

Statistique Descriptive Unidimensionnelle

Séverine Affeldt

UFR Mathématiques et Informatique
MLDS - Centre Borelli

Université de Paris

Statistique: Ensemble des méthodes permettant d'*analyser* des ensembles d'*observations* (ou données)

Statistique descriptive: ensemble des méthodes permettant de décrire les *unités statistiques* qui composent une *population*

Unité statistique: *entité abstraite* qui représente une personne, un animal, un objet...
On parle aussi d'*individu*

Echantillon: Prélèvement d'une sous-partie de la population par une méthode qui assure la représentativité de l' échantillon par rapport à la population

Variable: Une variable est une caractéristique étudiée pour une population donnée. La couleur préférée, le nombre de plantes de votre foyer ou l'âge sont des variables.

	Sexe	Age	Revenu mensuel net
individu 1	1	55	2068
individu 2	1	41	4687
individu 3	1	28	1235
individu 4	2	64	1941
individu 5	2	32	2456

Variable qualitative

Variable qualitative est représenté par des *qualités* (e.g. état civil, programme d'études). Ce type de variable s'exprime en *modalités* qui sont comme des choix de réponses à la variable étudiée.

On distingue,

- **Variable qualitative nominale** qui correspond à des noms, sans ordre précis (e.g. couleur préférée)
- **Variable qualitative ordinale** qui contient un ordre (e.g. *très satisfait, satisfait, insatisfait, très insatisfait*)

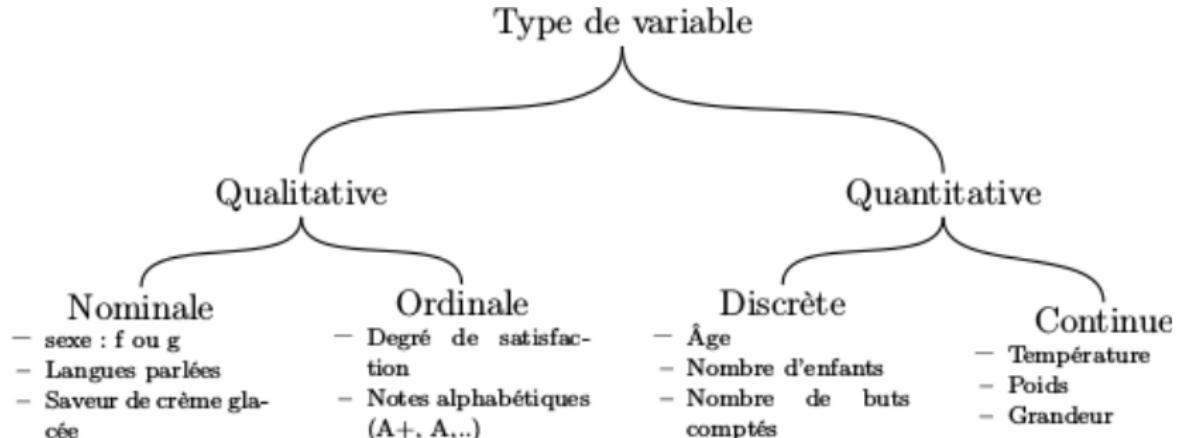
Variable quantitative

Variable quantitative est représenté par des *quantités* (e.g. âge, poids, taille). Ce type de variable s'exprime en *valeurs* qui sont comme des choix de réponses à la variable étudiée.

On distingue,

- **Variable quantitative discrète** qui correspond à des valeurs qu'on peut énumérer, il est inutile d'utiliser des classes (e.g. nombre de personnes dans le foyer, nombre de téléphone portable)
- **Variable quantitative continues** qui ont des valeurs très nombreuses, qu'on ne peut énumérer. Il est préférable de les exprimer en classes (e.g. poids).

Diagramme des différents types de variables



Variable quantitative discrète

Variable quantitative ne prenant que des valeurs entières (rarement décimales). Le nombres de valeurs distinctes est généralement faible (≤ 20)

Exemple: Age de 48 salariés

43	29	57	45	50	29	37	59	46	31	46	24	33	38	49	31
62	60	52	38	38	26	41	52	60	49	52	41	38	26	37	59
57	41	29	33	33	43	46	57	46	33	46	49	57	57	46	43

Contenu du tableau statistique

1^e colonne: les r observations x_i (non répétées!) de X par ordre croissant

puis... (selon la taille de l' échantillon)

Effectif: nombre de réplications n_i associées à la valeur x_i

Effectif cumulé: nombre de réplications cumulées $N_i = \sum_{j=1}^i n_j$ pour la valeur x_i

ou...

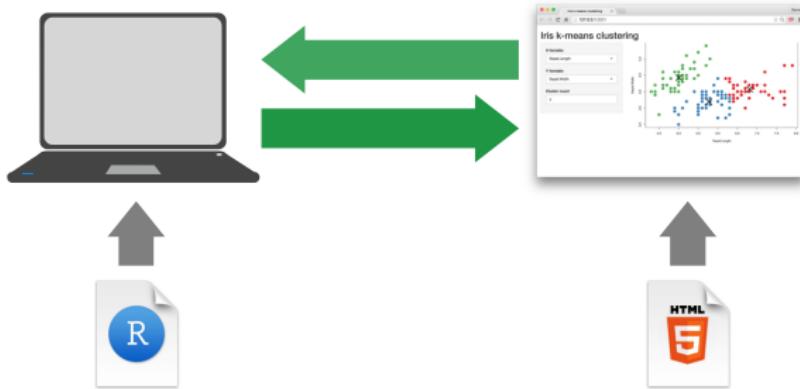
Fréquence: fréquence $f_i = \frac{n_i}{n}$ de la valeur x_i

Fréquence cumulée: fréquence cumulée $F_i = \sum_{j=1}^i f_j$ de la valeur x_i

Exemple: tableau statistique des âges de 48 salariés

x_i	n_i	N_i	$f_i(\%)$	$F_i(\%)$
24	1	1	2,08	2,08
26	2	3	4,17	6,25
29	3	6	6,25	12,50
31	2	8	4,17	16,67
33	4	12	8,33	25,00
37	2	14	4,17	29,17
38	4	18	8,33	37,50
41	3	21	6,25	43,75
43	3	24	6,25	50,00
45	1	25	2,08	52,08
46	6	31	12,50	64,58
49	3	34	6,25	70,83
50	1	35	2,08	72,91
52	3	38	6,25	79,16
57	5	43	10,42	89,58
59	2	45	4,17	93,75
60	2	47	4,17	97,92
62	1	48	2,08	100,00
Total	48	—	100,00	—

(Tableau statistique) avec Shiny



- Exemple de dashboards/interfaces Shiny
 - <https://shiny.rstudio.com/gallery/>
 - <https://shiny.rstudio.com/gallery/#user-showcase>
 - COVID19-tracker <https://shiny.rstudio.com/gallery/covid19-tracker.html>
- Se créer un compte Shiny pour publier
 - <https://www.shinyapps.io/>
- Tutorial pour la publication
 - <https://shiny.rstudio.com/articles/shinyapps.html>

Template Shiny

Installation de votre environnement R et de Shiny

1. Installation de RStudio
2. Dans le terminal de RStudio > install.packages('shiny')

Template basic pour votre application Shiny

```
library(shiny)

# Contenu de l'interface
ui <- fluidPage("Hello World")

# Commandes à executer
server <- function(input, output)

# Association interface & commandes
shinyApp(ui = ui, server = server)
```

Hello World App

The screenshot shows the RStudio interface with the following components:

- File Explorer:** Shows files: 01-template.R, 02-hist-app.R, app.R, and 01_tabStats.R.
- Code Editor:** Displays the R code for the "Hello World" application.
- Console:** Shows the execution of the code and the resulting output.
- Environment:** Shows the current environment variables.
- Plots:** Shows the plots tab, which is currently disabled.
- Packages:** Shows the available packages.
- Help:** Shows the help documentation.
- Viewer:** Shows the viewer pane.

```
library(shiny)
# Contenu de l'interface
ui <- fluidPage("Hello World")
# Commandes à executer
server <- function(input, output){}
# Association interface & commandes
shinyApp(ui = ui, server = server)
```

Console output:

```
> library(shiny); runApp('Documents/Enseignement/Paris5/statsDescriptives_M2/01_Introduction/src_shiny/01_tabStats.R')
Warning message:
R graphics engine version 12 is not supported by this version of RStudio. The Plots tab will be disabled until a newer version of RStudio is installed.

Listening on http://127.0.0.1:6364
> |
```

01_shinyTemplate.R

Hello World App

The screenshot shows the RStudio interface with the following components:

- File Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Project Bar:** Go to file/function, Addins, Project: (None).
- Code Editor:** Tabs for 01-template.R, 02-hist-app.R, app.R, and 01_tabStats.R. The app.R tab contains the following R code:


```

1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage("Hello World")
5
6 # Commandes à executer
7 server <- function(input, output){}
8
9 # Association interface & commandes
10 shinyApp(ui = ui, server = server)
11
      
```
- Environment Tab:** Shows the command runApp('Documents/Enseignement/Paris5/statsDescriptives_M2/01_Introduction/src_shiny/01_tabStats.R').
- Plots Tab:** Shows a single plot titled "Hello World".
- Console Tab:** Displays the following messages:
 - A warning about graphics engine version being unsupported by this version of RStudio, which will be disabled until a newer version of RStudio is installed.
 - Output from runApp showing the application is listening on http://127.0.0.1:6364.
 - Output from the browser showing the application is listening on http://127.0.0.1:6364.

[01_shinyTemplate.R](#)

NB: Essayez dans un browser internet

Chargement de données à partir d'un fichier

```

1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage(
5   # Bouton de recherche du fichier à charger
6   fileInput(inputId = "file1", label = "Choose CSV File",
7             accept = c("text/plain", ".csv"))
8   ),
9   # Affichage des données
10  tableOutput(outputId = "contents")
11 )
12
13 # Commandes à exécuter
14 server <- function(input, output){
15   # Commande pour le chargement de données dans 'output'
16   output$contents <- renderTable({
17
18     # Initialement, class(input$file1) = NULL
19     # Après chargement, class(input$file1) = data.frame
20     # avec les colonnes 'size', 'type', and 'datapath' columns.
21     inFile <- input$file1
22     if (is.null(inFile)) return(NULL)
23
24     read.csv(inFile$datapath, header = FALSE)
25   })
26 }
27
28 # Association interface & commandes
29 shinyApp(ui = ui, server = server)

```

[02_tabStats_loadData_simple.R](#)

TBD: Tutoriel de déploiement sur votre compte pour publier cette application simple!
→ <https://shiny.rstudio.com/articles/shinyapps.html>

input\$file1 → entrée reactive

Les entrées réactives **envoient** des notifications aux fonctions **render***

output\$contents → sortie reactive

Les sorties réactives créées par les fonctions **render*** reçoivent des notifications

Le **lien entrée/sortie** reactive se fait via les fonctions **render***

Entrées réactives

Buttons

Action

Submit

`actionButton()`
`submitButton()`

Single checkbox

Choice A

Checkbox group

- Choice 1
- Choice 2
- Choice 3

Date input

2014-01-01

Date range

2014-01-24 to 2014-01-24

`dateRangeInput()`

File input

Choose File No file chosen

`fileInput()`

Numeric input

1

`numericInput()`

Password Input

.....

`passwordInput()`

Radio buttons

- Choice 1
- Choice 2
- Choice 3

`radioButtons()`

Select box

Choice 1

`selectInput()`

Sliders



`sliderInput()`

Text input

Enter text...

`textInput()`

© CC 2015 RStudio, Inc.

<https://shiny.rstudio.com/gallery/widget-gallery.html>

Sorties réactives pour affichage dans l'interface

Function	Inserts
dataTableOutput()	an interactive table
htmlOutput()	raw HTML
imageOutput()	image
plotOutput()	plot
tableOutput()	table
textOutput()	text
uiOutput()	a Shiny UI element
verbatimTextOutput()	text

© CC 2015 RStudio, Inc.

Utilisez les fonctions *render** pour créer les types de données souhaitées

function	creates
renderDataTable()	An interactive table (from a data frame, matrix, or other table-like structure)
renderImage()	An image (saved as a link to a source file)
renderPlot()	A plot
renderPrint()	A code block of printed output
renderTable()	A table (from a data frame, matrix, or other table-like structure)
renderText()	A character string
renderUI()	a Shiny UI element

Variable réactive

(stockage intermédiaire des données)

Chargement de données dans une variable intermédiaire réactive

```

1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage(
5   # Bouton de recherche du fichier à charger
6   fileInput(inputId = "file1", label = "Choose CSV File",
7             accept = c("text/plain", ".csv")
8           ),
9   # Affichage des données
10  tableOutput(outputId = "contents"),
11  # Affichage d'un summary
12  verbatimTextOutput(outputId = "summary")
13 )
14
15 # Commandes à exécuter
16 server <- function(input, output){
17
18   data <- reactive({
19     # Initialement, class(input$file1) = NULL
20     # Après chargement, class(input$file1) = data.frame
21     # avec les colonnes 'size', 'type', and 'datapath' columns.
22     inFile <- input$file1
23     if (is.null(inFile)) return(NULL)
24
25     read.csv(inFile$datapath, header = FALSE)
26   })
27
28   # Commande pour le calcul du summary
29   output$summary <- renderPrint({ t(summary(data())) })
30   # Commande pour le chargement de données dans 'output'
31   output$contents <- renderTable({ data() })
32 }
33 # Association interface & commandes
34 shinyApp(ui = ui, server = server)

```

Les données sont chargées dans **data** quand le chemin de fichier est modifié

data() → variable reactive

Les blocks de commandes dépendant de **data()** sont exécutées lorsque **data()** est modifiée via **reactive**, après mise à jour de **fileInput**.

output\$summary → sortie reactive

Un **summary** est calculé et affiché via la fonction **renderPrint**

output\$contents → sortie reactive

Une **table** est affichée via la fonction **renderTable**

03_tabStats_loadData_saveInData.R

Chargement de données avec retard

```

1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage(
5   # Bouton de recherche du fichier à charger
6   fileInput(inputId = "file1", label = "Choose CSV File",
7             accept = c("text/plain", ".csv")),
8   # Bouton de chargement 'en retard'
9   actionButton(inputId = "go", label = "Load"),
10  # Affichage des données
11  tableOutput(outputId = "contents"),
12  # Affichage d'un summary
13  verbatimTextOutput(outputId = "summary")
14 )
15
16 # Commandes à exécuter
17 server <- function(input, output){
18
19   data <- eventReactive(input$go, {
20     # Initialement, class(input$file1) = NULL
21     # Après chargement, class(input$file1) = data.frame
22     # avec les colonnes 'size', 'type', and 'datapath' columns.
23     inFile <- input$file1
24     if (is.null(inFile)) return(NULL)
25     read.csv(inFile$datapath, header = FALSE)
26   })
27
28   # Commande pour le calcul du summary
29   output$summary <- renderPrint({ t(summary(data())) })
30   # Commande pour le chargement de données dans 'output'
31   output$contents <- renderTable({ data() })
32 }
33 # Association interface & commandes
34 shinyApp(ui = ui, server = server)

```

Les données sont chargées dans **data** après un click sur le bouton **ActionButton**

data() → variable reactive

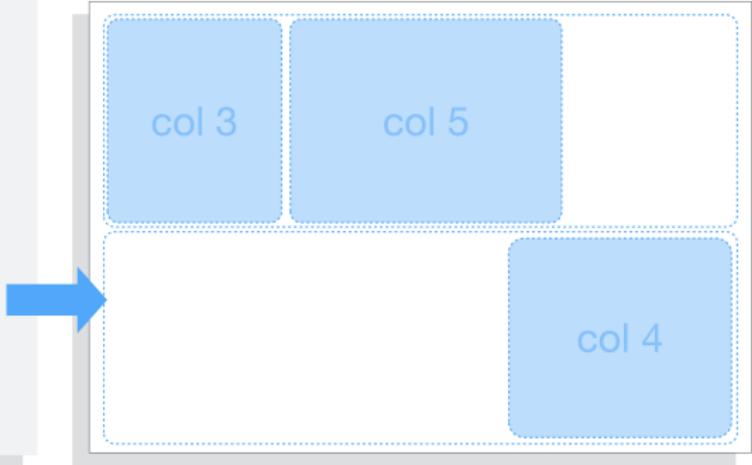
Les blocks de commandes dépendant de **data()** sont executées lorsque **data()** est modifiée via **eventReactive**, après click sur **ActionButton**.

04_tabStats_loadData_saveInData_actionBtt.R

Organiser son interface (simplement...)

Utilisez les fonction `fluidRow` et `column` dans l'ui

```
fluidPage(  
  fluidRow(  
    column(3),  
    column(5, sliderInput(...))  
)  
  ,  
  fluidRow(  
    column(4, offset = 8)  
)  
)
```



© 2014 RStudio, Inc. All rights reserved.

[05_tabStats_loadData_saveInData_layout.R](#)

Utilisez les fonctions *fluidRow* et *column* dans l'ui

```
1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage(
5
6   fluidRow(
7     column(3,
8       # Bouton de recherche du fichier à charger
9       fileInput(inputId = "file1", label = "Choose CSV File",
10                  accept = c("text/plain", ".csv"))
11      )),
12     column(4,
13       # Affichage d'un summary
14       verbatimTextOutput(outputId = "summary"))
15   ),
16
17   fluidRow(
18     column(2,
19       # Buton de chargement 'en retard'
20       actionButton(inputId = "go", label = "Load")),
21
22     column(6, offset = 1,
23       # Affichage des données
24       tableOutput(outputId = "contents"))
25   ),
26
27   textOutput(outputId = "seeVar")
28 )
```

1^e ligne → deux éléments

file1 Bouton de recherche du fichier
summary Zone d'affichage à fond gris

2^e ligne → deux éléments

go Bouton de chargement
contents Zone d'affichage de tableau

05_tabStats_loadData_saveInData_layout.R

Retour au tableau statistique (avec Shiny...)

Construisez la table dans la variable réactive tabStats

```

31 v server <- function(input, output){
32
33 v   data <- eventReactive(input$go, {
34     # Initialement, class(input$file1) = NULL
35     # Après chargement, class(input$file1) = data.frame
36     # avec les colonnes 'size', 'type', and 'datapath' columns.
37     inFile <- input$file1
38     if (is.null(inFile)) return(NULL)
39     read.csv(inFile$datapath, header = FALSE)
40   })
41
42   # Colonnes du tableau statistique
43 v   tabStats <- reactive({
44     # Calculer les effectifs et les effectifs cumulés
45     table.tmp <- as.data.frame(table(data()))
46     table.tmp <- cbind(table.tmp, cumsum(table.tmp[[2]]))
47     # Calculer les fréquences et les fréquences cumulées
48     table.tmp <- cbind(table.tmp,
49       table.tmp[[2]]/nrow(data())*100,
50       table.tmp[[3]]/nrow(data())*100)
51     # Ajouter des noms de colonnes
52     colnames(table.tmp) <- c("Ages", "Effectifs", "Effectifs Cum.",
53                               "Fréquences", "Fréquences Cum.")
54     # Renvoyer le tableau statistique
55     table.tmp
56   })
57
58   # Commande pour le calcul du summary
59   output$summary <- renderPrint({ t(summary(data())) })
60   # Commande pour le chargement de données dans 'output'
61   output$contents <- renderTable({ tabStats() })
62 }
63 # Association interface & commandes
64 shinyApp(ui = ui, server = server)

```

1^e **colonne** → données chargées

2^e **colonne** → effectifs cumulés

3^e & 4^e **colonne** → fréquences et
fréquences cumulées

Choose CSV File

Browse...

agesSalaries.csv

Upload complete

Load

V1 Min. :24.00 1st Qu.:36.00 Median :44.00 Mean :43.

Ages	Effectifs	Effectifs Cum.	Fréquences	Fréquences Cum.
24	1	1	2.08	2.08
26	2	3	4.17	6.25
29	3	6	6.25	12.50
31	2	8	4.17	16.67
33	4	12	8.33	25.00
37	2	14	4.17	29.17
38	4	18	8.33	37.50
41	3	21	6.25	43.75
43	3	24	6.25	50.00
45	1	25	2.08	52.08
46	6	31	12.50	64.58
49	3	34	6.25	70.83
50	1	35	2.08	72.92
52	3	38	6.25	79.17
57	5	43	10.42	89.58
59	2	45	4.17	93.75
60	2	47	4.17	97.92
62	1	48	2.08	100.00

06_tabStats_final.R

**Diagramme en bâtons des effectifs
et
Diagramme cumulatif des fréquences**

Nouvelle layout pour les diagrammes

```

1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage(
5
6   fluidRow(
7     column(3,
8       # Bouton de recherche du fichier à charger
9       fileInput(inputId = "file1", label = "Choose CSV File",
10                  accept = c("text/plain", ".csv"))
11     )))
12   ),
13   fluidRow(
14     column(2,
15       # Bouton de chargement 'en retard'
16       actionButton(inputId = "go", label = "Load"))
17   ),
18
19   fluidRow(
20     column(4,
21       # Zone d'affichage du diagramme en bâtons des effectifs
22       plotOutput(outputId = "effectifsDiag")),
23     column(4,
24       # Zone d'affichage du diagramme en bâtons des effectifs cumulés
25       plotOutput(outputId = "effectifsCumDiag"))
26   ),
27
28   fluidRow(
29     column(3,
30       # Zone d'affichage des données
31       tableOutput(outputId = "contents")))
32 )
33 )

```

1^e ligne → un élément

file1 Bouton de recherche du fichier

2^e ligne → un élément

go Bouton de chargement

3^e ligne → deux éléments

effectifsDiag Zone d'affichage du diagramme en bâton

effectifsCumDiag Zone d'affichage du diagramme des effectifs cumulés

4^e ligne → un élément (avec offset)

contents Zone d'affichage de la table de données

Nouvelles commandes pour la création des diagrammes

```
64  # Commande pour l'affichage du plot des effectifs
65  output$effectifsDiag <- renderPlot({
66    plot(table(data()), col ="green4", xlab ="âge", ylab ="Effectifs",
67    main ="Distribution des effectifs pour l'âge")
68  })
69
70  # Commande pour l'affichage du plot des fréquences cumulées
71  output$effectifsCumDiag <- renderPlot({
72    plot(ecdf(as.numeric(as.character(tabStats()[,1]))),
73        col ="green4", xlab ="âge", ylab ="Fréquences cumulées",
74        main ="Fréquences cumulés pour l'âge")
75  })
76 }
77 # Association interface & commandes
78 shinyApp(ui = ui, server = server)
```

ecdf(): Empirical Cumulative Distribution Function

Choose CSV File

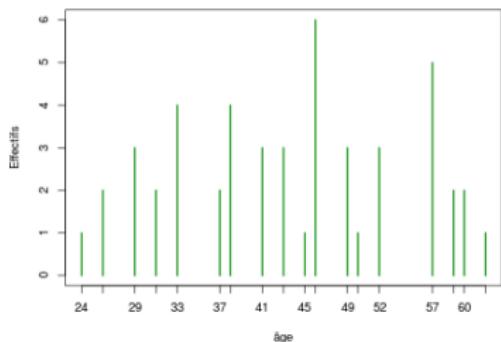
Browse...

agesSalaries.csv

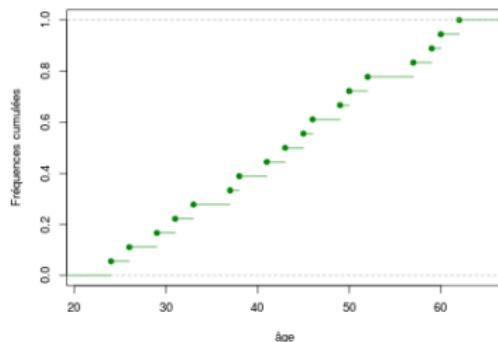
Upload complete

Load

Distribution des effectifs pour l'âge



Fréquences cumulés pour l'âge



Ages	Effectifs		Fréquences	
	Effectifs	Cum.	Fréquences	Cum.
24	1	1	2.08	2.08
26	2	3	4.17	6.25
29	3	6	6.25	12.50
31	2	8	4.17	16.67
33	4	12	8.33	25.00
37	2	14	4.17	29.17
38	4	18	8.33	37.50

07_tabStats_diagBatons.R

Notions de quantile

Quantile d'ordre α : On se donne a priori une valeur $\alpha \in [0, 1]$ et on recherche la valeur x_α telle qu'une proportion α des observations lui soient inférieurs ou égales. La valeur x_α est appelée **quantile** d'ordre α de la série.

Déciles et centiles: Un décile contient 10% des observations. Un centile contient 1% des observations (attention au nombre d'observations). On utilise généralement les rapport entre déciles (e.g., rapport du neuvième décile au premier décile)

Ages	Effectifs	Effectifs Cum.	Fréquences	Fréquences Cum.
24	1	1	2.08	2.08
26	2	3	4.17	6.25
29	3	6	6.25	12.50
31	2	8	4.17	16.67
33	4	12	8.33	25.00
37	2	14	4.17	29.17
38	4	18	8.33	37.50
41	3	21	6.25	43.75
43	3	24	6.25	50.00
45	1	25	2.08	52.08
46	6	31	12.50	64.58
49	3	34	6.25	70.83
50	1	35	2.08	72.92
52	3	38	6.25	79.17
57	5	43	10.42	89.58
59	2	45	4.17	93.75
60	2	47	4.17	97.92
62	1	48	2.08	100.00

Médiane → quantile d'ordre $\frac{1}{2}$

Colonne des fréquences cumulées $\leadsto \text{age} = 44$

Premier quartile → quantile d'ordre $\frac{1}{4}$

Colonne des fréquences cumulées $\leadsto \text{age} = 35$

Le **Troisième quartile** → quantile d'ordre $\frac{3}{4}$

Colonne des fréquences cumulées $\leadsto \text{age} = 52$

Boîte à moustaches

La boîte à moustaches résume la série à partir de ses **valeurs extrêmes**, de ses **quartiles** et de sa **médiane**. La boîte est la partie comprise entre le **premier quartile** et le **troisième quartile**. La médiane est le trait horizontal à l'intérieur de la boîte. Les **moustaches** lient les valeurs extrêmes aux quartiles.

Nouvelle layout de notre application

```
3 # Contenu de l'interface
4 ui <- fluidPage(
5
6     fluidRow(
7         column(3,
8             # Bouton de recherche du fichier à charger
9             fileInput(inputId = "file1", label = "Choose CSV File",
10                 accept = c("text/plain", ".csv")
11             )))
12     ),
13     fluidRow(
14         column(2,
15             # Bouton de chargement 'en retard'
16             actionButton(inputId = "go", label = "Load"))
17     ),
18
19     fluidRow(
20         column(4,
21             # Zone d'affichage du diagramme en bâtons des effectifs
22             plotOutput(outputId = "effectifsDiag")),
23         column(4,
24             # Zone d'affichage du diagramme en bâtons des effectifs cumulés
25             plotOutput(outputId = "effectifsCumDiag"))
26     ),
27
28     fluidRow(
29         column(4,
30             # Zone d'affichage de la boîte à moustaches
31             plotOutput(outputId = "boiteMoustaches")),
32         column(4,
33             # Zone d'affichage des données
34             tableOutput(outputId = "contents"))
35     )
36 )
```

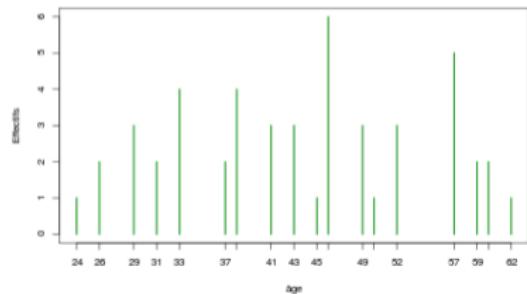
Nouvelles commandes pour la création des diagrammes

```
80  # Commande pour l'affichage de la boîte à moustaches
81  output$boiteMoustaches <- renderPlot({
82    # Boîte à moustaches
83    boxplot( data(), col = grey(0.8),
84             main = "Age des salariés",
85             ylab = "Age", las = 1)
86    # Affichage complémentaires en Y des différents âges
87    rug(data()[,1], side = 2)
88  })
89 }
90 # Association interface & commandes
91 shinyApp(ui = ui, server = server)
```

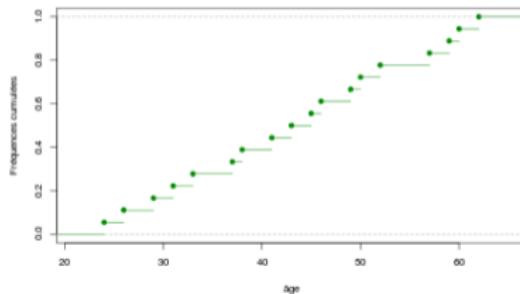
Choose CSV File

 agesSalaries.csv

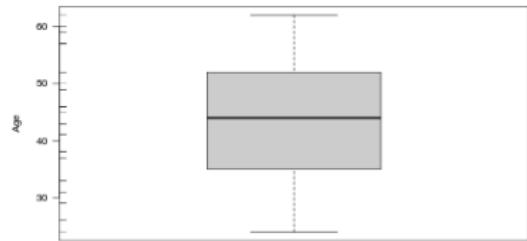
Distribution des effectifs pour l'âge



Fréquences cumulés pour l'âge



Age des salariés



Ages	Effectifs	Effectifs Cum.	Fréquences	Fréquences Cum.
24	1	1	2.08	2.08
26	2	3	4.17	6.25
29	3	6	6.25	12.50
31	2	8	4.17	16.67
33	4	12	8.33	25.00
37	2	14	4.17	29.17
38	4	18	8.33	37.50
41	3	21	6.25	43.75
43	3	24	6.25	50.00
45	1	25	2.08	52.08
46	6	31	12.50	64.58
49	3	34	6.25	70.83

08_tabStats_boiteMoustaches.R

Caractéristiques de tendance centrale et de dispersion

Caractéristiques de tendance centrale

Objectif: fournir un ordre de grandeur de la série

Médiane (définie précédemment)

Moyenne: La moyenne pour une variable quantitative discrète X est définie par,

$$\bar{x} = \frac{1}{n} \sum_{i=1}^r n_i x_i = \sum_{i=1}^r f_i x_i$$

NB: La moyenne est un indicateur plus naturel et moins ambigu que la médiane. Mais la moyenne peut-être faussée dans le cas de séries très dissymétriques.

Caractéristiques de dispersion

Objectif: préciser la variabilité, ie. résumer l'éloignement des observations par rapport à leur tendance centrale.

Etendue: Ecart entre la plus grande et la plus petite valeur, $x_r - x_1$. Cette grandeur est peu utilisée car peu fiable étant donné sa dépendance aux valeurs extrêmes. Elle peut néanmoins donner une première idée de la dispersion.

Intervalle interquartile: Ecart entre le troisième et le premier quartiles, ie. longueur de la boîte à moustaches, $x_{\frac{3}{4}} - x_{\frac{1}{4}}$. Cette grandeur est fiable.

Ecart-type: C'est la grandeur la plus utilisée. Il s'agit de la racine carrée, s_X , de la variance, $\text{var}(X)$. La variance est la moyenne des carrés des écarts à la moyenne. L'écart-type résume donc bien la dispersion de la série autour de la tendance centrale

$$\text{var}(X) = s_X^2 = \frac{1}{n} \sum_{i=1}^r n_i(x_i - \bar{x})^2 = \frac{1}{n} \sum_{i=1}^r n_i(x_i)^2 - (\bar{x})^2$$

Ages	Effectifs	Effectifs Cum.	Fréquences	Fréquences Cum.
24	1	1	2.08	2.08
26	2	3	4.17	6.25
29	3	6	6.25	12.50
31	2	8	4.17	16.67
33	4	12	8.33	25.00
37	2	14	4.17	29.17
38	4	18	8.33	37.50
41	3	21	6.25	43.75
43	3	24	6.25	50.00
45	1	25	2.08	52.08
46	6	31	12.50	64.58
49	3	34	6.25	70.83
50	1	35	2.08	72.92
52	3	38	6.25	79.17
57	5	43	10.42	89.58
59	2	45	4.17	93.75
60	2	47	4.17	97.92
62	1	48	2.08	100.00

Moyenne $\rightarrow \bar{x} = \frac{1}{n} \sum_{i=1}^r n_i x_i \approx 43,6 \text{ ans}$

Variance $\rightarrow s_X^2 = \frac{1}{n} \sum_{i=1}^r n_i (x_i - \bar{x})^2 \approx 109,7760$

Ecart-type $\rightarrow \sqrt{s_X^2} \approx 10,5 \text{ ans}$

Nouvelle layout de notre application

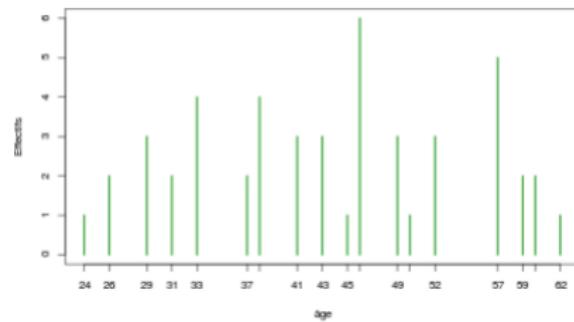
```
1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage(
5
6   fluidRow(
7     column(4,
8       # Bouton de recherche du fichier à charger
9       fileInput(inputId = "file1", label = "Choose CSV File",
10                  accept = c("text/plain", ".csv"))
11     )),
12     column(2,
13       # Buton de chargement 'en retard'
14       actionButton(inputId = "go", label = "Load"))
15   ),
16
17   fluidRow(
18     column(6,
19       # Zone d'affichage du diagramme en bâtons des effectifs
20       plotOutput(outputId = "effectifsDiag")),
21     column(6,
22       # Zone d'affichage du diagramme en bâtons des effectifs cumulés
23       plotOutput(outputId = "effectifsCumDiag"))
24   ),
25
26   fluidRow(
27     column(6,
28       # Zone d'affichage de la boîte à moustaches
29       plotOutput(outputId = "boiteMoustaches")),
30     column(4, offset = 2,
31       # Zone d'affichage dispersion / tendance centrale
32       tableOutput(outputId = "centreDisp"))
33   )
34 )|
```

Nouvelles commandes le calcul des caractéristiques

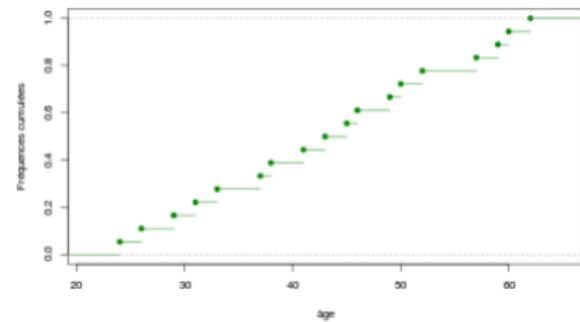
```
64 v tabCentreDisp <- reactive({  
65     # Noms des caractéristiques  
66     names.tmp <- c("Maximum", "Minimum", "Moyenne", "Médiane",  
67             "1e quartile", "3e quartile", "Variance", "Ecart-type")  
68     # Calcul des caractéristiques  
69     summary.tmp <- c(max(data()[,1]), min(data()[,1]), mean(data()[,1]), median(data()[,1]),  
70                 quantile((data()[,1]))[2], quantile((data()[,1]))[4],  
71                 var(data()[,1]), sqrt(var(data()[,1])))  
72     # Ajout des noms au vecteur de valeurs  
73     summary.tmp <- cbind.data.frame(names.tmp, summary.tmp)  
74     # Ajout des noms de colonnes  
75     colnames(summary.tmp) <- c("Caractéristique", "Valeur")  
76  
77     summary.tmp  
78 })  
79 # Commande pour le chargement de données dans 'output'  
80 #output$contents <- renderTable({ tabStats() })  
81 output$centreDisp <- renderTable({tabCentreDisp()})
```

Choose CSV File

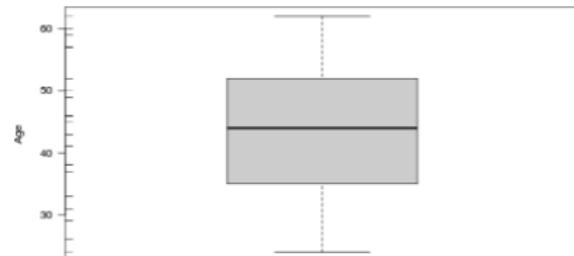
Distribution des effectifs pour l'âge



Fréquences cumulés pour l'âge



Age des salariés



Caractéristique	Valeur
Maximum	62.00
Minimum	24.00
Moyenne	43.62
Médiane	44.00
1 ^e quartile	36.00
3 ^e quartile	52.00
Variance	112.11
Ecart-type	10.59

09_tabStats_centreDispersion.R

Variable quantitative continue

Les observations associées à une variable quantitative continue sont des **intervalles**. Les valeurs ont été divisé en r intervalles contigus appelés **classes**. Ce type de variables est utile pour un grand nombres d'observations distinctes ou lorsqu'une valeur précise ne peut être recueillie.

Intervalles: Ils sont notés $\{(b_0; b_1), \dots, (b_{r-1}; b_r)\}$. Les valeurs (b_{i-1}, b_i) sont les **bornes** de la classe i , la quantité $x_i = \frac{(b_i + b_{i-1})}{2}$ est le **centre**. L'**amplitude** de la classe i est $a_i = (b_i - b_{i-1})$.

Récapitulatif

	Var. Quantitative Discrète	Var. Quantitative Continue
Tableau statistique	✓	✓
Répartition	Diagramme Bâtons	Histogramme
Répartition cumulée	Diagramme cumulatif	Courbe cumulative
Caractéristiques	✓	✓ (<i>quelques adaptations</i>)

Les données

On s'intéresse aux statistiques macroéconomiques de 27 pays de l'UE et de huit de ses principaux partenaires commerciaux. Un *individu* est un pays. On a donc 35 individus (ou unité statistiques).

Informations macroéconomiques des pays de l'UE à 27 et de huit de ses partenaires commerciaux

Variable	Description
Monnaie	la monnaie du pays
Superficie	la superficie du pays en km^2
Adhesion	l'année d'adhésion du pays à l'UE
nbrFrontieres	le nombre de frontière terrestres avec d'autres pays de l'UE à 27
Population	nombre d'habitants estimés en juillet 2007
PIB	PIB en milliards de dollars US en 2007
PIBppa	PIB en milliards de dollars PPA en 2007
Age	Age médian estimé en juillet 2007
Fecondite	indice de fécondité en nombre d'enfants par femme estimé en juillet 2007

NB: Le PIC exprimé en dollar PPA (dollar de Parité de Pouvoir d'Achat) reflète mieux la réalité car il élimine les fluctuations de taux de change.

Tableau statistique de la variable quantitative continue fecondite

La variable fecondite est divisée en 11 classes de largeur 0.2

Fecondite	Effectifs	Effectifs Cum.	Fréquences	Fréquences Cum.	x_i
($r = 1$) de 0.8 à 1.0	1	1	2,86	2,86	0.9
($r = 2$) de 1.0 à 1.2	1	2	2,86	5,71	1.1
($r = 3$) de 1.2 à 1.4	15	17	42,86	48,57	1.3
($r = 4$) de 1.4 à 1.6	5	22	14,29	62,86	1.5
($r = 5$) de 1.6 à 1.8	8	30	22,86	85,71	1.7
($r = 6$) de 1.8 à 2.0	3	33	8,57	94,29	1.9
($r = 7$) de 2.0 à 2.2	1	34	2,86	97,14	2.1
($r = 8$) de 2.2 à 2.4	0	34	0,00	97,14	2.3
($r = 9$) de 2.4 à 2.6	0	34	0,00	97,14	2.5
($r = 10$) de 2.6 à 2.8	0	34	0,00	97,14	2.7
($r = 11$) de 2.8 à 3.0	1	35	2,86	100,00	2.9

Histogramme et courbe cumulative

Histogramme

Chaque classe x_i est représentée par un rectangle qui indique l'**effectif** ou la **densité de fréquence**. Pour la classe x_i , la base du rectangle est définie par les bornes $(b_{i-1}; b_i)$. Si l'histogramme représente les effectifs, la hauteur du rectangle est simplement l'effectif n_i de la classe. Si l'histogramme représente la densité de fréquence, la hauteur du rectangle représente la ratio entre la fréquence et la base du rectangle, $\frac{n_i/n}{(b_i - b_{i-1})}$. Dans ce dernier cas, l'aire totale de l'histogramme vaut 1.

Courbe cumulative

Chaque classe x_i est représentée par un point dont la valeur en abscisse correspond à la borne supérieure b_i de la classe et la valeur en ordonnée correspond à l'effectif ou à la fréquence.

Nouvelle layout de notre application

```
3 # Contenu de l'interface
4 ui <- fluidPage(
5
6   fluidRow(
7     column(3,
8       # Bouton de recherche du fichier à charger
9       fileInput(inputId = "file1", label = "Choose CSV File",
10      accept = c("text/plain", ".csv"))
11    )))
12  ),
13  fluidRow(
14    column(2,
15      # Buton de chargement 'en retard'
16      actionButton(inputId = "go", label = "Load"))
17  ),
18
19  fluidRow(
20    column(4,
21      # Zone d'affichage de l'histogramme
22      plotOutput(outputId = "effectifsHist")),
23    column(4,
24      # Zone d'affichage de l'histogramme
25      plotOutput(outputId = "effectifsHistFreqDens"))
26  ),
27
28  fluidRow(
29    column(4,
30      # Zone d'affichage de la courbe cumulative
31      plotOutput(outputId = "effectifsCumCurve")),
32    column(4,
33      # Zone d'affichage de la courbe cumulative
34      plotOutput(outputId = "freqCumCurve"))
35  )
36 )
```

Nouvelles commandes pour les histogrammes

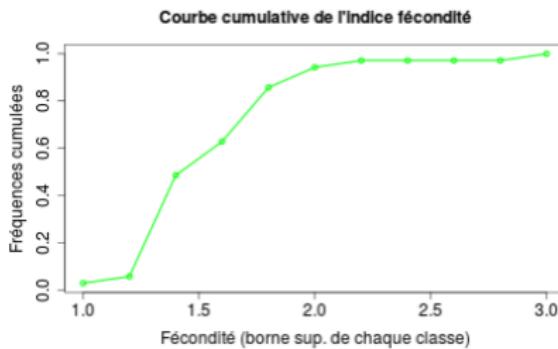
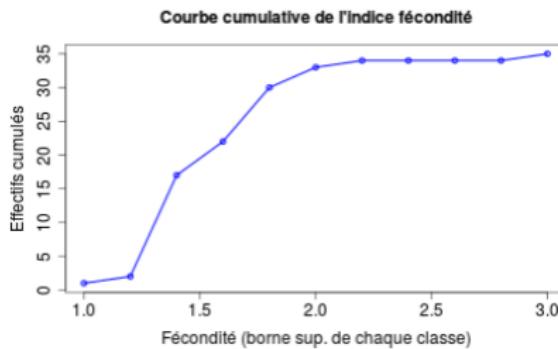
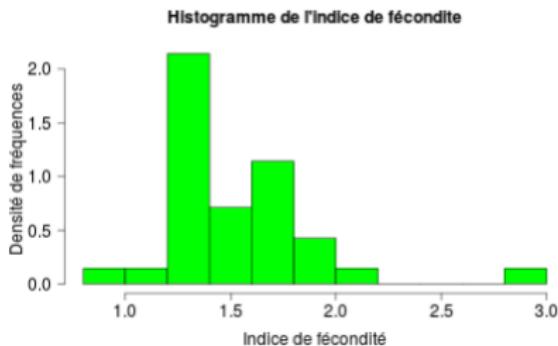
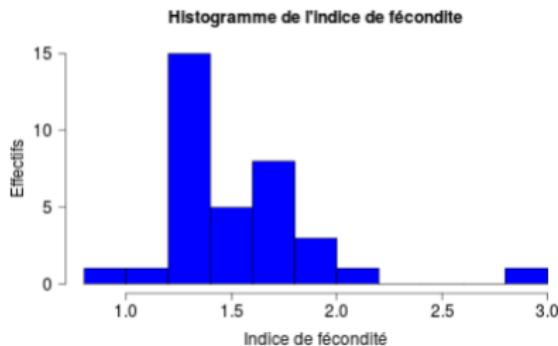
```
38 # Commandes à exécuter|
39 v server <- function(input, output){
40
41 v   data <- eventReactive(input$go, {
42     inFile <- input$file1
43     if (is.null(inFile)) return(NULL)
44     read.csv(inFile$datapath, header = TRUE)
45   })
46
47   # Récupération des valeurs fecondite
48 v   fecondite <- reactive({
49     if(!"fecondite" %in% colnames(data())) return(NULL)
50     data()$fecondite
51   })
52
53   # Histogrammes
54 v   # ----
55 v   output$effectifsHist <- renderPlot({
56     # Histogramme des effectifs
57     hist( fecondite(), freq = TRUE, cex.axis = 1.5, cex.main = 1.5,
58           main = "Histogramme de l'indice de fécondité", col = "blue",
59           xlab = "Indice de fécondité", ylab = "Effectifs", las = 1,
60           breaks = seq(0.8, 3, by = 0.2), right = FALSE, cex.lab = 1.5)
61   })
62
63 v   output$effectifsHistFreqDens <- renderPlot({
64     # Histogramme des densités de fréquences
65     hist( fecondite(), freq = FALSE, cex.axis = 1.5, cex.main = 1.5,
66           main = "Histogramme de l'indice de fécondité", col = "green",
67           xlab = "Indice de fécondité", ylab = "Densité de fréquences", las = 1,
68           breaks = seq(0.8, 3, by = 0.2), right = FALSE, cex.lab = 1.5)
69   })
```

Nouvelles commandes pour les courbes cumulatives

```
72  # Courbe cumulative
73  # ----
74  output$effectifsCumCurve <- renderPlot({
75      # Récupération des infos à partir de l'histogramme
76      tmp.hist <- hist( fecondite(), plot = FALSE,
77                          breaks = seq(0.8, 3, by = 0.2),
78                          right = FALSE)
79      # Courbe cumulative (effectifs)
80      plot(x = tmp.hist$breaks[-1], y = cumsum(tmp.hist$counts),
81            xlab = "Fécondité (borne sup. de chaque classe)",
82            ylab = "Effectifs cumulés", cex.axis = 1.5, cex.lab = 1.5,
83            main = "Courbe cumulative de l'indice fécondité",
84            type = "o", col = "blue", lwd = 2, cex.main = 1.5)
85
86  })
87
88  output$freqCumCurve <- renderPlot({
89      # Récupération des infos à partir de l'histogramme
90      tmp.hist <- hist( fecondite(), plot = FALSE,
91                          breaks = seq(0.8, 3, by = 0.2),
92                          right = FALSE)
93      # Courbe cumulative (fréquences)
94      plot(x = tmp.hist$breaks[-1], y = cumsum(tmp.hist$density*rep(0.2, 11)),
95            xlab = "Fécondité (borne sup. de chaque classe)",
96            ylab = "Fréquences cumulées", cex.axis = 1.5, cex.lab = 1.5,
97            main = "Courbe cumulative de l'indice fécondité",
98            type = "o", col = "green", lwd = 2, cex.main = 1.5)
99
100 })
101 }
102 # Association interface & commandes
103 shinyApp(ui = ui, server = server)
```

Choose CSV File

macroeconomie_UE.csv

[10_tabStats_macoEco_hist.R](#)

Liste de *reactiveValues* avec Shiny

Les deux histogrammes que nous avons affichés se basent sur les mêmes informations. Nous pouvons définir **une seule zone d'affichage**, et **modifier l'affichage** (sans calculer à nouveau les informations!) *via un click bouton*.

Effectifs

Choose CSV File

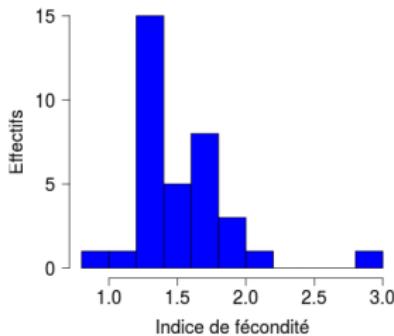
Browse... macroeconomie_UE.csv
Upload complete

Load

fréquences

effectifs

Histogramme de l'indice de fécondité



Densité de fréquences

Choose CSV File

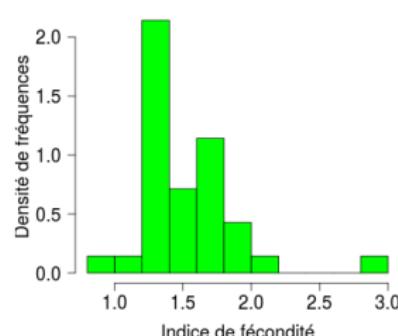
Browse... macroeconomie_UE.csv
Upload complete

Load

fréquences

effectifs

Histogramme de l'indice de fécondité



Nouvelle layout de notre application

```
1 library(shiny)
2
3 # Contenu de l'interface
4 ui <- fluidPage(
5
6   fluidRow(
7     column(3,
8       # Bouton de recherche du fichier à charger
9       fileInput(inputId = "file1", label = "Choose CSV File",
10      accept = c("text/plain", ".csv")
11    )))
12  ),
13  fluidRow(
14    column(2,
15      # Bouton de chargement 'en retard'
16      actionButton(inputId = "go", label = "Load")),
17    column(1, align="center",
18      # Bouton de mise à jour de la liste rv
19      actionButton(inputId = "frequences", label = "fréquences")),
20    column(1, align="center",
21      # Bouton de mise à jour de la liste rv
22      actionButton(inputId = "effectifs", label = "effectifs"))
23  ),
24
25  fluidRow(
26    column(4,
27      # Zone d'affichage de l'histogramme
28      plotOutput(outputId = "hist")))
29 )
30 )
```

1^e ligne → un élément

file1 Bouton de recherche du fichier

2^e ligne → trois éléments

go Bouton de chargement

effectifs Bouton 'effectifs'

frequences Bouton 'frequencies'

3^e ligne → un élément

hist Zone d'affichage de l'histogramme

```

46  # On initialise liste de valeurs réactives
47 #
48 rv <- reactiveValues(hist_isFreq = TRUE,
49                      hist_yLabel = "Effectifs",
50                      hist_col = "blue")
51 # On observe les clicks
52 observeEvent(input$effectifs,{
53   rv$hist_isFreq <- TRUE;
54   rv$hist_yLabel <- "Effectifs";
55   rv$hist_col <- "blue";
56 })
57 observeEvent(input$frequences,{
58   rv$hist_isFreq <- FALSE;
59   rv$hist_yLabel <- "Densité de fréquences";
60   rv$hist_col <- "green"
61 })
62
63 # Histogramme
64 #
65 output$hhist <- renderPlot({
66   hist(fecondite(), freq = rv$hist_isFreq, cex.axis = 1.5, cex.main = 1.5,
67         main = "Histogramme de l'indice de fécondité", col = rv$hist_col,
68         xlab = "Indice de fécondité", ylab = rv$hist_yLabel, las = 1,
69         breaks = seq(0.8, 3, by = 0.2), right = FALSE, cex.lab = 1.5)
70 })
71 }

```

reactiveValues → liste de valeurs réactives **rv**

hist_isFreq Choix de la distribution en effectifs ou en fréquences

hist_yLabel Choix du y label en fonction de effectifs ou fréquences

hist_col Couleur de l'histogramme

observeEvent → mise à jour de la liste **rv** selon le click

[11_tabStats_macroEco_listOfRV.R](#)

Caractéristiques numériques

Dans le cas d'un variable quantitative continue, les formules de la variance, de l'écart-type et de la moyenne peuvent se baser sur le centre des classes x_i plutôt que sur les observations.

Moyenne

On utilise les centres x_i et les fréquences f_i des classes, $\bar{x} = \sum_{i=1}^{11} x_i f_i$.

Exemple Pour l'indice de fécondité, $\bar{x} = \frac{152.318}{100} \simeq 1.52$

Médiane

Pour l'indice de fécondité, la première classe à dépasser 50% est la 4^e classe ($b_3 = 1,4$ et $b_4 = 1,6$). On peut simplement prendre le centre de cette classe, *i.e.* 1,3, pour médiane.

On peut également faire une estimation plus précise à l'aide d'une interpolation linéaire dans la classe médiane,

$$x_{\frac{1}{2}} = b_{i-1} + a_i \frac{50 - F_i}{F_i - F_{i-1}} = b_{i-1} + a_i \frac{50 - F_i}{f_i}$$

Exemple Pour l'indice de fécondité, $x_{\frac{1}{2}} = b_2 + a_3 \frac{50 - F_2}{f_3} = 1.4 + 0.2 \frac{50 - 48.57}{14.29} \simeq 1.42$

NB: La médiane est légèrement plus faible que la moyenne, indiquant une série concentrée sur les petites valeurs.

Histogramme et discréétisation

Crabs frontal lob size

```
1 library(MASS)
2
3 # Charger les données dans l'environnement
4 # (Morphological Measurements on Leptograpsus Crabs)
5 data(crabs)
6 # Visualisation de la structure de la table
7 str(crabs)
8
9 # Répartition des données 'Frontal lob size' selon les espèces
10 # (species - "B" or "O" for blue or orange)
11 dataB <- crabs[which(crabs$sp=="B"), 4]
12 dataO <- crabs[which(crabs$sp=="O"), 4]
13
14 # Histogramme avec plot
15 hist(dataB, col = "magenta")
16 # Histogramme sans plot
17 hist(dataB, plot = FALSE)
```

12_hist_discretisation.R

Les effectifs

\$counts \Rightarrow 1 10 18 20 24 19 7 1

Les intervalles

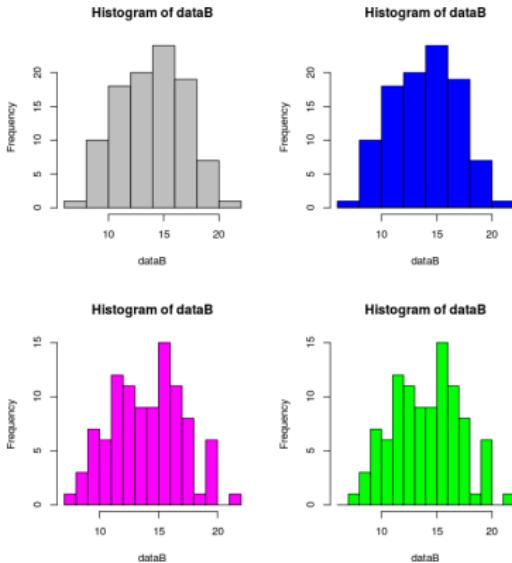
\$breaks \Rightarrow 6 8 10 12 14 16 18 20 22

Comment choisir au mieux 'breaks' ?

Comment choisir au mieux 'breaks' ?

```
19 # Différents histogrammes suivant les breaks
20 par(mfrow=c(2,2))
21 hist(dataB, col = "grey")
22 hist(dataB, col = "blue", breaks = seq(from = 6, to = 22, by = 2))
23 hist(dataB, col = "magenta", breaks = 20)
24 hist(dataB, col = "green", breaks = seq(from = 6, to = 22, by = 1))
25 par(mfrow=c(1,1))
```

12_hist_discrétisation.R



Choisir les bornes (approches simples)

Intervalles de largeurs égales On choisit le nombre de classes K , et on construit des intervalles de largeur $a = \frac{\max - \min}{K}$. Les bornes sont donc $b_1 = \min + a$, $b_2 = b_1 + a$, etc...

- Calcul simple et rapide
- Forme de la distribution conservée
- Choix arbitraire de K
- Sensibilité aux points extrêmes
- Intervalles possiblement vides ou quasi-vides

Intervalles de fréquences égales On choisit le nombre de classes K , et on construit des intervalles de même fréquence (eg, quantile d'ordre 0.25)

- Calcul simple et rapide
- *Lissage* des points extrêmes
- Intervalles avec un nombre déterminé d'individus
- Egalisation de la distribution
- Choix arbitraire de K
- Seuil ne tenant pas compte de la proximité des individus

Calcul de K

Formule de Sturges La taille des classes est basée sur le nombre d'observations, n . On fait l'hypothèse d'une distribution symétrique, binomiale ou gaussienne.

$$\text{Nombre de classes} = 1 + \log_2(n)$$

Formule de Scott La taille des classes est basée sur la variance (écart-type σ) et le nombre d'observations, n .

$$\text{Nombre de classes} = \frac{\max - \min}{3.5 \times \sigma \times n^{-1/3}}$$

Formule de Freedman-Diaconis Le nombre de classes est basés sur l'espace inter-quartile $IQ(x)$ et le nombre d'observations, n . Cette formule est plus adaptée aux distributions asymétriques que la formule de Sturges.

$$\text{Nombre de classes} = \frac{\max - \min}{2 \times IQ \times n^{-1/3}}$$

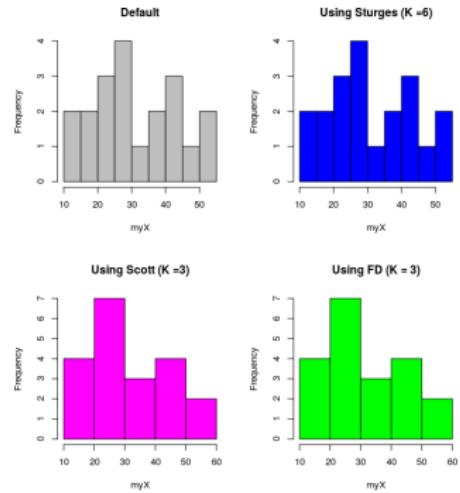
Différentes estimations du nombre de classes K

```

27 # Différents histogrammes suivant le nombre de classes
28 myX = c(14, 15, 17, 19, 21, 23, 25, 26, 27, 28, 29, 31,
29   38, 39, 41, 44, 45, 49, 51, 53)
30 par(mfrow=c(2,2))
31 hist(myX, col = "grey", main = "Default (... is Sturges!)")
32 hist(myX, col = "blue", breaks = "Sturges",
33   main = paste("Using Sturges (K =", nclass.Sturges(myX), ")"))
34 hist(myX, col = "magenta", breaks = "Scott",
35   main = paste("Using Scott (K =", nclass.scott(myX), ")"))
36 hist(myX, col = "green", breaks = "FD",
37   main = paste("Using FD (K =", nclass.FD(myX), ")"))
38 par(mfrow=c(1,1))

```

[12_hist_discretisation.R](#)

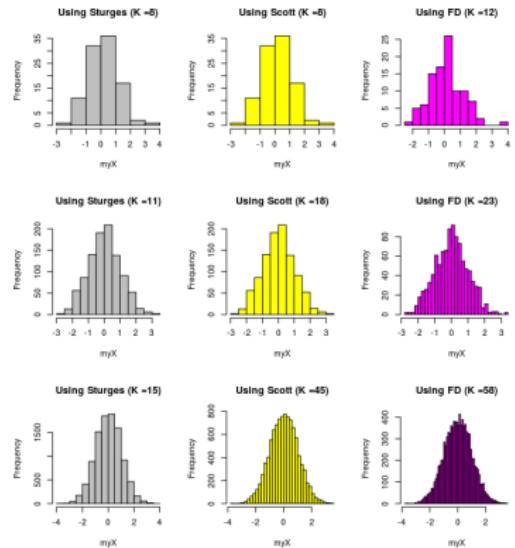


```

40 # Nombre de classes selon le nombre d'observations
41 par(mfrow=c(3,3))
42 for(n in c(100, 1000, 10000)){
43   myX<-rnorm(n)
44   hist(myX, col = "grey",
45         main = paste("Using Sturges (K =", nclass.Sturges(myX), ")", sep = ""))
46   hist(myX, col = "yellow", breaks = "Scott",
47         main = paste("Using Scott (K =", nclass.scott(myX), ")", sep = ""))
48   hist(myX, col = "magenta", breaks = "FD",
49         main = paste("Using FD (K =", nclass.FD(myX), ")", sep = ""))
50 }
51 par(mfrow=c(1,1))

```

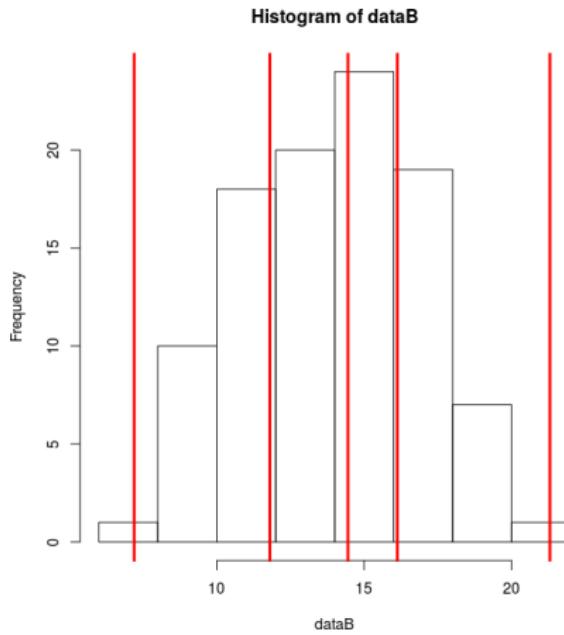
12_hist_discretisation.R



Histogramme et quantile

```
53 # Superposition des quantiles à l'histogramme  
54 hist(dataB)  
55 abline(v = quantile(dataB), col = "red", lwd = 3)
```

12_hist_discretisation.R



Histogramme et courbe de densité

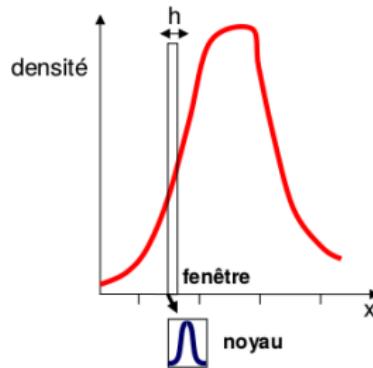
Courbe de densité

Soit x_1, x_2, \dots, x_n n observations d'une variable continue. On peut estimer la densité de cette distribution par la fonction,

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$$

h = largeur de fenêtre, K = densité de probabilité

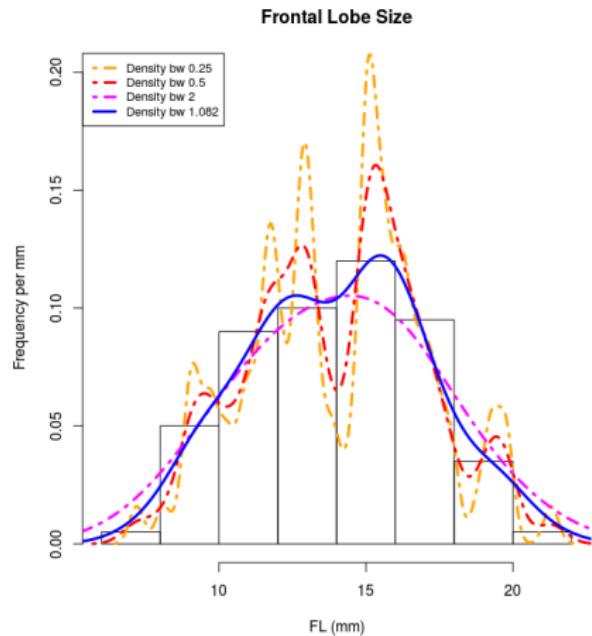
Cette approche consiste à utiliser une fenêtre mobile que l'on déplace sur l'axe des x et à compter le nombre d'occurrences appartenant à cette fenêtre. Cette méthode donne une estimation peu régulière. Si on veut une fonction lisse, il est possible de moyenner via une fonction de densité connue (*noyau*) le long des données observées.



Histogramme et densité

```
57 # Histogramme et courbe de densité
58 hist(dataB, probability = TRUE,
59       ylim = range(0, 0.2), main = "Frontal Lobe Size",
60       xlab = "FL (mm)", ylab = "Frequency per mm")
61
62 lines(density(dataB, bw = 0.25), col = "orange", lwd = 3, lty = 4)
63 lines(density(dataB, bw = 0.5), col = "red", lwd = 3, lty = 4)
64 lines(density(dataB, bw = 2), col = "magenta", lwd = 3, lty = 4)
65 lines(density(dataB), col = "blue", lwd = 3)
66
67 ds = density(dataB)
68
69 # Add a legend
70 legend("topleft",
71        legend = c("Density bw 0.25", "Density bw 0.5",
72                   "Density bw 2", "Density bw 1.082"),
73        col = c("orange", "red", "magenta", "blue"),
74        lty = c(4, 4, 4, 1), lwd = rep(3, 4), cex = 0.8)
```

12_hist_discretisation.R



Variable qualitative

Les observations pour ce type de variable ne sont pas des valeurs numériques, mais des **modalités**.

S'il existe une ordre naturel (e.g. mauvais, moyen, bon), la variable est dit **ordinale**, sinon (e.g. rouge, vert, jaune), la variable est dite **nominale**.

Dans le cas d'une variable ordinale, on peut définir des effectifs et des effectifs cumulés.

Les données

On s'intéresse à la répartition des emplois de la population active en France selon la Catégorie SocioProfessionnelle (CSP), en mars 1988¹. Le tableau ci-dessous résume cette répartition,

CSP	effectifs en milliers	fréquences (%)
agriculteurs exploitants	1312	6,1
artisans, commerçants, chefs d'entreprises	1739	8,1
cadres, professions intellectuelles supérieures	2267	10,6
professions intermédiaires	4327	20,1
employés	5815	27,0
ouvriers	6049	28,1
Total	21509	100,0

Les différentes modalités d'une variable peuvent être représentées par des parties d'un graphique dont la surface est proportionnelle à l'effectif, la fréquence ou le pourcentage. Les graphiques usuels sont le diagramme en **colonne** et le diagramme en **secteurs**.

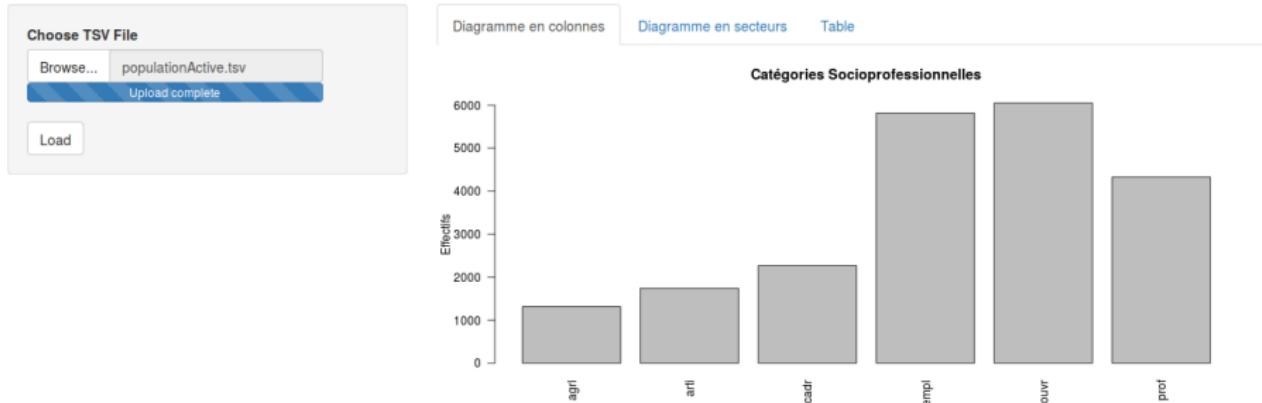
¹ Tableaux de l'Economie Francaise, INSEE, 1989, p. 59

Shiny: tabsetPanel

Le **tabsetPanel** permet de séparer l'interface Shiny en un **sidePanel** et un **mainPanel**. Le mainPanel peut comporter plusieurs onglets ou **tabPanel**.

Nous allons construire un application Shiny qui utilise cette layout afin de charger et afficher via différents graphiques les données d'un variable qualitative.

Variable qualitative



[12_tabsetPanel_varQualitative.R](#)

Template Shiny tabsetPanel

```
library(shiny)

# Contenu de l'interface
ui <- fluidPage(
  titlePanel("Tabssets"),
  sidebarLayout(
    sidebarPanel(
      ..../..
    ),
    mainPanel(
      tabsetPanel(
        tabPanel("Plot", plotOutput("plot")),
        tabPanel("Summary", verbatimTextOutput("summary")),
        tabPanel("Table", tableOutput("table"))
      )
    )
  )
))

# Commandes à executer
server <- function(input, output)
# Association interface & commandes
shinyApp(ui = ui, server = server)
```

Chargement des données

```

36 # Commandes à exécuter
37 server <- function(input, output){
38
39   # Recherche et chargement du fichier de données
40   data <- eventReactive(input$go, {
41     inFile <- input$file1
42     if (is.null(inFile)) return(NULL)
43     read.table(inFile$datapath, header = TRUE, sep = "\t")
44   })
45
46   # Calcul des effectifs
47   effectifs <- reactive({table(data())})

```

data() → entrée reactive

Contient la série de modalités

effectifs → reactive (à data())

Contient les effectifs de data()

Vue de l'application Shiny

Variable qualitative

Choose TSV File

 populationActive.tsv

[Diagramme en colonnes](#)
Diagramme en secteurs
[Table](#)

Diagrammes en colonnes

```

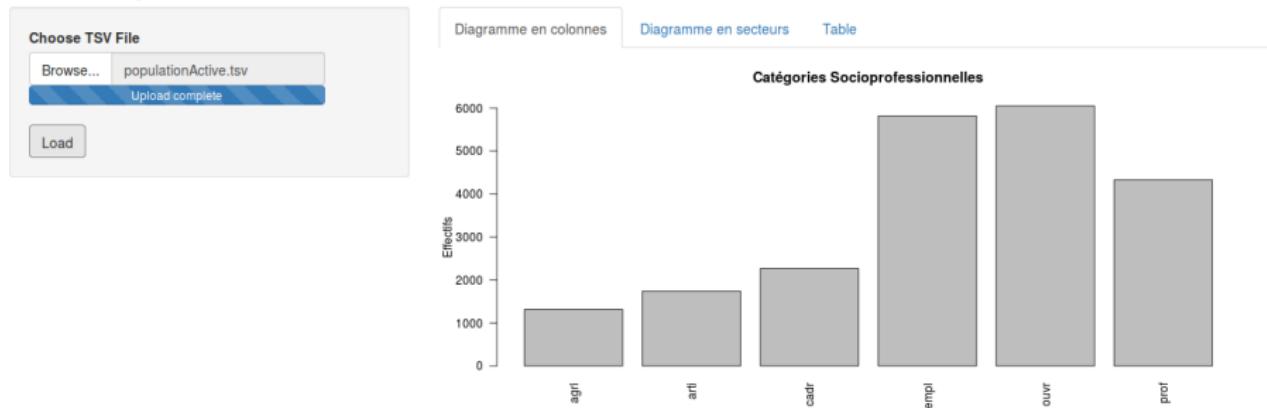
49 # Diagramme en colonnes
50 v output$colonnes <- renderPlot({
51   barplot(effectifs(), main = "Catégories Socioprofessionnelles",
52           ylab="Effectifs", las = 2,
53           names.arg = substr(names(effectifs()), 1, 4))
54
55 })

```

output\$colonnes → sortie reactive
 Affichage en colonnes des effectifs

Vue de l'application Shiny

Variable qualitative



12_tabsetPanel_varQualitative.R

Diagrammes en secteurs

```

57  # Diagramme en secteurs
58  output$secteurs <- renderPlot({
59    pie(effectifs(), labels = substr(names(effectifs()), 1, 4),
60         main = "Catégories Socioprofessionnelles", col=c())
61  })

```

output\$colonnes → sortie reactive

Affichage en colonnes des effectifs

Vue de l'application Shiny

Variable qualitative

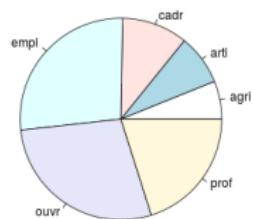
Choose TSV File

Browse... populationActive.tsv
Upload complete

Load

Diagramme en colonnes Diagramme en secteurs Table

Catégories Socioprofessionnelles



12_tabsetPanel_varQualitative.R

Table des effectifs

```
63  # Table des effectifs
64  # ----
65  output$table <- renderTable(effectifs(), colnames = FALSE)
```

output\$table → sortie reactive

Affichage des effectifs

Vue de l'application Shiny

Variable qualitative

Choose TSV File

 populationActive.tsv

	Diagramme en colonnes	Diagramme en secteurs	Table
agriculteurs exploitants		1312	
artisans, commerçants, chefs d'entreprises		1739	
cadres, professions intellectuelles supérieures	2267		
employés		5815	
ouvriers		6049	
professions intermédiaires		4327	

[12_tabsetPanel_varQualitative.R](#)

Bibliography

- Baccini, Alain. *Statistique Descriptive Elementaire*. (2010).
- Mazerolle, Fabrice. *Statistique Descriptive*. (2009).
- Grolemund, Garrett. *Teach Yourself Shiny*.
- *The R Graph Gallery*.
- *R ggplot2 Examples*.
- *Histogrammes avec R*, S. ZIRAH.