Science des données V : module 1

Modularisation

Philippe Grosjean & Guyliann Engels

Université de Mons, Belgique
Laboratoire d'Écologie numérique des Milieux aquatiques

http://biodatascience-course.sciviews.org
sdd@sciviews.org

UMONS

Pourquoi modulariser son analyse ?

UMONS

Les dérives du Notebook

- Le R Notebook (et R Markdown) sont très pratiques pour rassembler les analyses (les instructions R) et la partie narrative du rapport.
- Mais si les analyses nécessitent beaucoup de code, la partie narrative est morcelée
- Les instructions ne sont **pas réutilisables** d'un notebook à l'autre

Solution

Extraire les instructions volumineuses dans des scripts et récupérés à l'aide de la fonction `source()`. Rendre son code réutilisable en écrivant des **fonctions()**, de préférence optimisées.

UMONS

Ranger avant d'arranger...

- Avant de modulariser vos analyses, pensez à **faire du ménage !**
  - Utilisez-vous un projet (et un dépôt Git) ?
  - Les sous-répertoires et noms de fichiers sont-ils logiques et organisés ?
  - Utilisez-vous des **chemins relatifs** pour que le proejt soit portable ?
  - Les chunks dans le notebook sont-ils morcelés de manière logique ?

UMONS

Sortir le code du Notebook

- Si les chunks restent trop volumineux, ou si les instructions doivent être réutilisées, **il faut les sortir**
- Un script R est tout indiqué pour les acceuillir

Exercice

Sur base du document Markdown fourni, extrayez le code volumineux dans un script R.

UMONS

Fonctions

- Les fonctions sont, dans R, les briques de construction de base.
- La règle : “si des instructions sont réutilisées plus de deux fois, il faut les transformer en fontion.”

Exercice

Transformation de mois (1, 2, 3, ..., 12) en trimestres (1, 2, 3, ou 4). Ecrivez le code qui fait ce calcul et transformez-le en fonction.

UMONS

Optimisation

- Dans les fonctions plus encore que dans les chunks, il est important que le code soit **optimisé**
  - Code plus clair, plus concis, **plus lisible**
  - Code **plus rapide** (vérifiez avec `bench::mark()` + menu 'Profile)
  - Code utilisant **moins de mémoire** (`pryr::mem_used()`)
  - Les **tests** sont utiles pour vérifier que la fonction effectuent bien le travail voulu

La référence : Advanced R par Hadley Wickham, section "performant code".

Exercice

Optimisez le code de la fonction `trimester()`. Comparez et discutez des différentes solutions que vous aurez obtenues.

UMONS

Simulation

Les simulations, basées sur la générations de nombres pseudo-aléatoires dans R, sont très utiles pour tester des idées en statistique. Le mieux est de voir en pratique avec un petit exemple simple.

Exercice

Considérez le jet d'une pièce de monnaie en l'air. Répondez à la question suivante : à quelle vitesse obtient-on en moyenne FPP ou FPF ? Est-ce la même valeur ?

UMONS

Application

Optimisez votre Notebook en écrivant des fonctions là où c'est utile.

UMONS