLPA_proc_20150323.vdx	109.6 KB	Nov 1, 2015, 10:02 PM
CHLPA_report20150323.docx	1.4 MB	Nov 1, 2015, 10:02 PM
CHLPA_simple_simulation_20150323.xlsx	84.4 KB	Nov 1, 2015, 10:02 PM
CHLPAdocumentflow_20150323.pdf	109.5 KB	Nov 1, 2015, 10:02 PM
CHLPAdocumentflow_20150323.vdx	95.3 KB	Nov 1, 2015, 10:02 PM

# NOTEBOOKS JUPYTER

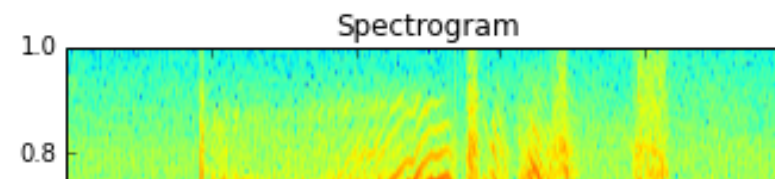
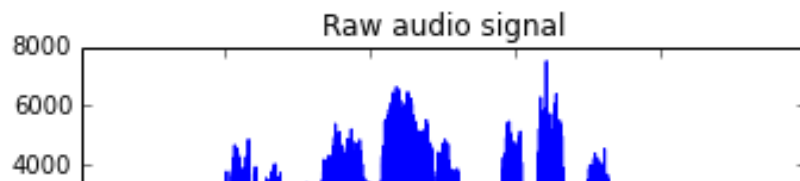
$$X_k = \sum_{n=0} x_n e^{-\frac{j2\pi}{N}kn} \quad k = 0, \dots, N-1$$

We begin by loading a datafile using SciPy's audio file support:

```
In [1]: from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin specgram routine:

```
In [2]: %matplotlib inline
from matplotlib import pyplot as plt
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
ax1.plot(x); ax1.set_title('Raw audio signal')
ax2.specgram(x); ax2.set_title('Spectrogram');
```



# NOTEBOOKS R ET PYTHON POUR LE PRÉSENT COURS

Durant le cours, nous vous remettrons de nombreux notebooks d'exemples de code.

Vous pouvez utiliser ces notebooks pour :

- découvrir ce que vous pouvez faire;
- voir de nombreux exemples de la syntaxe du langage;
- vous aider à rédiger votre propre code;
- apprendre pourquoi le code fonctionne comme il le fait et relever certaines théories sous-jacentes au code.

## HCLUST()

Let's start by clustering the entire `mtcars` dataset, using the Euclidean distance metric, and plot the result. Hierarchical clustering is implemented in the `cluster` function `hclust()`.

```
(hclustcars <- hclust(dist(mtcars)))  
plot(hclustcars)
```

Call:  
`hclust(d = dist(mtcars))`

Cluster method : complete  
Distance : euclidean  
Number of objects: 32



The output of `hclust` gives us some information about the parameters being used to create the hierarchy. In this case the distance is Euclidean (as expected) and the cluster formation strategy (the **linkage**) is complete (these are the default settings).

# RESSOURCES EN LIGNE

Stack Exchange/Stack Overflow/Cross Validated

Blogues (p. ex. R Bloggers)

Sites officiels :

- Python Software Foundation : <https://www.python.org>
- Comprehensive R Archive Network (CRAN) : <https://cran.r-project.org>



CRAN  
[Mirrors](#)  
[What's new?](#)  
[Task Views](#)  
[Search](#)

## The Comprehensive R Archive Network

### Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

# EXERCICES D'APPRENTISSAGE PRATIQUE

PROGRAMMATION EN LANGAGES R ET PYTHON

# PASSONS À LA PROGRAMMATION

Pour développer ou évaluer vos compétences en langages R et Python, faites les exercices suivants. Il n'y a aucun ordre pour faire les exercices.

Vous pouvez faire chaque exercice séparément ou écrire un seul programme qui exécutera tous les exercices.

**Vous trouverez une bonne partie du code de base dont vous aurez besoin dans les notebooks de cours de la semaine**, mais vous devrez modifier ce code ou y ajouter d'autre code pour réussir les exercices. Vous trouverez également de nombreux renseignements et du code utile sur Internet.



# EXPRESSIONS, VARIABLES, STRUCTURES DE DONNÉES ET OPÉRATEURS (I)

Créez trois variables et attribuez-leur chacune une valeur numérique.

Puis, rédigez au moins un énoncé pour exécuter les opérations suivantes sur ces variables : addition, soustraction, multiplication, division, puissance.

# EXPRESSIONS, VARIABLES, STRUCTURES DE DONNÉES ET OPÉRATEURS (2)

Créez trois variables et attribuez-leur chacune une chaîne.

Rédigez un énoncé pour unir les trois chaînes afin de former une seule chaîne.  
Rédigez du code pour imprimer la chaîne.

Rédigez du code pour tester si une sous-chaîne de votre choix fait partie de la grande chaîne.

# EXPRESSIONS, VARIABLES, STRUCTURES DE DONNÉES ET OPÉRATEURS (3)

Créez trois variables et attribuez-leur chacune une liste. Regroupez ces trois listes en une seule liste qui contient trois sous-listes distinctes (une liste de trois listes).

Créez une liste sans sous-liste (tous les éléments de chacune des trois listes font maintenant partie d'une seule liste plus longue).

Créez une quatrième liste en divisant cette longue liste en deux et attribuez une nouvelle variable à la seconde partie de la liste.

Extrayez le dernier élément de cette nouvelle liste (cet élément peut demeurer dans la liste originale ou être retiré de cette liste) et attribuez-le à une variable.

# ÉNONCÉS, BLOCS, FLUX DE COMMANDE, OPÉRATEURS LOGIQUES

Rédigez un énoncé qui contient au moins trois blocs imbriqués.

Utilisez au moins trois des options suivantes du flux de commande : if, if else, while, for, break, continue (Python seulement), next, switch.

# FONCTIONS

Rédigez une fonction qui accepte trois arguments en entrée et produit une valeur de sortie.

Appelez la fonction en utilisant des arguments de votre choix.

# BIBLIOTHÈQUES/TROUSSES/MODULES

Exécutez la commande pertinente pour afficher une liste de trousse (en langage R) ou de modules (en Python) présentement installés dans votre environnement de notebooks Jupyter.

- Indice : Utilisez Internet, les exemples dans les notebooks et les documents de cours pour vous aider à trouver la commande pertinente.

Utilisez les documents disponibles pour déterminer à quoi servent certaines de ces trousse et certains de ces modules.

- Indice : Consultez les notebooks Python et R disponibles – vous pourriez y trouver des renseignements utiles.
- Choisissez une trousse ou un module dans cette liste et chargez-le au besoin.

Rédigez du code qui utilise des fonctions et des objets fournis dans cette trousse.

# DONNÉES D'ENTRÉE/DONNÉES DE SORTIE (I)

Imprimez sur la sortie standard du notebook Jupyter (dans le présent cas, la sortie standard constitue l'espace sous une cellule de code du notebook créée lorsque vous exécutez une cellule) trois phrases de votre choix, sur trois lignes distinctes, au moyen d'un seul énoncé de code.

## DONNÉES D'ENTRÉE/DONNÉES DE SORTIE (2)

Trouvez un fichier CSV (valeurs séparées par des virgules) stocké sur votre ordinateur.

- (Indice – Il devrait y avoir un dossier désigné « Data » dans le répertoire principal du notebook).

Chargez ce fichier dans le notebook et stockez les résultats dans au moins une variable.



## DONNÉES D'ENTRÉE/DONNÉES DE SORTIE (3)

Créez un nouveau fichier et écrivez quatre lignes en format CSV dans ce fichier.

Au moyen d'un énoncé distinct, écrivez quatre autres lignes dans ce fichier, sans écraser le fichier original.

# INTERPRÉTEURS/COMPILATEURS

Rédigez assez de code pour que l'interpréteur de l'application Jupyter Notebook produise au moins cinq messages d'erreur différents.

Copiez ces messages d'erreur dans une cellule Markdown et rédigez une courte note sous chaque cellule pour expliquer le message d'erreur et indiquer la solution à chaque erreur.

# EXERCICES ET LECTURES COMPLÉMENTAIRES

PROGRAMMATION EN LANGAGES R ET PYTHON

## EXERCICES COMPLÉMENTAIRES

1. Au moyen du langage de votre choix, rédigez une fonction qui, lorsqu'elle traite un ensemble de données, relève 5 renseignements intéressants sur cet ensemble de données. Chargez un ensemble de données et exécutez la fonction sur cet ensemble de données.
2. Au moyen du langage de votre choix, rédigez deux fonctions. Les données de sortie de la première fonction devraient servir comme données d'entrée de la seconde fonction. La première fonction doit lire un ensemble de données et produire un sous-ensemble en fonction de certains critères. La seconde fonction doit lire un ensemble de données et produire des données sommaires pour chaque colonne de l'ensemble de données. Chargez un ensemble de données et exécutez les deux fonctions sur cet ensemble de données.

# LECTURES COMPLÉMENTAIRES

Meilleures ressources sur Python : <https://www.fullstackpython.com/best-python-resources.html>

Meilleures ressources sur le langage R (pour améliorer vos compétences relatives aux données) :  
<https://www.computerworld.com/article/2497464/business-intelligence/top-r-language-resources-to-improve-your-data-skills.html>