Nouvelles technologies sous R

Laurent DELSOL

12 janvier 2026

Introduction

Chapitre 1: A la découverte de RMarkdown

La syntaxe Markdown

Inclusion de codes R

Exemples d'utilisation

Exercice A. Régression linéaire

Exercice B. Entre chien et loup

Exercice C. Randonnée en montagne

Bibliographie

Introduction

Organisation et modaltés d'évaluation

- Pré-requis
  - connaissances de base sur **R**
  - savoir manipuler des fichiers de type word, html, latex...
  - connaissances de base en statistique
- 20 heures de cours-TP
- Evaluation : Examen de 3h en fin de semaine

Plan du cours

- Chapitre 1: A la découverte de RMarkdown
  - La syntaxe Markdown
  - Inclusion de codes **R**
  - Exemples d'utilisation
- Chapitre 2: Documents interactifs avec Shiny
  - Généralités
  - Syntaxe et mise en forme
  - Exemples d'utilisation
- Chapitre 3: Conception de dashboards

Pourquoi utiliser R?

- logiciel libre (gratuit) et open source
- utilisation, modification et implémentation faciles
- dernières méthodes statistiques implémentées
- des outils nombreux et variés dédiés à:
  - la statistique
  - la parallélisation de code
  - le big data
  - le traitement du signal et d'image
- des extensions permettant notamment la création de:
  - rapports automatiques (rmarkdown)
  - applications web interactives (shiny)
  - tableaux de bords (dashboard)

Installation de R

Pour une installation sous Windows, on se rendra sur cette page : http://cran.r-project.org/bin/windows/base/ et l'on suivra le premier lien pour télécharger le programme d'installation. Une fois le programme d'installation lancé, il suffira d'installer R avec les options par défaut.

Pour Mac OS X, les fichiers d'installation sont disponibles à http://cran.r-project.org/bin/macosx/.

Si vous travaillez sous Linux, vous devriez pouvoir trouver **R** via votre gestionnaire de paquets, cela pouvant dépendre d'une distribution de Linux à une autre.

Installation de RStudio et des packages rmarkdown, shiny et dashboard

Une fois R correctement installé, rendez-vous sur http://www.rstudio.com/products/rstudio/download/ pour télécharger la dernière version stable de RStudio. Plus précisément, il s'agit de l'édition Open Source de RStudio Desktop (en effet, il existe aussi une version serveur).

Vous pouvez ensuite utiliser les packages **rmarkdown**, **shiny** et **dashboard** depuis **CRAN** avec les commandes:

```
install.packages("rmarkdown")
install.packages("shiny")
install.packages("dashboard")
```

Chapitre 1: A la découverte de RMarkdown

Pourquoi utiliser RMarkdown?

La syntaxe RMarkdown permet la création de rapports statistiques:

- ayant une mise en page soignée sans que ce soit trop lourd à mettre en place
- pouvant inclure des graphiques, formules mathématiques (LaTeX), commandes R utilisées...
- automatisés et pouvant donc être mis à jour très facilement
- de natures très différentes (html, pdf, word, présentations, tableaux de bord, ...) à partir d'un même code

La syntaxe Markdown

Les bases

Résultat	***Syntaxe***
Texte standard	`Texte standard`
italique	`*italique* et _italique_`
gras	`**gras** et __gras__`
exposant^2 ou indice$_0$	`exposant^2^ ou indice~0~`
~~texte barr'e~~	`~~texte barr\'e~~`
`[lien](www.rstudio.com)`	lien
$A = \pi \times r^2$	`$A = \pi\times r^{2}$`
Note[1]	`Note^[de bas de page]`

[1] de bas de page

Les bases (suite)

On peut combiner ***gras et italique*** et tapant `*__ ... __*`

On obtient un **titre de niveau p** en commençant la ligne par p fois le symbole #. En guise d'exemple, donne

Titre 5

Remarque:

```
Un retour à la ligne n'est pas pris en compte...
sauf si on a laissé deux espaces vides
avant de passer à la ligne suivante.
```

Un retour à la ligne n'est pas pris en compte... sauf si on a laissé deux espaces vides
avant de passer à la ligne suivante.

Blocs, listes à puces, numérotées et imbriquées

Résultat

- pas à...

1. idée 1
2. idée 2
    a. premier point
    b. deuxième point
3. idée 3
    - autre idée
    - autre idée
    - autre idée

Syntaxe

```
>- pas à...
```

```
1. idée 1
1. idée 2
   a. premier point
   a. deuxième point
1. idée 3
   - autre idée
   * autre idée
   + autre idée
```

Blocs, listes à puces, numérotées et imbriquées

Résultat

- pas à...
- pas

1. idée 1
2. idée 2
    a. premier point
    b. deuxième point
3. idée 3
    - autre idée
    - autre idée
    - autre idée

Syntaxe

```
>- pas à...
>- pas

1. idée 1
1. idée 2
   a. premier point
   a. deuxième point
1. idée 3
   - autre idée
   * autre idée
   + autre idée
```

Compléments:

- Pour qu'un titre de section ne soit pas numéroté (avec bookdown par exemple), on ajoute à la suite `{-}` ou `{.unnumbered}`
- L'ajout d'un bloc de citation se fait avec le symbole `>`

```
> "La statistique a démontré que la mortalité dans l'armée augmente sensiblement en temps de guerre." *Alphonse Allais*, **Le Chat noir**, 11 Janvier 1890.
```

> "La statistique a démontré que la mortalité dans l'armée augmente sensiblement en temps de guerre." *Alphonse Allais*, **Le Chat noir**, 11 Janvier 1890.

Figure 1: Exemple de citation obtenue au format HTML

Inclure un tableau

Fonctionne seulement pour certains formats de sortie

```
|  Tables   |      Are      |  Cool  |
|:----------|:-------------:|------:|
| col 1 is  |  left-aligned | $1600 |
| col 2 is  |    centered   |   $12 |
| col 3 is  | right-aligned |    $1 |
```

Tables	Are	Cool
col 1 is	left-aligned	$1600
col 2 is	centered	$12
col 3 is	right-aligned	$1

Figure 2: Exemple de tableau obtenu au format ioslides_presentation

Inclure une image

Image

```
![Burren, Irlande](images/Burren.jpg){width=2cm}
```



Figure 3: Burren, Irlande

Inclure un lien vers un fichier ou un dossier local

- Pour un fichier local

Syntaxe: `[Lien pour la vidéo](videos/VideoB.mp4)`

Résultat: Lien pour la vidéo

- Pour un dossier local

Syntaxe: `[Lien pour le dossier](videos/.)`

Résultat: Lien pour le dossier

Entête

```
---
title: "Nouvelles technologies sous R"
author: "Laurent DELSOL"
date: '12 janvier 2026'
output:
  beamer_presentation:
    colortheme: beaver
    keep_tex: yes
    slide_level: 3
    theme: Montpellier
    toc: yes
    df_print: kable
  runtime:shiny
---
```

Formats de sortie : Documents

- **html_notebook** - Interactive R Notebooks
- **html_document** - HTML document w/ Bootstrap CSS
- **pdf_document** - PDF document (via LaTeX template)
- **word_document** - Microsoft Word document (docx)
- **odt_document** - OpenDocument Text document
- **rtf_document** - Rich Text Format document
- **md_document** - Markdown document (various flavors)

Formats de sortie : Présentations (transparents)

- **ioslides_presentation** - HTML presentation with ioslides
- **revealjs::revealjs_presentation** - HTML presentation with reveal.js
- **slidy_presentation** - HTML presentation with W3C Slidy
- **beamer_presentation** - PDF presentation with LaTeX Beamer
- **powerpoint_presentation** - PowerPoint presentation

Formats de sortie : Autres

- **flexdashboard::flex_dashboard** - Interactive dashboards
- **tufte::tufte_handout** - PDF handouts in the style of Edward Tufte
- **tufte::tufte_html** - HTML handouts in the style of Edward Tufte
- **tufte::tufte_book** - PDF books in the style of Edward Tufte
- **html_vignette** - R package vignette (HTML)
- **github_document** - GitHub Flavored Markdown document

Packages complémentaires: **bookdown**, **blogdown**, **shiny**

Quelques personnalisations et options

- mise en place d'une table des matières:
  toc : yes
- mise à jour automatique de la date:
  ‘`r format(Sys.time(), “%d %B %Y”)`’
- utilisation de paramètres

```
params:
  year: 2018
  region: Europe
  printcode: TRUE
  data: file.csv
```

Choix interactif des paramètres (via le package **shiny**)

```
params:

  year:
    label: "Year"
    value: 2017
    input: slider
    min: 2010
    max: 2018
    step: 1
    sep: ""

  printcode:
    label: "Display Code:"
    value: TRUE
```

Choix interactif des paramètres (via le package **shiny**)

```
params:

  region:
    label: "Region:"
    value: Europe
    input: select
    choices: [North America, Europe, Asia, Africa]

  data:
    label: "Input dataset:"
    value: results.csv
    input: file
```

Inclusion de codes R

Syntaxe générale

Pour inclure un bloc de code R on utilise la syntaxe

```
```{r}
Commandes_R
```
```

Dans un texte, il suffit d'utiliser `r *Commandes_R* `

Un premier exemple

```
```{r sim, prompt=TRUE, comment='Résultat obtenu:'}
a=runif(1,0,10); x=rexp(100,a); a.e=1/mean(x); a.e
```
```

Paramètre de la loi exponentielle: `r a`.

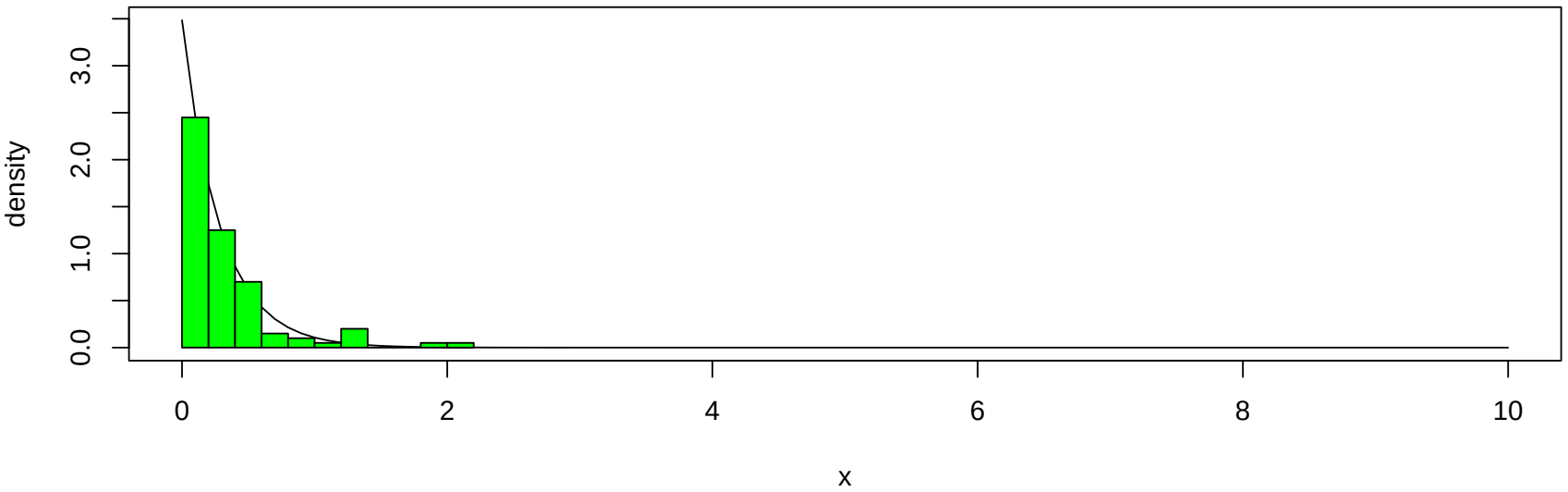
```
```{r hist, fig.height=4, echo=FALSE}
plot(function(x) dexp(x,a),0,10,ylab="density")
hist(x,freq=FALSE,add=TRUE,col='green')
```
```

Résultat

```
> a=runif(1,0,10); x=rexp(100,a); a.e=1/mean(x); a.e
```

```
Résultat obtenu: [1] 3.198303
```

Paramètre de la loi exponentielle: 3.4839973.



Options (quelques exemples)

- **eval** (booléen) indique si le code du bloc doit être évalué
- **echo** (booléen) indique si le code du bloc doit être affiché
- **include** (booléen) indique si le bloc doit être inclus
- **results** ('markup','hide','hold','asis') indique la manière dont les résultats doivent apparaître
- **error** (booléen) indique si les messages d'erreur doivent être inclus
- **message** (booléen) indique si les messages doivent être inclus
- **warning** (booléen) indique si les warnings doivent être inclus
- **cache** (booléen) indique si le bloc doit être mis en cache (et donc exécuté uniquement en cas de modification de son contenu)
- **fig.show** ('asis', 'hide', 'hold','animate') indique la manière dont les figures doivent apparaître

Choix des options par défaut

```
```{r include=FALSE}
knitr::opts_chunk$set(
echo = TRUE,
include=TRUE,
warning=FALSE,
fig.height=3,
fig.show='hold',
results='hold',
comment=' ',
fig.align='center'
)
```
```

Afficher un tableau avec **kable** (package knitr)

```
kable(HairEyeColor_c,'latex',caption="Un exemple de table")
```

Table 1: Un exemple de table

	Brown	Blue	Hazel	Green
Black	68	20	15	5
Brown	119	84	54	29
Red	26	17	14	14
Blond	7	94	10	16

Afficher un jeu de données avec **kable** (package knitr et kableExtra)

```
library(kableExtra)
kable(mtcars,'html',caption="Données mtcars") %>%
  kable_styling() %>%
  scroll_box(width = "100%", height = "10cm")
```

Données mtcars

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

Figure 4: Exemple d'affichage obtenu au format HTML

Exemples d'utilisation

Exemples d'utilisation

Pour terminer, voici quelques illustration de ce qu'il est possible de faire avec RMarkdown:

- Rapport (PDF, WORD, HTML) concernant la simulation et étude d'une série temporelle [Reproductibilité, Adaptabilité] **Rapport**
- Page HTML présentant des simulations (de duels "Fortnite") et leur analyse statistique [Reproductibilité, Interactivité] **Page HTML**
- Présentation sous forme de diaporama (HTML ou PDF) concernant la régression linéaire [Interactivité] **Presentation**
- Article sous (HTML ou PDF) évoquant un tour de magie lié à la suite de Fibonacci [Reproductibilité] **Article**

Exercice A. Régression linéaire

Exercice A. Régression linéaire

Rédigez un rapport en RMarkdown permettant d'effectuer la régression linéaire (nuage de points, corrélation, droite de régression et qualité du modèle) de manière automatisée pour les données générées par les commandes:

```
a=rnorm(1)
b=rnorm(1,2,1)
sigma_b=runif(1)
x=runif(100)
y=a*x+b+rnorm(100,0,sigma_b)
```

Code R 1/3

```r
# Nuage de points
plot(x,y)

# Corrélation
(r=cor(x,y))

# Droite de régression
res=lm(y~x)
summary(res)
a0=res$coefficients[2]
b0=res$coefficients[1]

# Coefficient de détermination
R2=cor(x,y)^2
```

Code R 2/3

```r
# Droite de régression
plot(x,y,main='Droite de régression')
abline(b,a,col='orange')
abline(b0,a0,col='blue')

# Estimation
x0=.7
r=predict(res,data.frame(x=x0))
segments(x0,min(y),x0,r,col='green',lty=2)
segments(min(x),r,x0,r,col='green',lty=2)
```

Code R 3/3

```
# Graphique interactif
library(plotly)
p=plot_ly(x=~x,y=~y, name='data',type = 'scatter',
          mode='markers')
p=add_lines(p,x=~x,y=~a*x+b,name='y=ax+b',
            color=I('orange'))
p=add_lines(p,x=~x,y=~a0*x+b0,name='y=a0+b0',
            color=I('blue'))
p=add_segments(p,x0,min(y),x0,r,name='y=a0x0+b0',
               color=I('green'),linetype = I("dash"))
add_segments(p,min(x),r,x0,r,name='x=x0',
             color=I('green'),linetype = I("dash"))
```

A vous de jouer!

Reproduisez les documents automatisés suivants:

- document word

- document html

- document pdf, uniquement si vous avez latex installé

On pourra utiliser la commande
`outputFormat <- knitr:::pandoc_to()`
pour adapter l'affichage des graphiques au format de sortie.

Exercice B. Entre chien et loup

Exercice B. Entre chien et loup

Des ossements de canidés ont été retrouvés à Jussac. Générer une présentation expliquant en deux ou trois “slides” l’étude des données Jussac.data et le résultat de la classification supervisée par analyse discriminante linéaire.

LAM	LBM	LCB	LM	LMS	LP	Type
13.8	95	129	11.2	64	17.5	Chien
16.5	76	154	14.2	74	20.0	Chien
15.9	71	170	12.3	87	17.9	Chien
14.8	73	188	13.3	94	19.5	Chien
13.0	55	161	12.1	81	17.1	Chien
14.7	58	164	12.7	90	17.5	Chien

Code R 1/2

```r
library(MASS) # charger la librairie MASS pour l'ADL
load('Jussac.RData') # charger les observations

# Analyse discriminante linéaire
Type.lda <- lda(Type~., data=Jussac.learn)
LD1 <- Type.lda$scaling[,1] # Axe 1

# Calcul des projections sur l'axe 1
proj=as.matrix(Jussac.learn[,-7])%*%LD1

# Tracé des projections obtenues
plot(proj,Type=='Chien',xlab='LD1',ylab='Type',
     col=(Type=='Chien')+1,yaxt="n")
axis(2,at=c(0,1),c('Loup','Chien'))
```

Code R 2/2

```r
# Qualité de la classification
CV=lda(Type~., data=Jussac.learn,CV=TRUE)
Classe_est=CV$class
# Matrice de confusion
table(Type,Classe_est)
# Erreur
mean(Classe_est!=Type)
# Courbe ROC
Prob=CV$posterior[,1]
ROC(Type=='Chien',Prob,plot=TRUE)
# Résultats pour Jussac
predict(Type.lda,Jussac)$class # Classe
predict(Type.lda,Jussac)$posterior # Probas a posteriori
```

A vous de jouer!

Reproduisez le document ioslides.

On pourra utiliser les commandes

```
<div class="columns-2">
<div>
<p>...</p>
</div>
<div>
<p>...</p>
</div>
</div>
```

pour adapter l'affichage en deux colonnes.

Exercice C. Randonnée en montagne

Exercice C. Randonnée en montagne

Les données **Randonnée.RData** décrivent le relief d'une zone de montagne. Sur un document HTML, représentez à l'aide d'un graphique en 3D le paysage de la randonnée. Tracer ensuite le profil altimétrique du parcours choisi (on pourra utiliser une estimation à noyau et les coordonnées fournies).

long	lat	alti
99.062468	53.307663	1503.552
-6.870042	28.219049	1565.064
-3.991788	92.239441	1523.288
31.888710	-9.171035	1495.187
20.895597	25.186065	1612.269
15.368273	80.504487	1511.161

Code R 1/3

```
load('Randonnee.RData')
library(plotly)
library(sm)
# Discrétisation
longitude=seq(min(long),max(long),length=100)
latitude=seq(min(lat),max(lat),length=100)
# Estimation
RES=sm.regression(cbind(long,lat),alti,
                  eval.points=cbind(longitude,latitude),
                  eval.grid=TRUE,method='cv',
                  display='none',nbins=0)$estimate
Est_altitude=matrix(RES,100,100)
```

Code R 2/3

```
# Data frame avec les altitudes  estimées sur la grille
DT=data.frame(longitude,latitude,Est_altitude)
# Tracé
plot_ly(DT,x=~latitude,y=~longitude,z = ~Est_altitude,
        type = "surface")

# Estimation d'altitude pour le parcours
Alti_est=sm.regression(cbind(long,lat),alti,
                       eval.points=Position,
                       eval.grid=FALSE,method='cv',
                       display='none',nbins=0)$estimate
```

Code R 3/3

```
# Data frame avec les altitudes  estimées sur le parcours
lo=Position[,1]
la=Position[,2]
DT2=data.frame(Time,lo,la,Alti_est)

# Tracé
plot_ly(DT,x=~latitude,y=~longitude,z = ~Est_altitude,
        type = "surface")%>%
add_trace(x=~la,y=~lo,z = ~Alti_est, type = "scatter3d")
plot_ly(DT2,x=~Time,y=~Alti_est, type = 'scatter',
        mode = 'lines')
```

A vous de jouer!

Reproduisez le document html.

On pourra utiliser les commandes

```
output:
  html_document:
    code_folding: "hide"
```

et

```
# Titre 1 {.tabset .tabset-fade .tabset-pills}
## Onglet 1
## Onglet 2
```

pour cacher/afficher le code et mettre en place les onglets.

Bibliographie

Bibliographie

- Tutorial RMarkdown
- The RMarkdown Cheasheet
- The RMarkdown Reference guide
- R Markdown: The Definitive Guide, Yihui Xie, J. J. Allaire,Garrett Grolemund
- Guide de démarrage en R markdown, Claire Della Vedova