Construire votre propre package *R*

ANF « *R* avancé et performances »
Aussois - octobre 2015

Aurélie SIBERCHICOT
LBBE – UCBL – Lyon

1. Avant de commencer

- la dernière version de *R*
- un code propre

1. Avant de commencer

- la dernière version de *R*
- un code propre
- un suivi de version
  - *svn* ou *git*
  - R-Forge *http://r-forge.r-project.org/*
  -  *https://github.com/*

1. Avant de commencer

- la dernière version de *R*
- un code propre
- un suivi de version
  - *svn* ou *git*
  - R-Forge *http://r-forge.r-project.org/*
  - *https://github.com/*
- un outil de développement

...

1. Avant de commencer

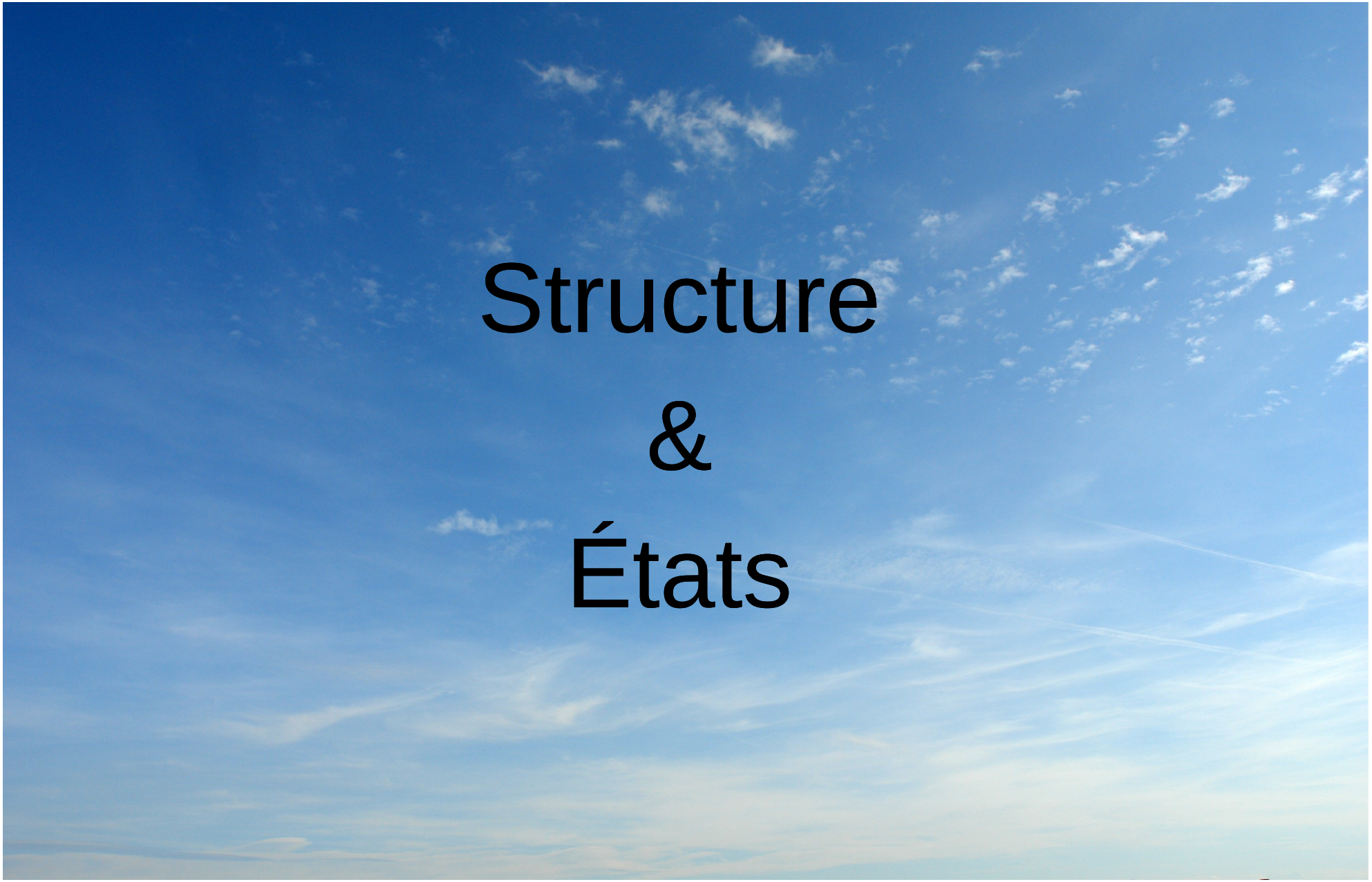
- un développement qui vous voulez diffuser
- un développement qui peut intéresser quelqu'un
- un nom pour votre package

1. Commencez !

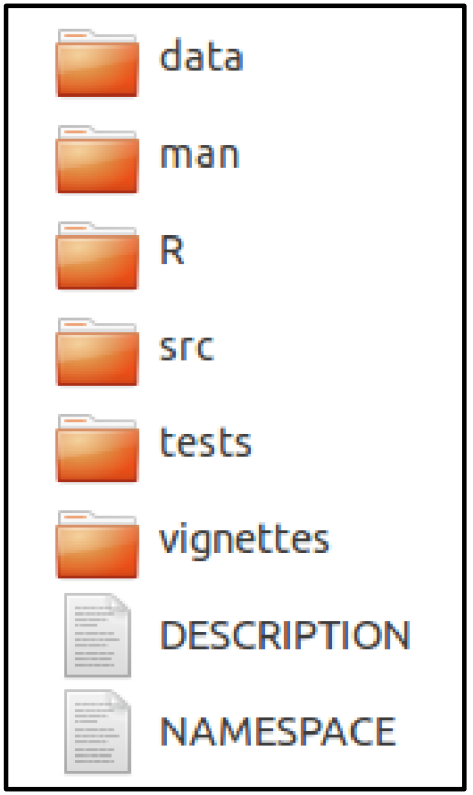
Pour suivre :

- *Writing R Extensions*
- *RStudio*
- packages *devtools, Rcpp* et *roxygen2*

Votre package : *myhclust*


Structure
&
États

2. Une structure normalisée



- architecture minimale obligatoire
- dossiers/fichiers facultatifs selon la nature du package
- contraintes de + en + fortes dans les nouvelles versions de *R*

- *package.skeleton("myhclust")*
- *devtools::create("myhclust")*
- "New project" >> "New directory" >> "R Package"

C'est à vous #1

- créez un package *myhclust* contenant les fichiers *myhclust.R, find_closest.R* et *update.R*

3. Les états d'un package

- 5 états importants

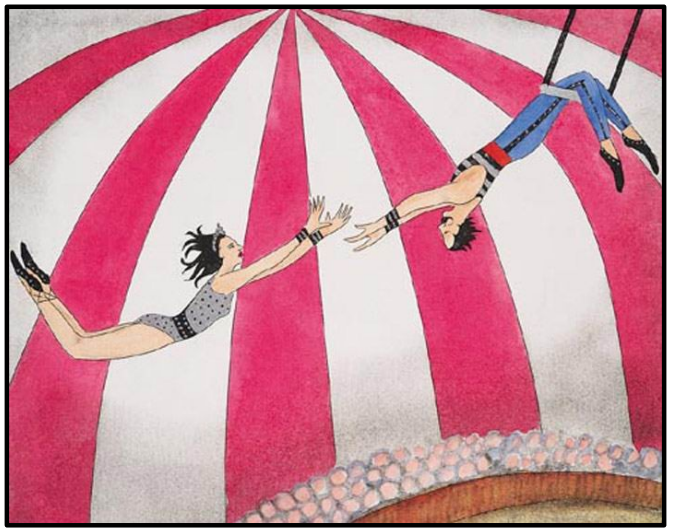
SOURCE

BUNDLE

BINAIRE

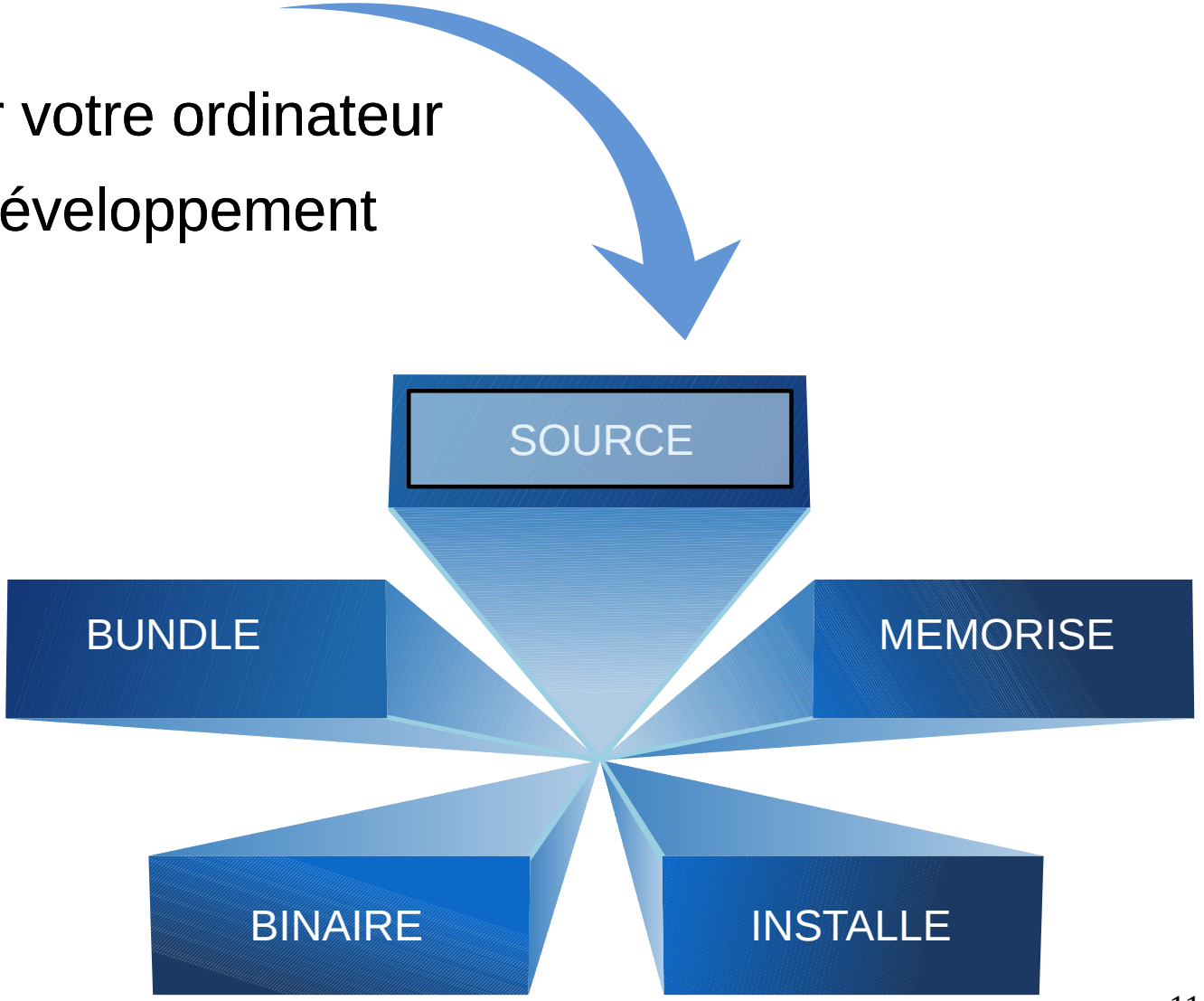
INSTALLE

MEMORISE



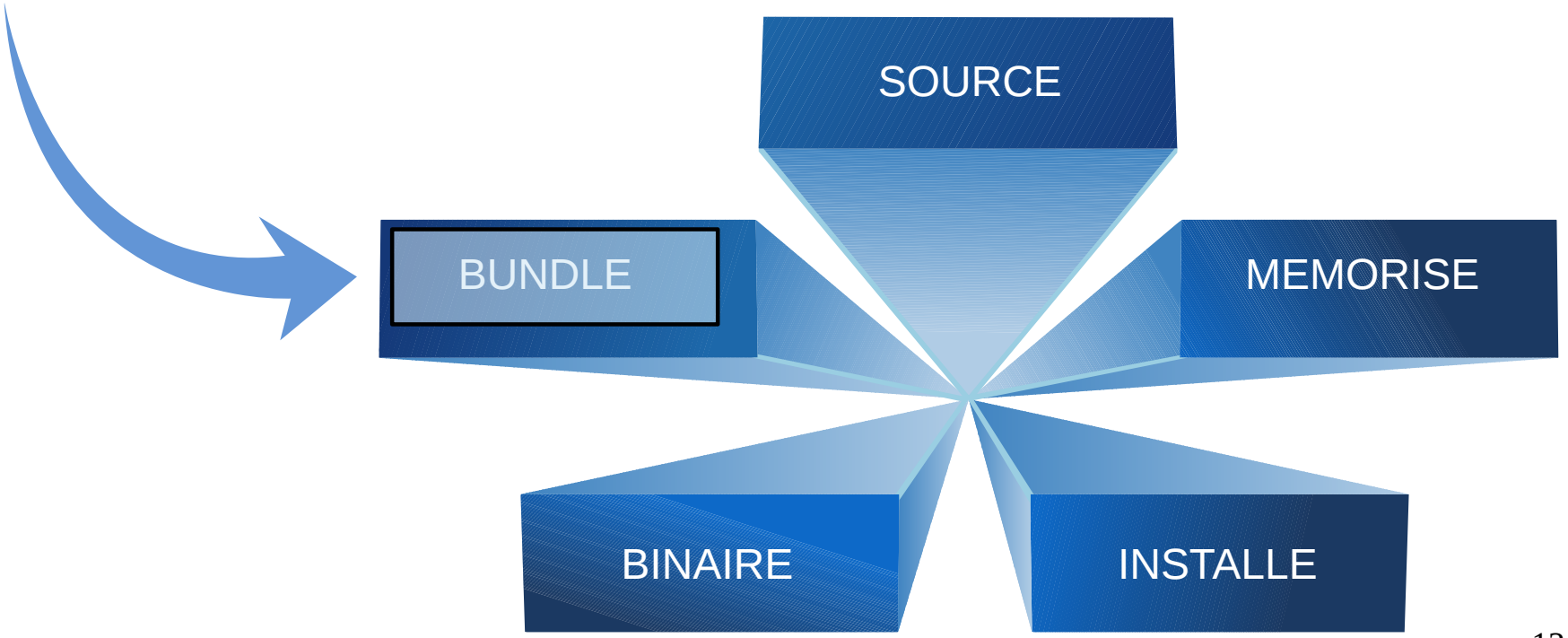
3. Les états d'un package

- Package source
  - celui qui vit sur votre ordinateur
  - la version en développement



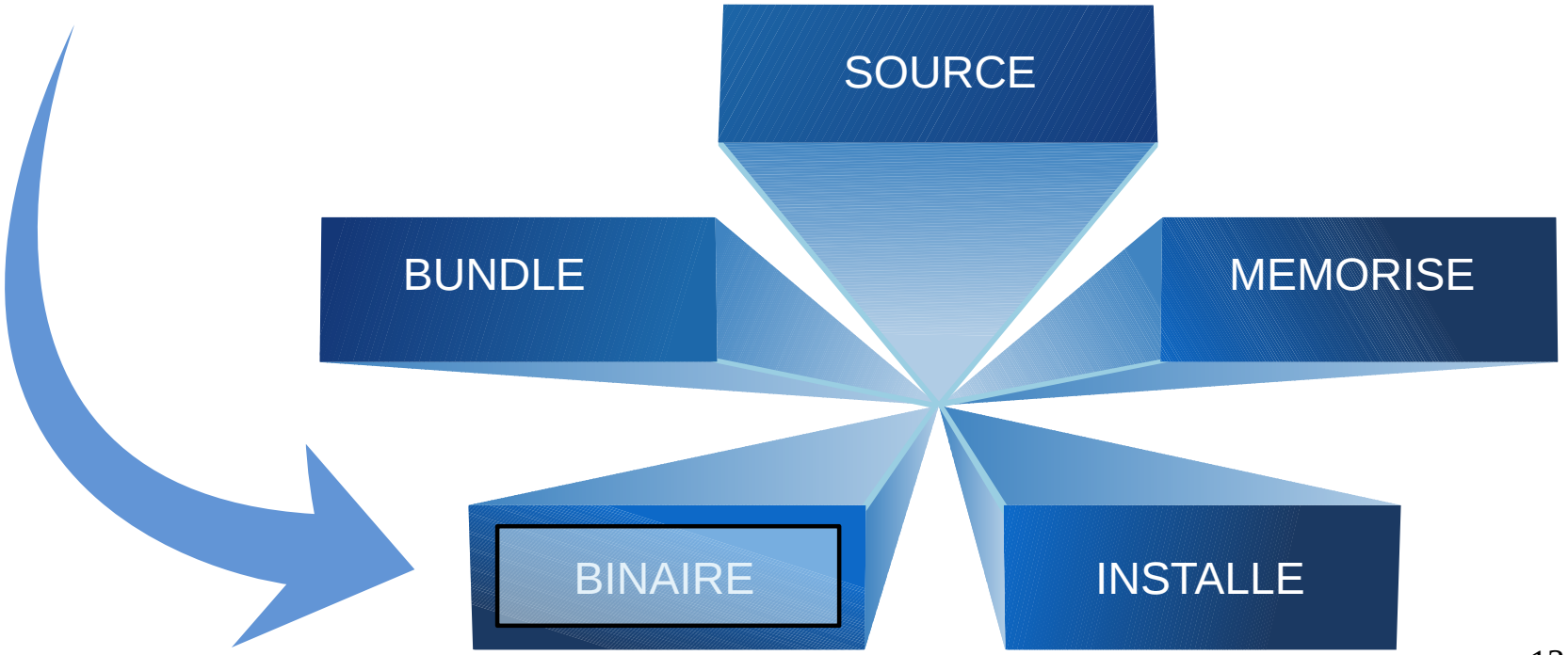
3. Les états d'un package

- Package "bundled"
  - compression en un seul fichier (*.tar.gz*)
  - forme distribuable
  - *R CMD build* ou *devtools::build()*



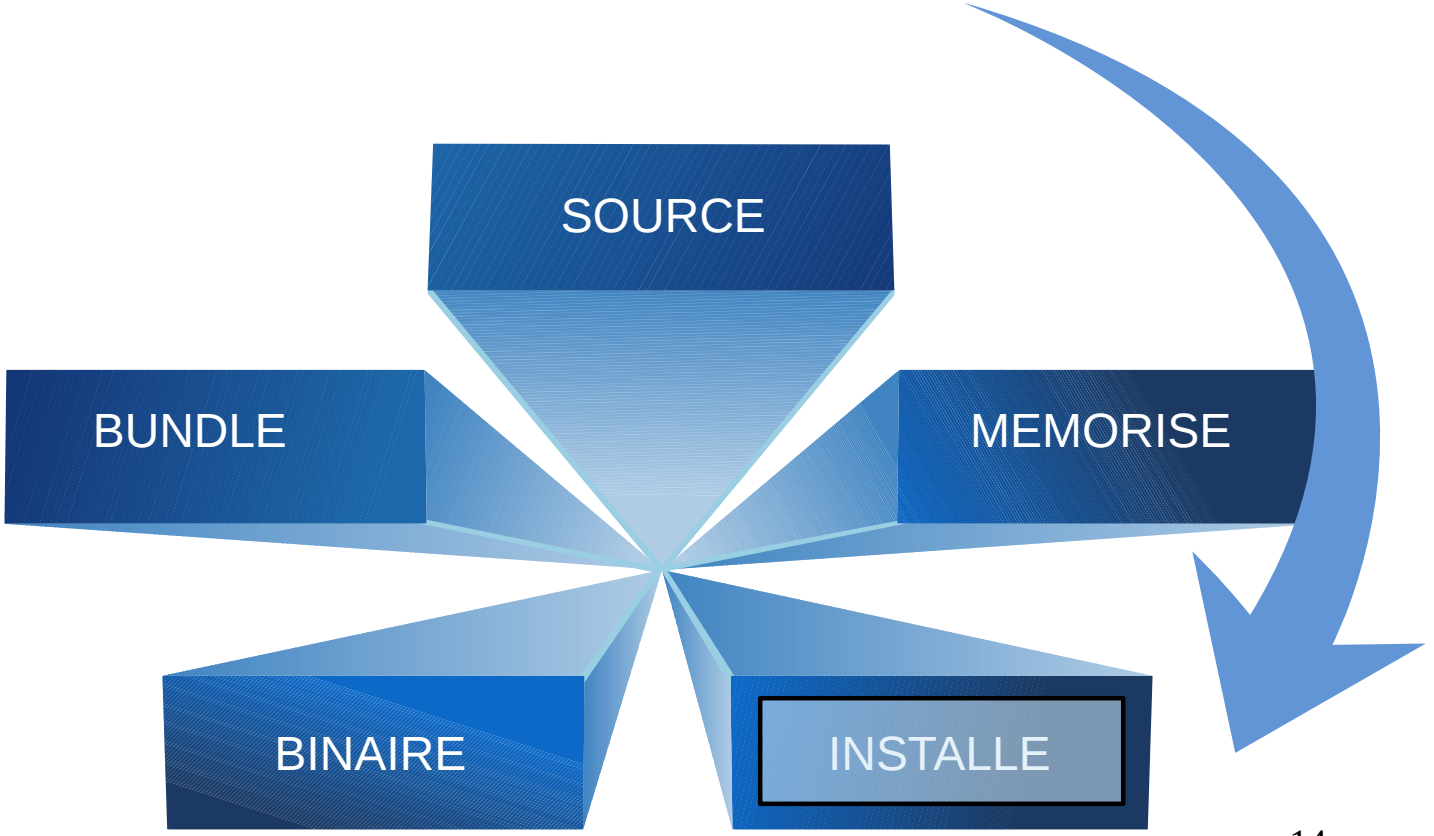
3. Les états d'un package

- Package binaire
  - dépendant de votre système d'exploitation
  - compression en un seul fichier
  - *R CMD INSTALL --build* ou *devtools::build(binary = TRUE)*



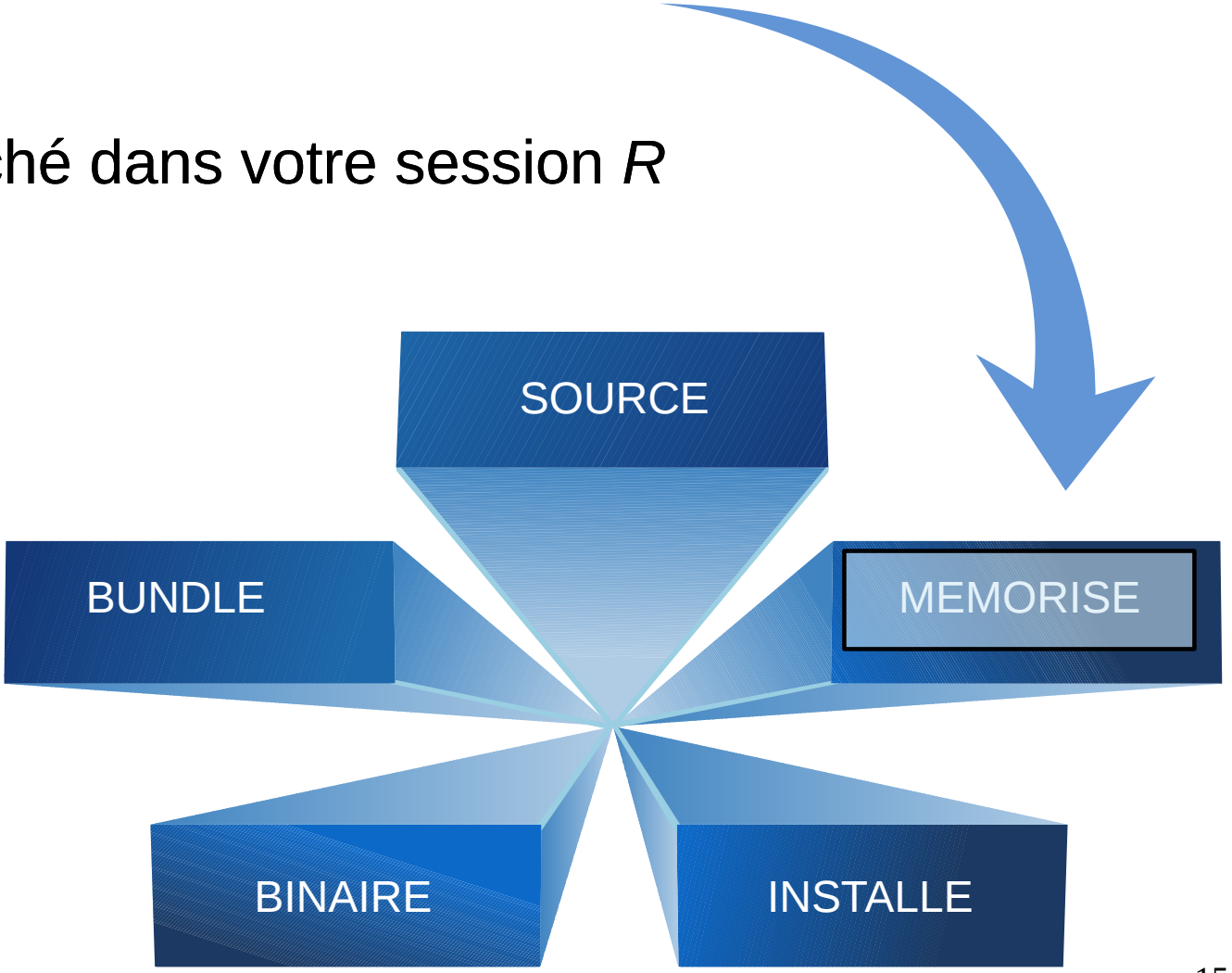
3. Les états d'un package

- Package installé
  - dans votre bibliothèque locale
  - *R CMD INSTALL* ou *devtools::install()*



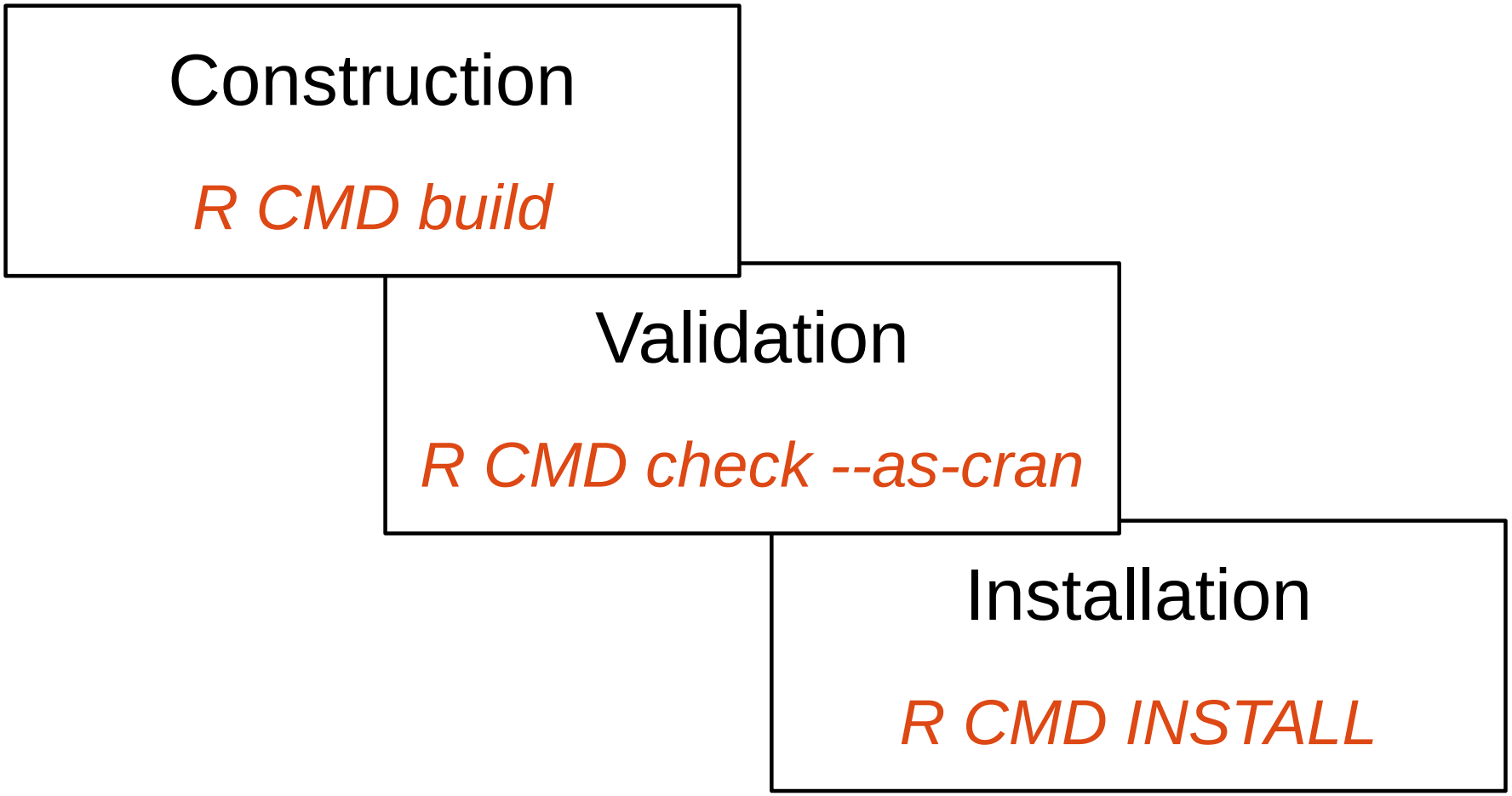
3. Les états d'un package

- Package mémorisé
  - *library()*
  - chargé et attaché dans votre session *R*



4. Les étapes d'un processus

En pratique, 3 étapes à exécuter :



4. Les étapes d'un processus

Construction

R CMD build

- construit la forme distribuable de votre package (*.tar.gz)* depuis le package source

- diagnostic et vérifications généraux
- étape intermédiaire avant la validation et l'installation

4. Les étapes d'un processus

Validation

R CMD check --as-cran

- vérifie le contenu et la structure de votre package
- option *--as-cran* pour contrôler plus de points

- crée un dossier *pkgname.Rcheck*
- notions de 'error', 'warning', 'note'
- un package avec des erreurs sera rejeté par CRAN

4. Les étapes d'un processus

Installation

R CMD INSTALL

- installation depuis le source, le bundle ou le binaire
- dans une bibliothèque de packages

C'est à vous #2

- créez la version distribuable de votre package
- vérifiez-la et notez le résultat de la validation
- installez votre package

C'est à vous #2

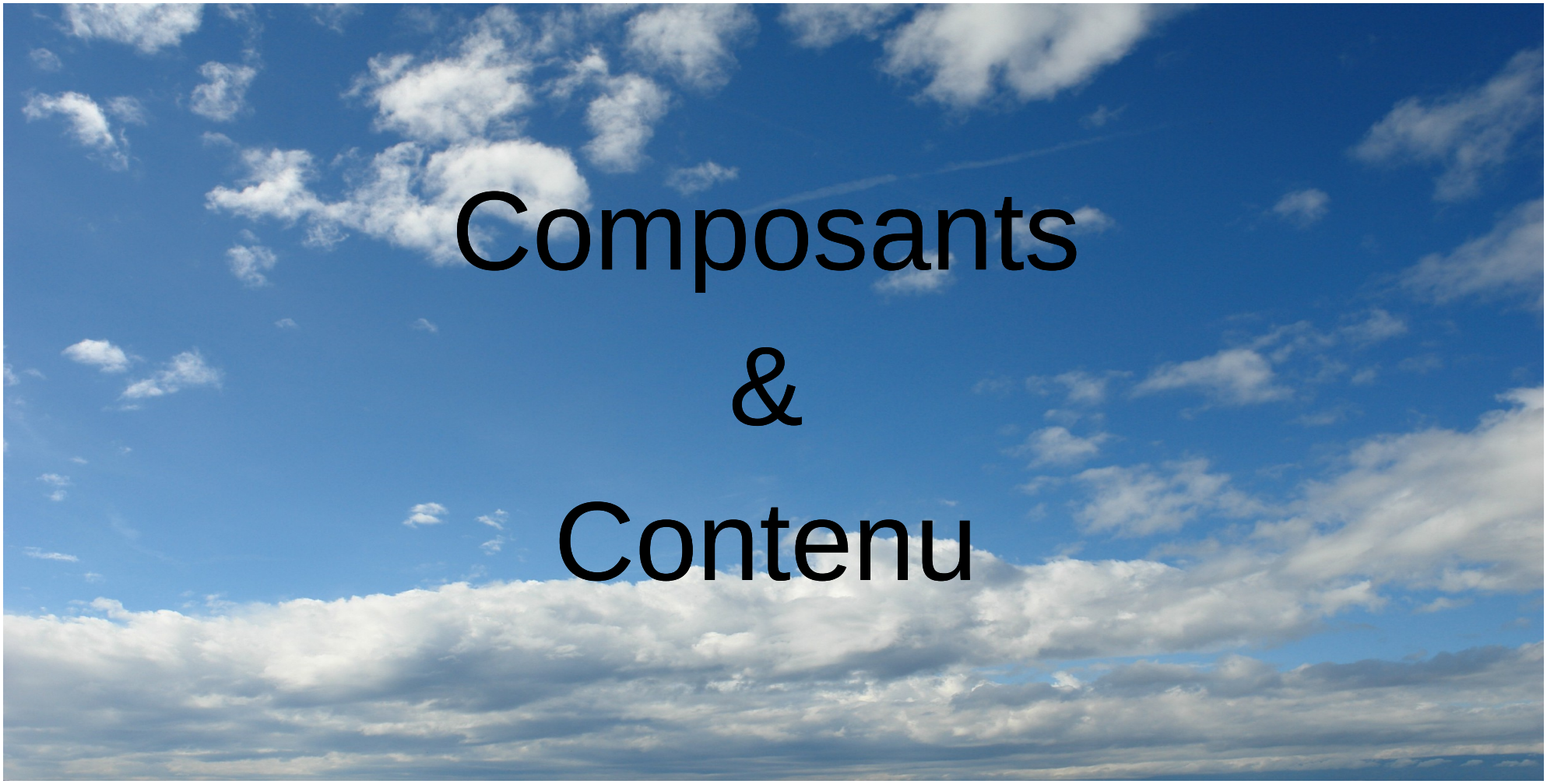
- créez la version distribuable de votre package

  → *build* votre package bundled *.tar.gz*
- vérifiez-la et notez le résultat de la validation

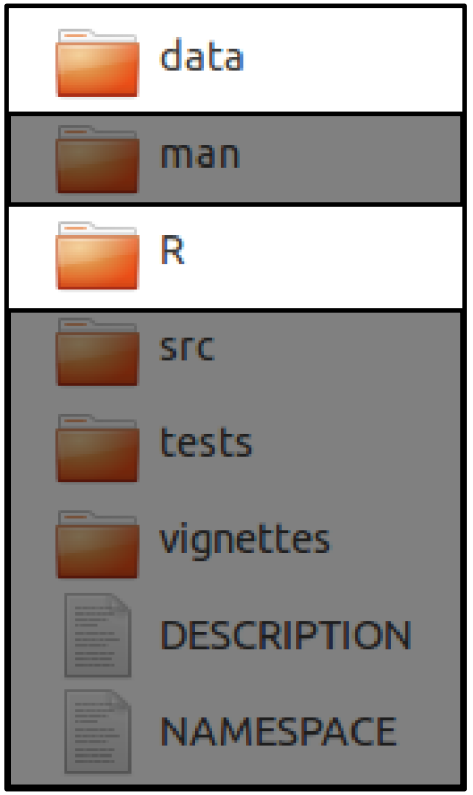
  → *check*

  → *ERROR*, *WARNING*, *NOTE*
- installez votre package

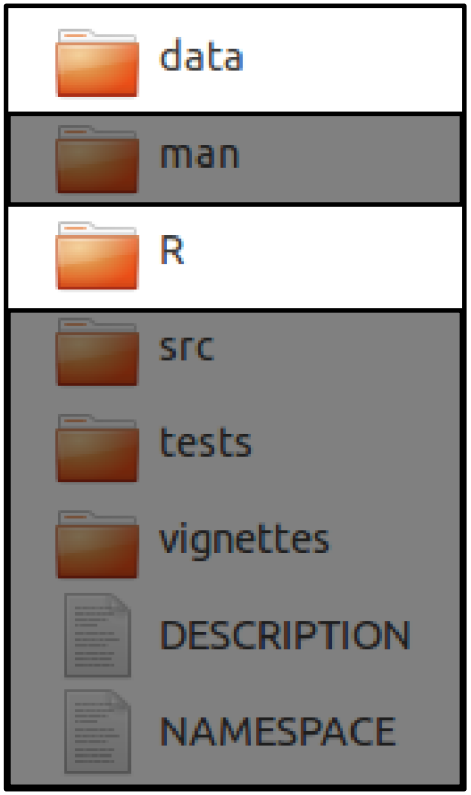
  → *INSTALL*


Composants
&
Contenu

5. Le « cœur » du package



5. Le « cœur » du package

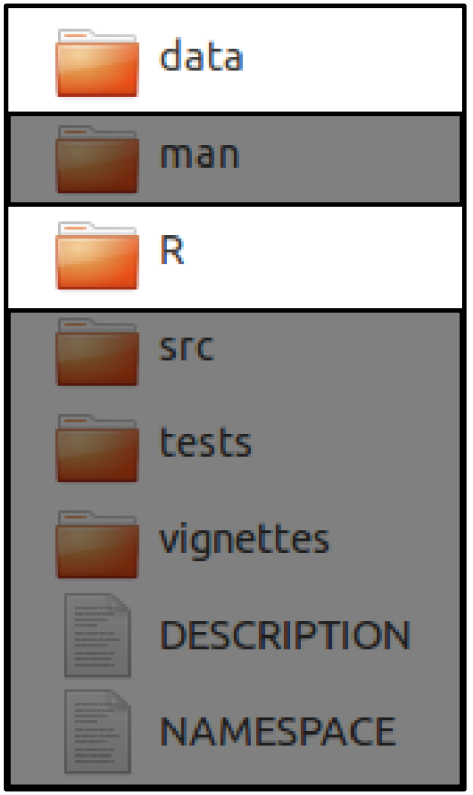


R/

OBLIGATOIRE

- scripts ***.R***
- organisation libre des fonctions et des fichiers
- convention de codage

5. Le « cœur » du package



data/

- fichiers ***.RData*** pour un format optimal
- *save()* pour créer le fichier *.RData*
- < 1Mb

R/

OBLIGATOIRE

- scripts ***.R***
- organisation libre des fonctions et des fichiers
- convention de codage

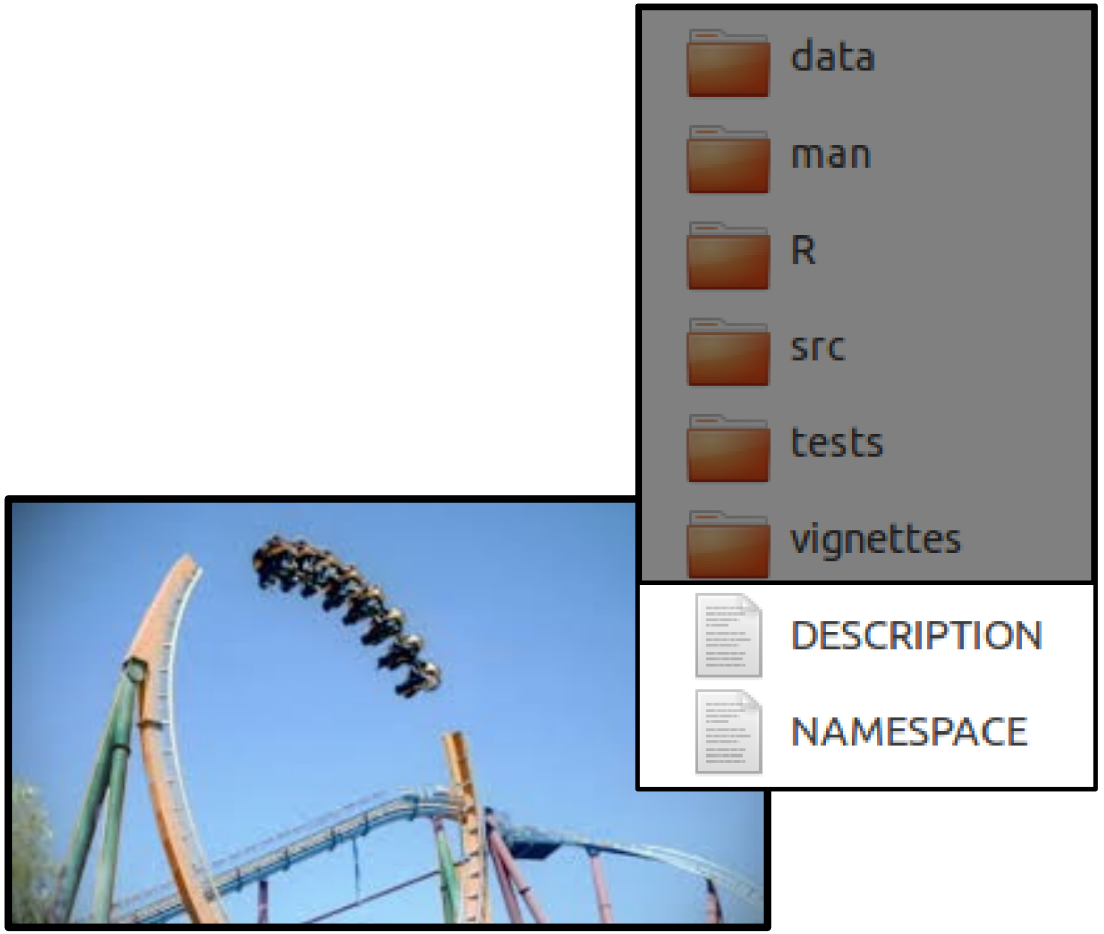
C'est à vous #3

- ajoutez *distaussois.RData* à votre package

```
N <- 100
x <- runif(N / 4)
y <- runif(N / 4)
mypoints <- cbind(c(x, x + 1.5, x, x + 1.5), c(y, y + 1.5, y + 1.5, y))
distaussois <- as.matrix(dist(mypoints))
```

- ajoutez à votre package le fichier *plot.R* contenant la fonction *myplotdist*
- identifiez le package dont vous avez besoin pour que la fonction *myplotdist* fonctionne

6. Les fichiers « orchestres »



6. Les fichiers « orchestres »

DESCRIPTION

OBLIGATOIRE

- définit votre package
  - de quoi votre package a besoin
  - qui peut utiliser votre package
  - qui doit être contacté en cas de problème
- format contraint : rubriques obligatoires
- mais structure simple : une ligne par élément

6. Les fichiers « orchestres »

7 champs OBLIGATOIRES

Package	le nom du package
Version	le numéro de version du package : *<major>.<minor>.<patch>*
Licence	gestion des droits d'utilisation du code par le public
Title	une courte description du package
Description	un paragraphe pour mieux comprendre ce que fait le package, plus détaillé que le titre
Author	la liste des personnes qui ont contribué au package
Maintainer	le nom et l'adresse mail (valide et pérenne) d'une seule personne, désignée comme mainteneur et correspondant

6. Les fichiers « orchestres »

7 champs OBLIGATOIRES

Package	le nom du package
Version	le numéro de version du package : *<major>.<minor>.<patch>*
Licence	gestion des droits d'utilisation du code par le public
Title	une courte description du package
Description	un paragraphe pour mieux comprendre ce que fait le package, plus détaillé que le titre
A	
Authors@R	code R exécutable permettant d'attribuer des rôles à un ensemble de personnes ("*cre*", "*aut*", "*ctb*", …) permet de gérer plus d'informations (mail, rôle, ...)
M	ne,

6. Les fichiers « orchestres »

7 champs OBLIGATOIRES

Package	le nom du package
Version	le numéro de version du package : <*major*>.<*minor*>.<*patch*>
Licence	gestion des droits d'utilisation du code par le public
Title	une courte description du package
Description	un paragraphe pour mieux comprendre ce que fait le package, plus détaillé que le titre
Authors@R	code R exécutable permettant d'attribuer des rôles à un ensemble de personnes ("*cre*", "*aut*", "*ctb*", …) permet de gérer plus d'informations (mail, rôle, ...)

d'autres rubriques facultatives : *Collate*, *LazyData*, *Date*, ...

6. Les fichiers « orchestres »

Depends	- packages obligatoirement présents - version minimale de *R* - attachés dans l'environnement global
Imports	- packages obligatoirement présents - chargés dans l'environnement de votre package
Suggests	- packages nécessaires uniquement pour les tests, les exemples, la vignette

- un même package ne peut pas être dans plusieurs rubriques
- possibilité de spécifier la version minimale requise du package

R (>= 3.0.1), *sp (>= 1.1.1)*, *ade4 (>= 1.4-3)*

6. Les fichiers « orchestres »

Depends	- packages obligatoirement présents - version minimale de *R* - attachés dans l'environnement global
Imports	- packages obligatoirement présents - chargés dans l'environnement de votre package
Suggests	- packages nécessaires uniquement pour les tests, les exemples, la vignette

- un même package ne peut pas être dans plusieurs rubriques
- possibilité de spécifier la version minimale requise du package

R (>= 3.0.1), *sp (>= 1.1.1)*, *ade4 (>= 1.4-3)*

C'est à vous #4

- dans DESCRIPTION, éditez l'auteur et le mainteneur, le titre, la description et la licence
- dans DESCRIPTION, signalez que votre package dépend du package *lattice*
- vérifiez votre package

6. Les fichiers « orchestres »

NAMESPACE

OBLIGATOIRE

Pour que votre package fonctionne, il faut spécifier

- les objets de votre package qui doivent être disponibles à l'extérieur (***export***)

- les objets dont vous avez besoin qui sont issus d'autres packages (***import***)

6. Les fichiers « orchestres »

NAMESPACE

OBLIGATOIRE

Pour **exporter** des objets de votre package, vous pouvez faire :

export(function1, function2)

exportPattern("^[[:alpha:]]+")

6. Les fichiers « orchestres »

NAMESPACE

OBLIGATOIRE

Pour **exporter** des objets de votre package, vous **devez** faire :

export(function1, function2)

exportPattern("^[^.]")

- en programmation objet : *S3Method()*, *exportMethods()*, *exportClasses()*

S3 *S4*

6. Les fichiers « orchestres »

NAMESPACE

OBLIGATOIRE

Pour **importer** des objets issus d'autres packages, vous pouvez faire :

import(pack1, pack2)

importFrom(pack1, v1, v2)

pack1::v1

pack1:::v1

} dans le code *R*

6. Les fichiers « orchestres »

NAMESPACE

OBLIGATOIRE

Pour **importer** des objets issus d'autres packages, vous **devez** faire :

import(pack1, pack2)

importFrom(pack1, v1, v2)

pack1::v1

pack1:::v1

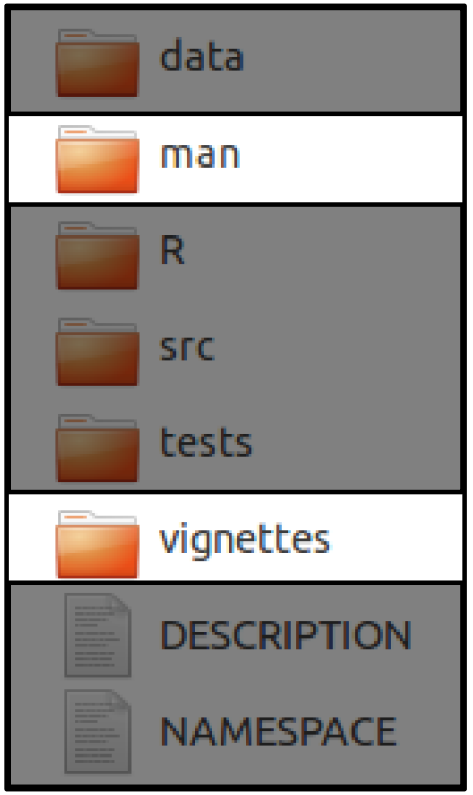
S4

- en programmation objet : import*MethodsFrom()*, im*portClassesFrom()*
- pour importer du C ou du C++ : *useDynLib()*

C'est à vous #5

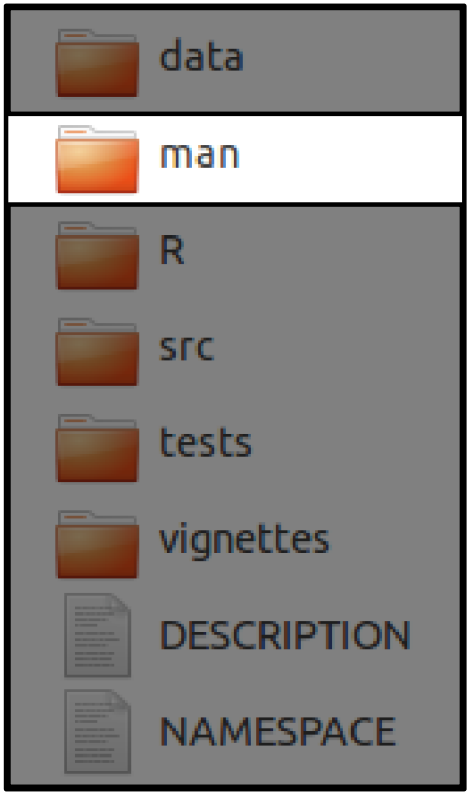
- dans NAMESPACE, exportez les fonctions *my.hclust* et *myplotdist*
- dans NAMESPACE, importez la fonction *histogram* du package *lattice*
- vérifiez votre package

7. La documentation



7. La documentation

OBLIGATOIRE



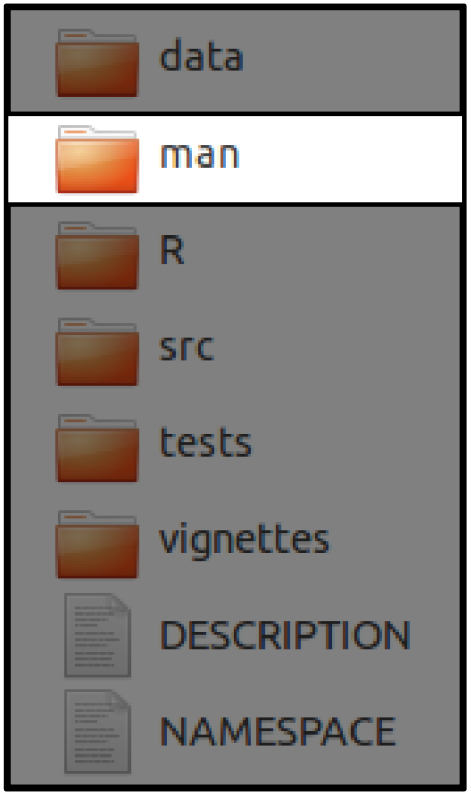
man/

- fichiers ***.Rd***
- syntaxe spécifique : mélange *LaTeX*, texte verbatim et *R*
- convertis en *HTML*, en texte brut ou en *pdf* pour les visualiser

- obligation de documenter tous les objets exportés
- format et rubriques standardisés

7. La documentation

OBLIGATOIRE



man/

Création facilitée ou automatique de la documentation :

- commande *prompt (+ promptData, promptPackage, promptClass, promptMethods)*
- packages *roxygen2* et *inlinedocs* pour une lecture dynamique des commentaires dans le code

7. La documentation

OBLIGATOIRE

\name{}	nom du fichier *Rd*. doit être unique dans votre dossier *man/*
\alias{}	nom des objets qui sont documentés dans le fichier *Rd*
\title{}	titre du fichier *Rd*
\description{}	quelques lignes pour décrire ce que l' (les) objet(s) fait (font)
\usage{}	précision sur comment est utilisée la fonction
\arguments{}	description de chacun des arguments de la fonction
\value{}	description de tous les éléments renvoyés par la fonction
\examples{}	code exécutable *R* fournissant des exemples d'application de l'objet documenté

7. La documentation

OBLIGATOIRE

\name{}	nom du fichier *Rd*. doit être unique dans votre dossier *man/*
\alias{}	nom des objets qui sont documentés dans le fichier *Rd*
\title{}	titre du fichier *Rd*
\description{}	quelques lignes pour décrire ce que l' (les) objet(s) fait (font)
\usage{}	précision sur comment est utilisée la fonction
\arguments{}	description de chacun des arguments de la fonction
\value{}	description de tous les éléments renvoyés par la fonction
\examples{}	code exécutable *R* fournissant des exemples d'application de l'objet documenté

d'autres rubriques : *details*, *references*, *author*, *note*, *seealso*, *keyword*, ...

7. La documentation

OBLIGATOIRE

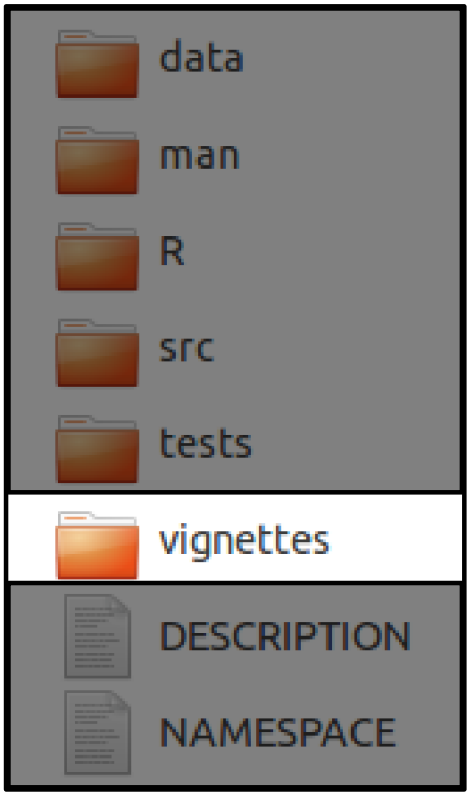
\name{}	nom du fichier *Rd*. doit être unique dans votre dossier *man/*
\alias{}	nom des objets qui sont documentés dans le fichier *Rd*
\title{}	titre du fichier *Rd*
\description{}	quelques lignes pour décrire ce que l' (les) objet(s) fait (font)
\usage{}	précision sur comment est utilisée la fonction
\arguments{}	description de chacun des arguments de la fonction
\value{}	description de tous les éléments renvoyés par la fonction
\examples{}	code exécutable *R* fournissant des exemples d'application de l'objet documenté

d'autres rubriques : *details*, *references*, *author*, *note*, *seealso*, *keyword*, ...

C'est à vous #6

- documentez la fonction *my.hclust* et vos données *distaussois* à l'aide des commandes *prompt*
- documentez les fonctions *myplotdist* avec les commandes de *roxygen2*
- vérifiez votre package

7. La documentation



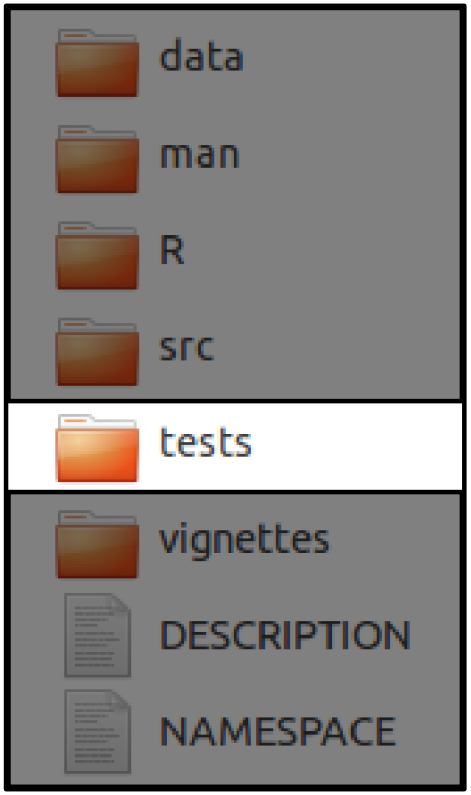
vignettes/

- guide d'utilisation complet
- contient du code *R* exécuté

- format Sweave (*.rnw*) par défaut
- possibilité d'utiliser Markdown (*.Rmd)*
- balise *%VignetteIndexEntry* nécessaire

- publier votre vignette dans le *R Journal*

8. Les fichiers de tests



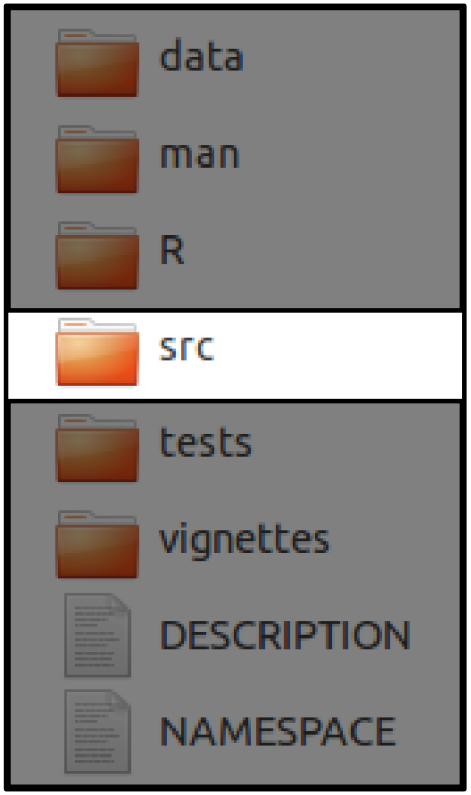
tests/

- assertions binaires
- un bug => un test

- scripts *R* exécutés au *R CMD check*
- résultats enregistrés dans *.Rout*
- < 1 minute

- package *testthat*
- bouton "Test Package" dans RStudio

9. Les sources

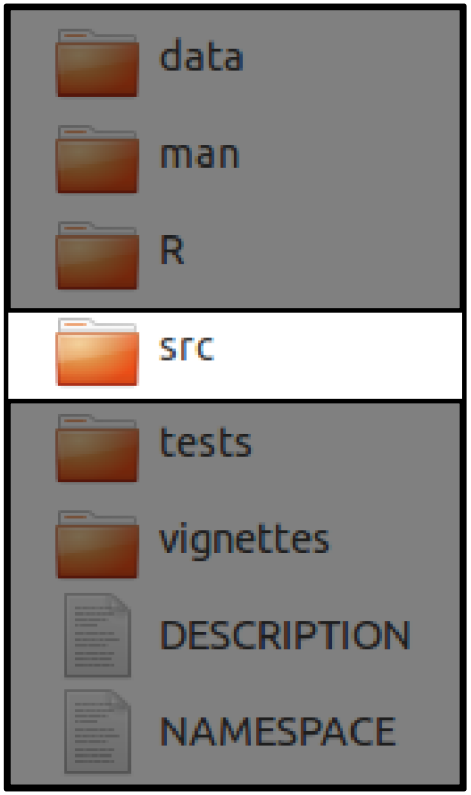


src/

- peut contenir du C, du C++, du Java ou du Fortran

- pour intégrer du C++, package *Rcpp*
- dans DESCRIPTION, *Rcpp* dans '*Imports*' et '*LinkingTo*'
- dans NAMESPACE, *useDynLib(myhclust)* et *importFrom(Rcpp, sourceCpp)*

9. Les sources



src/

Export de fonctions C++

- exporter la fonction "wrapper" R (du même nom) associée à la fonction C++
- documenter la fonction R exportée

Import de fonctions C++

- attention à la compatibilité de la licence
- dans DESCRIPTION,
  LinkingTo : otherpackage
- dans le fichier C++,
  #include <otherpackage.h>

C'est à vous #7

- ajoutez le fichier *find_closest.cpp* à votre package
- exportez la fonction *cpp_find_closest_elements* et documentez-la
- vérifiez votre package


Finalisation
&
Valorisation

10. Finalisation

Dernières vérifications

- sur la version actuelle de *R-dev*
- aucun *"errors", "warnings"* et *"notes"*
- *R CMD check --as-cran*
- avec la dernière version des packages dont votre package dépend (*update.packages()*)
- au moins sur deux OS (voir *devtools::build_win()* et *win-builder* pour Windows et *Travis* pour Linux)
- sans endommager les packages qui dépendent du vôtre (*tools::check_packages_in_dir(reverse = list())* ou *devtools::revdep_check()*)

11. Valorisation

Soumettre à CRAN

- en tant que maintainer
- lire "*CRAN Repository Policies*"
- commenter votre soumission

- *devtools::release()*
- ou formulaire de soumission à http://cran.r-project.org/submit.html
- pas d'envoi par mail

11. Valorisation

CRAN accepte votre package

- "On CRAN now"
- 48h pour réaliser toutes les vérifications
- création des binaires pour chaque plateforme

De Prof Brian Ripley <ripley@stats.ox.ac.uk>
Sujet **Re: CRAN submission CINID 1.2**
Pour Moi, CRAN

```
On CRAN now.

On 07/10/2014 10:37, Aurelie Siberchicot wrote:
 [This was generated from CRAN.R-project.org/submit.html]

 The following package was uploaded to CRAN:
 ===========================================

 Package Information:
 Package: CINID
 Version: 1.2
 Title: Curculionidae INstar IDentification
 Author(s): Aurelie Siberchicot, Adrien Merville, Marie-Claude Bel-Venner
    and Samuel Venner
 Maintainer: Aurelie Siberchicot <aurelie.siberchicot@univ-lyon1.fr>
 Description: This package provides functions to compute a method for
    identifying the instar of Curculionid larvae from the
    observed distribution of the headcapsule size of mature
    larvae.
 License: GPL (>=2.0)
 Imports: graphics, utils, stats


 The maintainer confirms that he or she
 has read and agrees to the CRAN policies.

 Submitter's comment: Sorry for the new update. Thanks.



--
Brian D. Ripley,                  ripley@stats.ox.ac.uk
Emeritus Professor of Applied Statistics, University of Oxford
1 South Parks Road, Oxford OX1 3TG, UK
```

11. Valorisation

Publier

- *R Journal*
- *Journal of Statistical Software*
- mailing list *r-packages*

11. La maintenance

- garder un œil sur le "*CRAN checks*" de votre package
- répondre aux sollicitations du CRAN
- tenir compte des évolutions de *R* et des packages dont vous dépendez
- veiller à la pérennité du rôle et du mail du maintainer
- préparer la prochaine version

12. A retenir

Soumettre à CRAN

- respecter les contraintes structurelles d'un package
- prendre soin de faire la documentation
- toujours se référer à "*Writing R Extensions*"

Une démarche qualité

- un code libre mais propre
- des outils adaptés pour le développement logiciel

Une maintenance raisonnée

- réagir aux évolutions de *R*
- conserver un mail de maintenance valide

13. A lire

- ***Writing R Extensions***, la référence de la *R Core Team,* mise à jour à chaque nouvelle version de *R*

- ***R Packages***, site internet et livre de H. Wickham, pour créer un package avec *devtools*, dans *RStudio*
- ***Software for Data Analysis***, livre de J. Chambers
- ***Advanced R***, livre de H. Wickham

- ***Guidelines for Rd files*** (http://developer.r-project.org/Rds.html)
- ***Parsing Rd files*** (http://developer.r-project.org/parseRd.pdf)


I
R!