

Presentation of the R-Fiddle, an online free environment to write and run R-code.

1 Introduction

[R-Fiddle](#) is a programming environment for R available online. It allows us to encode and to run a program written in R.

Although R is free and there are also good free programming environments for R (e.g. R-Studio desktop, Tinn-R), this type of tool has several interests. It is suitable for mobile users who frequently change machine. If we have an Internet connection, we can work on a project without having to worry about the R installation on PCs. Collaborative work is another context in which this tool can be particularly advantageous. It allows us to avoid the transfer of files and the management of versions. Last, the solution allows us to work on a lightweight front-end, a laptop for example, and export the calculations on a powerful remote server (in the cloud as we would say today).

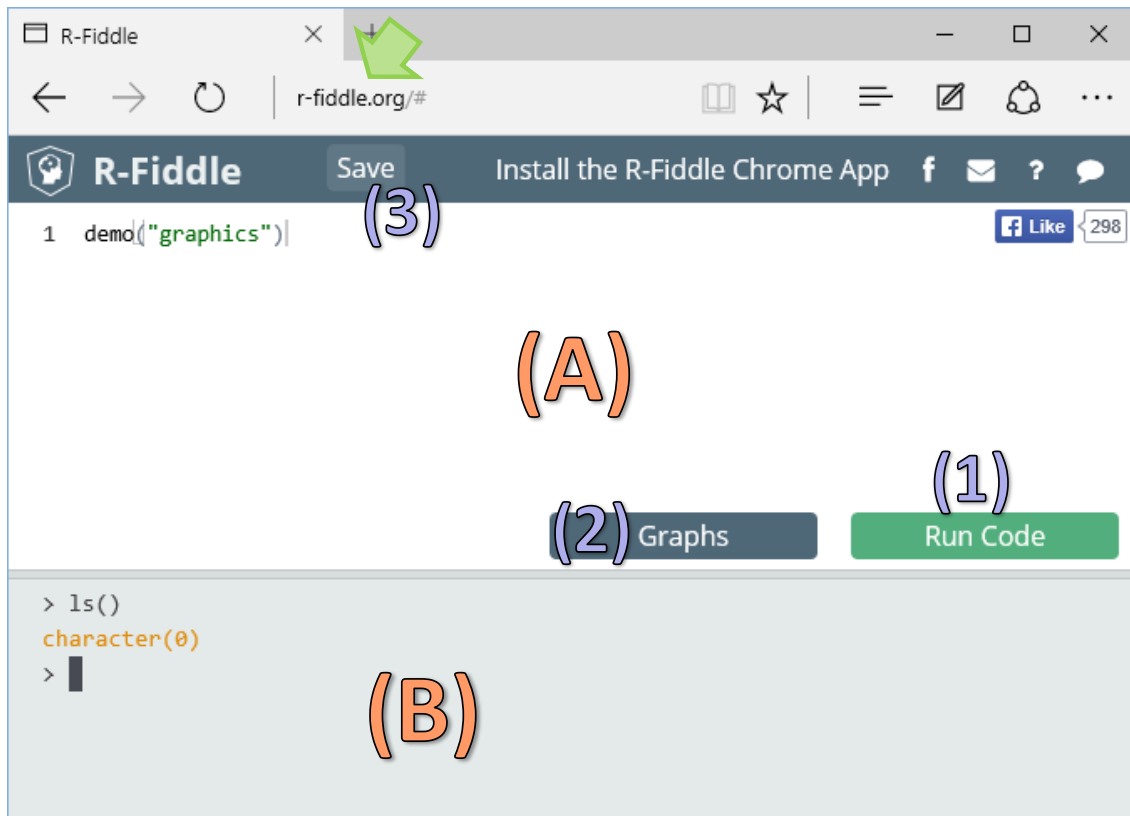
R-Fiddle is rather a prototype (PoC: "[proof of concept](#)"). It allows to run small applications. But it lacks several features to be really operational in a professional environment: it is not possible to have a user account to connect in a private manner; it is not possible to manage a project consisting of several programs; we have no space to store the data; a version management exists, but it is very basic; we cannot manage or modify the calculating capabilities that are at our disposal.

Nevertheless, R-Fiddle corresponds to a concept based on the "cloud" that takes more and more important role. [Azure](#) from Microsoft or other solutions we had analyzed reveal that new opportunities exist. They extend the fields of possibilities in organizational management, in the management of the resources of projects. Beyond the fads, we must identify the contexts of use where they are the most relevant.

In this tutorial, we will briefly review the features of R-Fiddle.

2 Presentation of R-Fiddle

The tool is available at this URL: <http://www.r-fiddle.org/#/>. The graphical user interface is split into two parts: (A) the top allows you to enter our code. (B) the bottom displays the output, we can also execute interactively commands (like for the R console).



We observe several command buttons:

1. RUN CODE launches the execution of the program written into the code editor.
2. GRAPHS displays the different charts generated by our program.
3. SAVE allows to save our code.

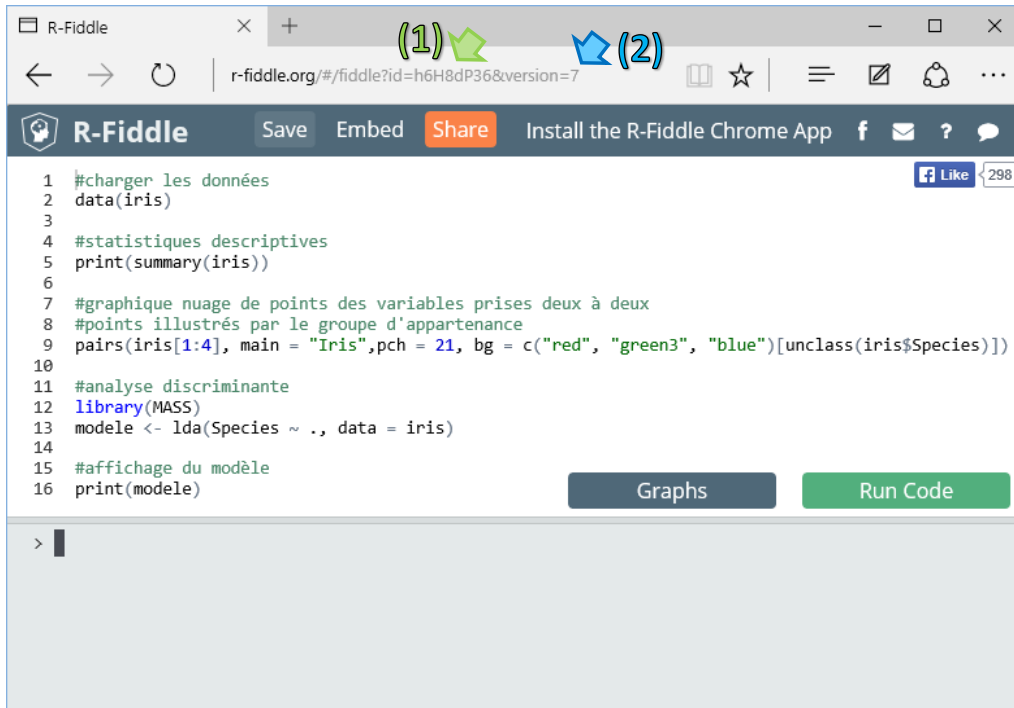
The default command **demo("graphics")** makes a demonstration of the graphics R potential when we launch it with RUN CODE.

3 Working with datasets incorporated into packages

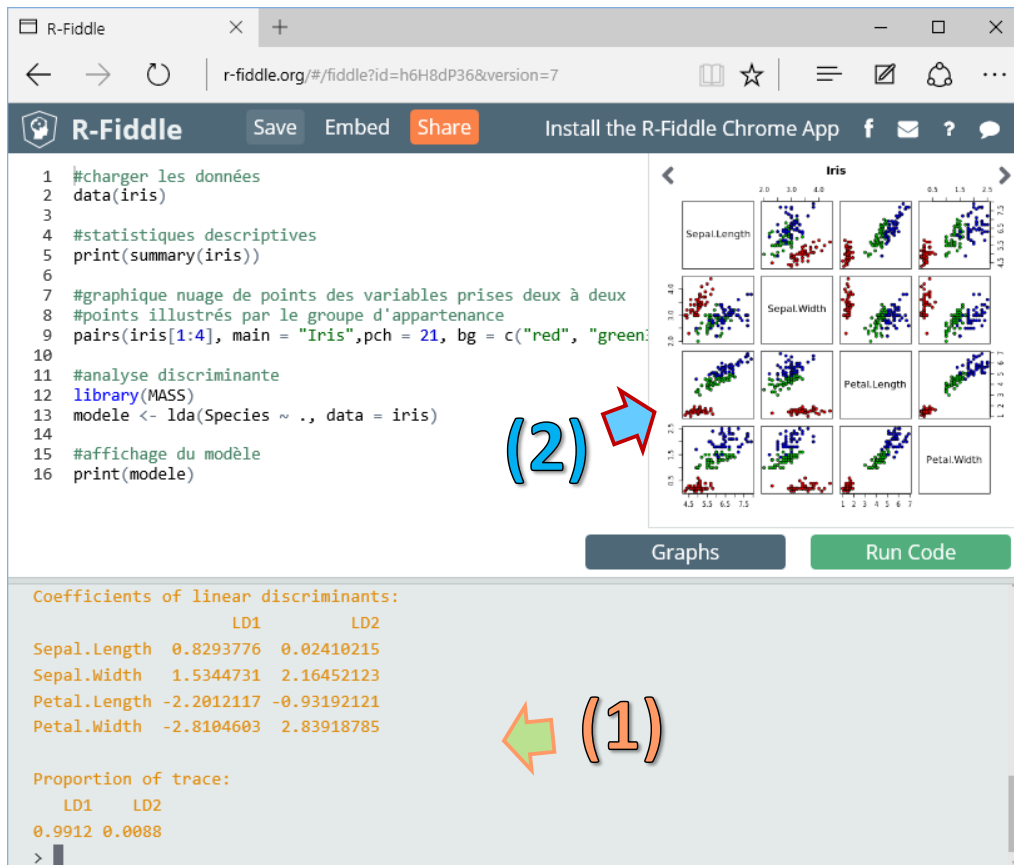
3.1 Using the usual packages

As a first step, we want to work on data included into the "Dataset" package. We can operate without having to tackle the tricky data file loading problem that we will examine in the next section. We write the code below. The outputs include graphics. We click on the SAVE button to save our work. R-Fiddle automatically assigns us (1) an identifier and (2) a version number that increments automatically.

Note: We have no information about the lifetime of our project on the server. We must be prudent if we want to use often the tool!



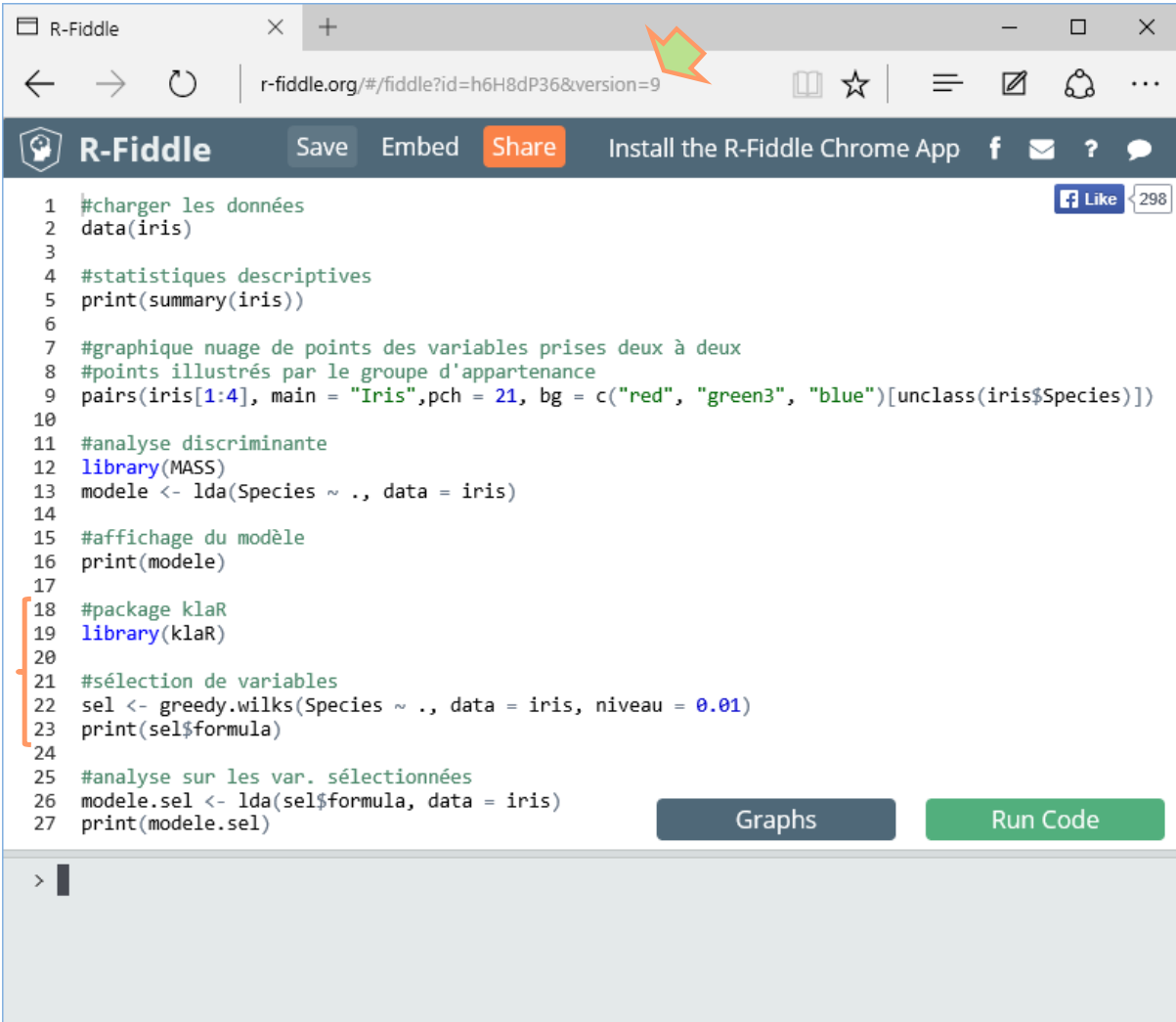
Two new buttons appear: EMBED allows to embed the framework in a web page; SHARE allows to share the address of the project.



We click on the RUN CODE button, numerical results appear in the console (1). The graphical outputs are incorporated in a dedicated window (2) that we can hide or make to appear using the GRAPHS button.

3.2 Using unusual packages

To evaluate the extensibility of the tool, we wished to use packages that are not included in the [standard distribution](#) of R. The [KlaR](#) library offers feature selection tools. The **greedy.wilks()** function is very similar to the [SAS STEPDISC](#) procedure. We modify our code (9th version as we can note in the URL of the project).



The screenshot shows the R-Fiddle web interface. The browser address bar contains the URL `r-fiddle.org/#fiddle?id=h6H8dP36&version=9`. The interface includes a header with the R-Fiddle logo, navigation buttons (Save, Embed, Share), and an option to install the R-Fiddle Chrome App. The main area displays R code for data analysis and feature selection. A red arrow points to the URL in the address bar. The code includes comments in French and uses the `greedy.wilks()` function from the `klaR` package. At the bottom right, there are buttons for 'Graphs' and 'Run Code'. A console window at the bottom shows a prompt `>`.

```
1 #charger les données
2 data(iris)
3
4 #statistiques descriptives
5 print(summary(iris))
6
7 #graphique nuage de points des variables prises deux à deux
8 #points illustrés par le groupe d'appartenance
9 pairs(iris[1:4], main = "Iris",pch = 21, bg = c("red", "green3", "blue")[unclass(iris$Species)])
10
11 #analyse discriminante
12 library(MASS)
13 modele <- lda(Species ~ ., data = iris)
14
15 #affichage du modèle
16 print(modele)
17
18 #package klaR
19 library(klaR)
20
21 #sélection de variables
22 sel <- greedy.wilks(Species ~ ., data = iris, niveau = 0.01)
23 print(sel$formula)
24
25 #analyse sur les var. sélectionnées
26 modele.sel <- lda(sel$formula, data = iris)
27 print(modele.sel)
```

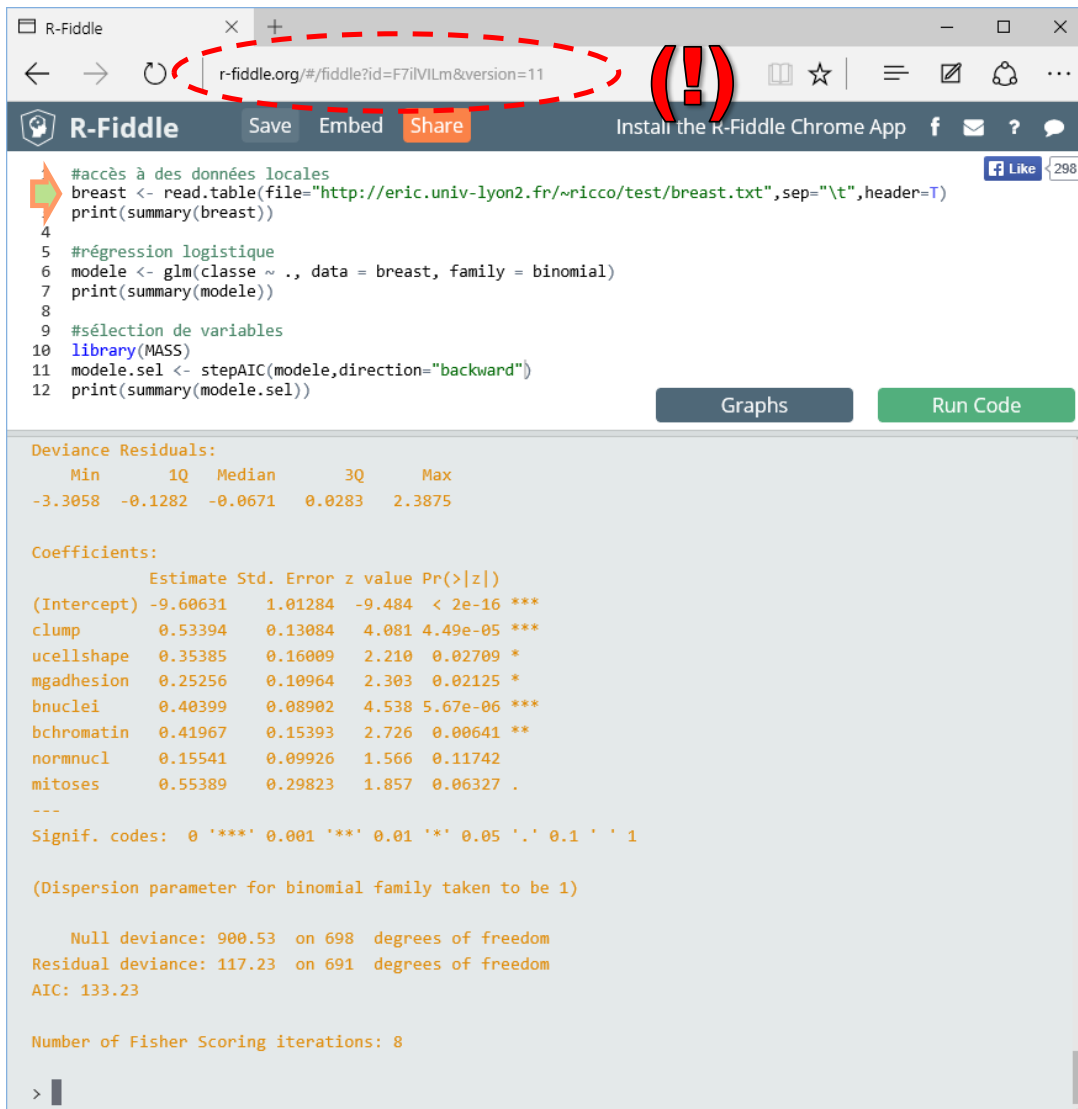
The program worked properly. The SEPAL.LENGTH variable has been removed.

I have not found the list of available packages. It is hoped that the majority of interesting libraries (all depends on what is called “interesting”) are present.

4 Working with our dataset

The matter becomes serious if we work on our own data. I searched how to directly treat a data file on my machine. I have not found. I do not know if I must deplore it or if I must be happy about that. Let an external program directly access our file system is may not be a good idea for the computing safety.

I have therefore download the well-known [BREAST](#) dataset on a server and I have indicated the URL (HTTP protocol) in the R `read_table()` command.



```
#accès à des données locales
breast <- read.table(file="http://eric.univ-lyon2.fr/~ricco/test/breast.txt", sep="\t", header=T)
print(summary(breast))
4
5 #régression logistique
6 modele <- glm(classe ~ ., data = breast, family = binomial)
7 print(summary(modele))
8
9 #sélection de variables
10 library(MASS)
11 modele.sel <- stepAIC(modele,direction="backward")
12 print(summary(modele.sel))
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-3.3058	-0.1282	-0.0671	0.0283	2.3875

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-9.60631	1.01284	-9.484	< 2e-16 ***
clump	0.53394	0.13084	4.081	4.49e-05 ***
ucellshape	0.35385	0.16009	2.210	0.02709 *
mgadhesion	0.25256	0.10964	2.303	0.02125 *
bnuclei	0.40399	0.08902	4.538	5.67e-06 ***
bchromatin	0.41967	0.15393	2.726	0.00641 **
normnucl	0.15541	0.09926	1.566	0.11742
mitoses	0.55389	0.29823	1.857	0.06327 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 900.53 on 698 degrees of freedom
Residual deviance: 117.23 on 691 degrees of freedom
AIC: 133.23

Number of Fisher Scoring iterations: 8

```
> █
```

A new ID was assigned to this new project. This was the 11th version of the backup in the screenshot. Concerning the execution of the program, the data loading and the subsequent logistic regression have functioned properly.

5 Performances

5.1 Handling large dataset

To get an idea about the resources at our disposal, I tried to treat the [COVERTYPE](#) dataset (581012 instances and 54 attributes). The source code below works properly under R on my local machine. I launched it on R-Fiddle.



The screenshot shows the R-Fiddle web interface. The code in the editor is as follows:

```
1 #accès aux données
2 print(system.time(covtype <- read.table(file="http://eric.univ-lyon2.fr/~ricco/test/covtype.txt", sep="\t", header=T)))
3 print(nrow(covtype))
4
5 #arbre de décision
6 library(rpart)
7 print(system.time(arbre <- rpart(TYPE ~ ., data = covtype)))
8 print(arbre)
```

The console output shows the following error messages:

```
Error : cannot allocate vector of size 3.9 Mb
Timing stopped at: 4.862 0.288 6.042
Error : object 'covtype' not found
Error : object 'covtype' not found
Timing stopped at: 0.001 0 0.002
Error : object 'arbre' not found
> |
```

A red arrow points to the first error message: "Error : cannot allocate vector of size 3.9 Mb".

The verdict is immediate. The volume that we can process is limited on R-Fiddle.

5.2 Speed of calculations

I have also analyzed the speed of the remote machine. I have executed the source code below. The purpose is to launch 1000 times the construction of a decision tree on training sets obtained by sampling with replacement on the BREAST dataset.

The computation time is 7.53 sec. on my personal computer (Core i7 - 3.1 Ghz - Windows 10). It was 11.67 sec. on the server of R-Fiddle. The difference is absolutely not prohibitive. The proposed power is quite appropriate given that the tool is freely accessible, and given that there has never been a timeout during all my tests (but I noticed that the system freezes when I tried to launch several times the same program).



The screenshot shows the R-Fiddle web interface. The browser address bar displays `www.r-fiddle.org/#/fiddle?id=4ZOr8uCx&version=11`. The R code in the editor is as follows:

```
1 #accès des données distantes
2 breast <- read.table(file="http://eric.univ-lyon2.fr/~ricco/test/breast.txt",sep="\t",header=T)
3 n <- nrow(breast)
4
5 #arbre de décision
6 library(rpart)
7
8 #compteur
9 tmp <- proc.time()
10
11 #generation d'une liste d'arbre
12 arbres <- list()
13 for (i in 1:1000){
14   #id échantillon
15   obs <- sample(1:n,n,replace=T)
16   #un arbre
17   one_tree <- rpart(classe ~ ., data = breast[obs,])
18   #stockage
19   arbres[[i]] <- one_tree
20 }
21
22 #affichage de contrôle
23 print(length(arbres))
24
25 #calcul durée
26 print(proc.time() - tmp)
```

The output of the code execution is shown in a terminal window:

```
[1] 1000
  user system elapsed
11.650  0.017 11.667
```

A red arrow points to the 'elapsed' column in the output table. Below the terminal window are two buttons: 'Graphs' and 'Run Code'.

6 Conclusion

R Fiddle has the merit to draw our attention to a usage mode of the data mining tool which - in my opinion - will be more and more widely used in the future. The statistical tools and data mining correspond to needs and to specific uses. It is important to understand the advantages of this new operating mode. I note in any case that R users are interested in this issue, and that more or less sophisticated solutions exist (see "[Any Online R console?](#)" on Data Science Stack Exchange).