Cours de Biostatistique

3ème année cycle ingénieur

Rym Jaroudi

ESSAI

Table of Contents

SAS, R et Python

SAS

- Référence dans les industries pharmaceutiques et cliniques.
- Solide pour les grands ensembles de données et conformité réglementaire.
- Coût élevé et flexibilité limitée.

$\rightarrow$ Idéal pour les industries réglementées (pharma, essais cliniques).

R

- Bibliothèques riches en statistiques et bioinformatique.
- Visualisation avancée des données (e.g., `ggplot2`).
- Gratuit, mais courbe d'apprentissage importante.

$\rightarrow$ Parfait pour la recherche académique et la modélisation statistique avancée.

Python

- Polyvalent pour la manipulation de données, le machine learning et l'automatisation.
- Intégration avec des outils modernes (e.g., API, cloud).
- Moins de bibliothèques spécialisées en biostat que R.

$\rightarrow$ Adapté aux projets interdisciplinaires et au machine learning.

In Vivo, In Vitro et In Silico

In Vivo

Études réalisées sur des organismes vivants (humains, animaux, plantes).

- Exemple :
  - Étudier la survie des patients sous différents traitements avec des modèles statistiques (ex. : modèle de Cox).
- Avantages :
  - Résultats réalistes et transférables.
  - Permet d'explorer des interactions biologiques complexes.
- Limites :
  - Coûts élevés, contraintes éthiques.
  - Variabilité biologique importante.

In Vitro

Études réalisées en laboratoire sur des cellules, tissus ou biomolécules isolés.

- Exemple :
  - Évaluer la relation entre la concentration d'un médicament et la réponse cellulaire, en utilisant la régression logistique pour modéliser la courbe dose-réponse.
- Avantages :
  - Moins coûteux et plus rapide que l'in vivo.
  - Contrôle précis des conditions expérimentales.
- Limites :
  - Résultats moins représentatifs de l'organisme entier.
  - Difficile de modéliser des interactions complexes.

In Silico

Études basées sur des modèles informatiques, simulations et analyses de données.

- Exemple :
  - Appliquer des techniques de machine learning pour prédire l'efficacité d'un médicament à partir de données transcriptomiques (expression génétique).
- Avantages :
  - Très rapide et économique.
  - Permet d'explorer de grands ensembles de données (big data).
- Limites :
  - Fortement dépendant de la qualité des données et modèles.
  - Nécessite une validation expérimentale (in vitro ou in vivo).

Les trois approches sont **complémentaires** en biologie et en biostatistique, et une stratégie optimale consiste à les combiner pour obtenir des résultats robustes :

1. **In Silico** :
   - Génération d'hypothèses.
   - Analyse exploratoire à partir de grandes bases de données.
   - Optimisation avant expérimentation.
2. **In Vitro** :
   - Étapes initiales pour tester des hypothèses spécifiques.
   - Contrôle précis des paramètres expérimentaux.
3. **In Vivo** :
   - Validation et confirmation des résultats dans un environnement naturel.
   - Analyse des interactions biologiques complexes.

Table of Contents

Qu'est-ce que la Régression ?

Définition La régression est une méthode statistique utilisée pour analyser la relation entre une variable dépendante et une ou plusieurs variables indépendantes.

En biostatistique, la régression permet de comprendre l'impact des variables (ex. facteurs de risque) sur les résultats (ex. la progression d'une maladie).

Types de Modèles de Régression

Les modèles de régression sont des outils statistiques pour explorer la relation entre une variable dépendante Y et des variables explicatives X.

- Régression linéaire : pour les variables continues.
- Régression logistique : pour les variables binaires.
- Régression de Poisson : pour les données de comptage.
- Modèle de Cox : pour les données de survie.
- Régression multinomiale : pour les variables catégorielles sans ordre particulier.
- Régression Ordinale : pour les variables catégorielles avec un ordre entre les catégories.

Régression Linéaire

Objectif Prédire une variable continue en fonction de variables indépendantes.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

- β_0 : Intercept.
- β_i : Coefficient associé à X_i, $i = 1 \ldots p$.
- ϵ : Erreur aléatoire, $\epsilon \sim N(0, \sigma^2)$.

Exemple : Modélisation de la pression artérielle

Problème : Prédire la pression artérielle systolique (Y) en fonction de :

- L'âge (X_1)
- Le poids (X_2)

Données hypothétiques :

Individu	**Âge (X_1)**	**Poids (X_2)**	**Pression (Y)**
1	25	70	120
2	40	80	135
3	35	75	128

$$\mathbf{X} = \begin{bmatrix} 1 & 25 & 70 \\ 1 & 40 & 80 \\ 1 & 35 & 75 \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} 120 \\ 135 \\ 128 \end{bmatrix}$$

En calculant $(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$, on obtient :

$$\beta_0 = 31, \quad \beta_1 = 0.2, \quad \beta_2 = 1.2$$

$$Y = 31 + 0.2X_1 + 1.2X_2$$

Interprétation :

- Chaque année supplémentaire (X_1) augmente la pression artérielle de 0.2 mmHg.
- Chaque kilogramme supplémentaire (X_2) augmente la pression artérielle de 1.2 mmHg.

Régression Logistique

Objectif Prédire une variable binaire (0/1) en fonction de variables indépendantes.

$$\text{Logit}(P) = \ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

- $P = \Pr(Y = 1)$: Probabilité d'occurrence.
- Coefficients β_i interprétés via e^{β_i} (Odds Ratio).

Exemple : Prédire le Diabète

Problème : Modéliser la probabilité qu'un patient ait un diabète ($Y = 1$) en fonction de l'indice de masse corporelle IMC (X_1) et de l'âge (X_2).

Données hypothétiques :

Patient	IMC (X_1)	Âge (X_2)	Diabète (Y)
1	30	50	1
2	25	40	0
3	35	55	1

Code R :

```
# Données
data <- data.frame(
  IMC = c(30, 25, 35),
  Age = c(50, 40, 55),
  Diabete = c(1, 0, 1)
)

# Modèle de régression logistique
model <- glm(Diabete ~ IMC + Age, data = data, family = binomial)

# Résumé des coefficients
summary(model)
```

Code Python :

```
# Import des bibliothèques nécessaires
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
import statsmodels.api as sm

# Données
data = pd.DataFrame({
    'IMC': [30, 25, 35],
    'Age': [50, 40, 55],
    'Diabete': [1, 0, 1]
})

# Scikit-learn : Modèle de régression logistique
X = data[['IMC', 'Age']]
y = data['Diabete']
model = LogisticRegression()
model.fit(X, y)
print("Coefficients avec scikit-learn:", model.coef_)

# Statsmodels : Modèle logistique
X_sm = sm.add_constant(X)  # Ajout d'une constante pour l'intercept
model_sm = sm.Logit(y, X_sm)
result = model_sm.fit()
print(result.summary())
```

Modèle hypothétique :

$$\text{Logit}(P) = -5 + 0.1X_1 + 0.05X_2$$

où P est la probabilité que le patient ait un diabète.

Interprétation :

- Pour chaque unité supplémentaire d'IMC (X_1), les cotes de développer un diabète augmentent de $e^{0.1} \approx 1.105$, soit 10.5%.
- Pour chaque année supplémentaire (X_2), les cotes augmentent de $e^{0.05} \approx 1.051$, soit 5.1%.

Régression de Poisson

Objectif Modéliser des données de comptage (par exemple, nombre d'hospitalisations).

$$\ln(\lambda) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

- λ : Taux moyen d'occurrences.
- $Y \sim \text{Poisson}(\lambda)$.

Exemple : Modélisation du nombre moyen d'hospitalisations

Problème : Modéliser le nombre moyen d'hospitalisations (Y) en fonction de l'âge (X_1) et du sexe (X_2, codé 1 = homme, 0 = femme).

Données hypothétiques :

Patient	Âge (X_1)	Sexe (X_2)	Nombre d'hospitalisations (Y)
1	65	1	3
2	50	0	1
3	70	1	4

Code R :

```
# Données hypothétiques
data <- data.frame(
  age = c(65, 50, 70),
  sex = c(1, 0, 1),
  hospitalizations = c(3, 1, 4)
)

# Modèle de régression de Poisson
model <- glm(
  hospitalizations ~ age + sex,
  family = poisson(link = "log"),
  data = data
)

# Résultats du modèle
summary(model)
```

Code Python :

```
import statsmodels.api as sm
import pandas as pd

# Données hypothétiques
data = pd.DataFrame({
    'age': [65, 50, 70],
    'sex': [1, 0, 1],
    'hospitalizations': [3, 1, 4]
})

# Variables indépendantes (ajouter constante pour l'intercept)
X = sm.add_constant(data[['age', 'sex']])

# Variable dépendante
y = data['hospitalizations']

# Modèle de régression de Poisson
model = sm.GLM(y, X, family=sm.families.Poisson())
result = model.fit()

# Résultats du modèle
print(result.summary())
```

Modèle hypothétique :

$$\ln(\lambda) = -2 + 0.03X_1 + 0.5X_2$$

Interprétation :

- Chaque année supplémentaire augmente le taux moyen d'hospitalisations de $e^{0.03} \approx 1.03$, soit une augmentation de 3%.
- Les hommes ont un taux moyen $e^{0.5} \approx 1.65$ fois plus élevé que celui des femmes.

Modèle de Cox

Objectif Modéliser les données de survie (temps jusqu'à l'événement).

$$h(t) = h_0(t) \cdot e^{\beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p}$$

- $h(t)$: Fonction de risque instantané à un temps t.
- $h_0(t)$: Fonction de risque de base, non paramétrique.
- e^{β_i} : Ratio de risque associé à X_i (Hazard Ratio).

Exemple : Pronostic du cancer

Problème : Étudier l'impact de l'âge (X_1) et du type de traitement (X_2, codé 1 = traitement A, 0 = traitement B) sur la survie après un diagnostic de cancer.

Données hypothétiques :

Patient	Âge (X_1)	Type de traitement (X_2)	Temps de survie (mois)	Événement (1 = décès, 0 = censuré)
1	70	1	24	1
2	60	0	36	0
3	65	1	30	1

Code R :

```
# Charger les packages nécessaires
library(survival)

# Créer un jeu de données hypothétique
data <- data.frame(
  age = c(70, 60, 65),
  treatment = c(1, 0, 1),
  time = c(24, 36, 30),
  status = c(1, 0, 1)
)

# Ajuster le modèle de Cox
cox_model <- coxph(Surv(time, status) ~ age + treatment, data = data)

# Afficher les résultats du modèle
summary(cox_model)
```

Code Python :

```
# Importer les bibliothèques nécessaires
import pandas as pd
from lifelines import CoxPHFitter

# Créer un jeu de données hypothétique
data = pd.DataFrame({
    'age': [70, 60, 65],
    'treatment': [1, 0, 1],
    'time': [24, 36, 30],
    'status': [1, 0, 1]
})

# Ajuster le modèle de Cox
cph = CoxPHFitter()
cph.fit(data, duration_col='time', event_col='status')

# Afficher les résultats du modèle
cph.print_summary()
```

Modèle hypothétique :

$$h(t) = h_0(t) \cdot e^{0.02X_1 - 0.4X_2}$$

Interprétation :

- Chaque année supplémentaire augmente le risque instantané de décès de $e^{0.02} \approx 1.02$, soit une augmentation de 2 %.
- Les patients recevant le traitement A ($X_2 = 1$) ont un risque instantané réduit de $e^{-0.4} \approx 0.67$, soit une réduction de 33 % par rapport au traitement B ($X_2 = 0$).

Régression Multinomiale

Objectif Modéliser une variable dépendante catégorielle avec plus de deux niveaux.

$$\ln\left(\frac{P(Y=k)}{P(Y=K)}\right) = \beta_{0k} + \beta_{1k}X_1 + \beta_{2k}X_2 + \cdots + \beta_{pk}X_p$$

- $P(Y=k)$: Probabilité que Y prenne la catégorie k.
- K : Catégorie de référence.
- Les coefficients β_{ik} sont spécifiques à chaque catégorie k.

Exemple : Prédiction de la classe d'un échantillon basé sur les niveaux d'expression génique

Problème : Classer un échantillon en fonction de l'origine tissulaire (par exemple, foie, rein, ou cœur) à partir de l'expression de deux gènes (X_1 et X_2).

Données hypothétiques :

Échantillon	**Expression du gène X_1**	**Expression du gène X_2**	**Origine tissulaire (Y)**
1	2.3	1.1	Foie
2	1.8	0.8	Rein
3	2.5	1.4	Cœur

Code R :

```
# Charger les bibliothèques nécessaires
library(nnet)

# Données
data <- data.frame(
  X1 = c(2.3, 1.8, 2.5),
  X2 = c(1.1, 0.8, 1.4),
  Y = factor(c("Foie", "Rein", "Cœur"))
)

# Modèle de régression multinomiale
model <- multinom(Y ~ X1 + X2, data = data)

# Résultats
summary(model)
```

Code Python :

```
import pandas as pd
from sklearn.linear_model import LogisticRegression

# Données
data = pd.DataFrame({
    'X1': [2.3, 1.8, 2.5],
    'X2': [1.1, 0.8, 1.4],
    'Y': ['Foie', 'Rein', 'Cœur']
})

# Encodage de la variable cible
data['Y_encoded'] = data['Y'].map({'Foie': 0, 'Rein': 1, 'Cœur': 2})

# Modèle de régression logistique multinomiale
model = LogisticRegression(multi_class='multinomial', solver='lbfgs')
model.fit(data[['X1', 'X2']], data['Y_encoded'])

# Afficher les coefficients
print("Coefficients:", model.coef_)
print("Intercepts:", model.intercept_)
```

Modèle hypothétique :

$$\log\left(\frac{P(Y=\text{Rein})}{P(Y=\text{Foie})}\right) = -1.2 + 0.8X_1 - 0.5X_2$$

$$\log\left(\frac{P(Y=\text{Cœur})}{P(Y=\text{Foie})}\right) = -0.9 + 0.6X_1 + 0.4X_2$$

Interprétation :

- Une augmentation de l'expression de X_1 augmente la probabilité d'appartenir au tissu "Rein" ou "Cœur" par rapport au tissu "Foie".
- Une augmentation de l'expression de X_2 diminue la probabilité d'appartenir au tissu "Rein" par rapport au "Foie", mais augmente celle pour le tissu "Cœur".

Régression Ordinale

Objectif Modéliser une variable catégorielle ordonnée Y (ex. : faible, moyen, élevé).

$$\ln\left(\frac{P(Y \leq k)}{P(Y > k)}\right) = \theta_k - (\beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p)$$

- $P(Y \leq k)$: Probabilité que Y prenne une valeur inférieure ou égale à k.
- θ_k : Seuil spécifique à chaque catégorie k.
- β_i : Effet des variables prédictives X_i, supposé constant pour toutes les catégories.

Exemple : Essai clinique

Problème : Un essai clinique étudie l'impact de deux traitements sur la qualité de vie des patients atteints de cancer. La qualité de vie est mesurée sur une échelle ordinale (1 = Mauvaise, 2 = Moyenne, 3 = Bonne). L'objectif est de prédire cette qualité de vie en fonction de variables cliniques comme l'Âge (X_1) et le Type de traitement (X_2).

Données hypothétiques :

Patient	Âge (X_1)	Type de traitement (X_2)	Qualité de vie (Y)
1	60	0	1 (Mauvaise)
2	55	1	3 (Bonne)
3	70	1	2 (Moyenne)
4	65	0	2 (Moyenne)
5	63	0	1 (Mauvaise)

Code R :

```
# Installer le package nécessaire
install.packages("MASS")
library(MASS)

# Données hypothétiques
data <- data.frame(
  age = c(60, 55, 70, 65, 63),
  treatment = c(0, 1, 1, 0, 0),
  quality_of_life = c(1, 3, 2, 2, 1)
)

# Régression ordinale avec le modèle polytomique
model <- polr(factor(quality_of_life) ~ age + treatment, data = data,
 Hess = TRUE)

# Résumé du modèle
summary(model)
```

Code Python :

```
# Installer le package nécessaire
!pip install statsmodels

import pandas as pd
import statsmodels.api as sm
from statsmodels.formula.api import mnlogit

# Données hypothétiques
data = pd.DataFrame({
    'age': [60, 55, 70, 65, 63],
    'treatment': [0, 1, 1, 0, 0],
    'quality_of_life': [1, 3, 2, 2, 1]
})

# Régression ordinale
model = sm.MNLogit.from_formula('quality_of_life ~ age + treatment', data)
result = model.fit()

# Résumé du modèle
print(result.summary())
```

Modèle hypothétique :

$$\text{Logit}(P(Y \leq 1)) = -2.0 + 0.05 \cdot X_1 + 1.2 \cdot X_2$$

$$\text{Logit}(P(Y \leq 2)) = 1.5 + 0.05 \cdot X_1 + 1.2 \cdot X_2$$

Interprétation :

- $\beta_1 = 0.05$: Chaque année supplémentaire augmente légèrement la probabilité d'une meilleure qualité de vie.
- $\beta_2 = 1.2$: Le traitement expérimental ($X_2 = 1$) augmente la probabilité d'une qualité de vie meilleure par rapport au traitement standard ($X_2 = 0$).

Table of Contents

Conclusion

Comparaison des caractéristiques : SAS, R et Python

Caractéristique	**SAS**	**R**	**Python**
Coût	Élevé	Gratuit	Gratuit
Facilité d'apprentissage	Moyenne	Moyenne-Difficile	Moyenne-Difficile
Fonctionnalités statistiques	Très bonnes	Avancées	Bonnes
Visualisation des données	Basique	Excellente	Bonne
Traitement des grands jeux de données	Excellent	Modéré	Modéré
Personnalisation	Faible	Élevée	Élevée
Usage réglementaire	Excellent	Limité	Limité

Comparaison des approches : In Vivo, In Vitro et In Silico

Approches	**In Vivo**	**In Vitro**	**In Silico**
Contexte	Organismes vivants	Systèmes isolés	Simulations informatiques
Coût	Élevé	Modéré	Faible
Durée	Long	Moyen	Rapide
Représentation	Réaliste	Partielle	Théorique

Comparaison des Types de Modèles de Régression

Type de Régression	**Variable Dépendante**
Régression Linéaire	Continue
Régression Logistique	Binaire (0/1)
Régression de Poisson	Compte (Nombre d'événements)
Modèle de Cox	Temps de survie
Régression Multinomiale	Catégorielle (plus de 2 catégories sans ordre)
Régression Ordinale	Catégorielle ordonnée

L'utilisation des méthodes de régression avec des outils tq R et Python permet d'analyser efficacement des données biostatistiques, qu'elles soient issues de contextes in vivo, in vitro ou in silico et de prédire des phénomènes complexes, avec des applications clés en recherche et en santé.