svTidy et le « formula-masking » : une alternative intéressante à {dplyr} et {tidyr} pour remanier vos jeux de données

Philippe Grosjean 1[1] Guyliann Engels 1[2]

Résumé (max 300 mots)

Le package {svTidy} est une alternative à {dplyr} et {tidyr} du Tidyverse pour remanier des tableaux de données. Il expose une interface similaire avec des fonctions qui ont des noms et des arguments quasi identiques à celles des deux packages bien connus. Cependant, il remplace la « tidy evaluation », l'approche non standard d'évaluation des arguments dans le Tidyverse, par le « formula-masking » (utilisation des objets formula de R). Cela a de nombreux avantages qui seront développés dans cette présentation : instructions plus explicites, pas de conflits de nom des variables, code plus rapide, plus facile à inclure dans une fonction et directement compatible avec le `R CMD check` pour les packages R.

Mots-clefs (3 à 5) : Manipulation de données - SciViews - svTidy - formula-masking - Tidyverse

Développement

Le Tidyverse ne doit plus être présenté : il est largement utilisé, notamment en remaniement de données sous R. Son succès tient à sa modularité et à sa syntaxe facile à lire et à écrire. Cependant, la « tidy evaluation » qui est à la base de l'implémentation des fonctions des packages {dplyr} et {tidyr}, n'a pas que des qualités. Elle est référentiellement non transparente et impose d'adopter des solutions contournées et complexes pour permettre le remplacement de variables dans le code Tidyverse. C'est le « data-masking » et les mécanismes complémentaires de « quasi-quotation » et d'injection de variables à l'aide des opérateurs `!!`, `{ }` et `:=`. De plus, la « tidy evaluation » n'est pas compatible avec les outils de vérification du code dans les packages R (`R CMD check`) et la solution proposée –précéder les variables du data frame par `.data$`, et les variables d'environnement par `.env$`– rend au final le code Tidyverse plus compliqué et répétitif que ne le ferait une simple évaluation standard des arguments de la fonction.

Nous proposons une approche alternative utilisant la même syntaxe que les fonctions {dplyr} et {tidyr}, mais avec, au choix, une évaluation standard des arguments (et donc, référentiellement transparente), ou avec une interface formule, appelée ici « formula-masking ». Cette dernière est intéressante à plus d'un titre :

[1]Service d'écologie numérique, Institut Complexys & Infortech, Université de Mons, Belgique, philippe.grosjean@umons.ac.be

[2]Service d'écologie numérique, Institut Complexys & Infortech, Université de Mons, Belgique, guyliann.engels@umons.ac.be

1. Il s'agit d'objets standard du langage R, bien connus des utilisateurs dans d'autres contextes, par exemple avec les fonctions `lm()` et `glm()`.
2. Les formules dans R sont invariablement construites à l'aide de l'opérateur ~ qui ne sert à rien d'autre. Il est ainsi très facile de déterminer lorsqu'une formule est utilisée dans du code R, et donc, lorsque le « formula-masking » entre en jeu. Cela enlève l'ambiguïté d'une évaluation standard ou non standard du code.
3. Le remplacement de variables est facile et rapide dans les formules à l'aide de la primitive `substitute()`. Cela permet d'écrire un code plus concis et plus rapide que la « tidy evaluation ».
4. Grâce aux formules, il est possible de contourner le problème de la création de noms de variables qui seraient identiques au nom d'arguments de la fonction. Dans le Tidyverse, la « solution » utilisée consiste à précéder les arguments qui peuvent entrer en conflit par un point : `.by=`, par exemple dans `filter()` ou `slice()`, alors qu'ailleurs c'est `by=` qui est utilisé, comme dans `slice_head()` ou `slice_min()`. Mais du coup, on ne peut plus appeler une variable `.by`, et cela ne fait que déplacer le problème. Ce dernier se rencontre aussi pour `.data=`, `.groups`, etc. créant par là même une hétérogénéité mal venue dans l'API Tidyverse. Avec le « formula-masking », il est possible de nommer les variables à créer avec des fonctions telles que `mutate_()` ou `summarise_()` sans entrer en conflit avec le nom des arguments de la fonction.
5. La syntaxe alternative de {svTidy} permet une transition plus facile et plus intuitive du code depuis un script vers une fonction. Les modifications à réaliser sont minimales.
6. Le code écrit avec le « formula-masking » est compatible avec le `R CMD check` pour les packages R, ce qui n'est pas le cas du code utilisant la « tidy evaluation ». Cela facilite grandement l'inclusion de fonctions de manipulation de données dans des packages R, sans devoir recourir à des astuces pour contourner les problèmes engendrés par l'évaluation non standard.

Le package {svTidy} réimplémente complètement la quasi-totalité des fonctions {dplyr} et {tidyr}, en acceptant les deux options (évaluation standard ou « formula-masking »). Ses fonctions sont également souvent plus rapides que leurs équivalents Tidyverse. {svTidy} représente au final une alternative à {dplyr} et {tidyr} qui permet d'écrire un code plus explicite, et donc, moins susceptible d'erreurs cachées. Il n'est pas encore disponible sur CRAN, mais il peut être installé depuis R-Universe :

```
install.packages('svTidy', repos = c('https://sciviews.r-universe.dev',
  'https://cloud.r-project.org'))
```

Des exemples illustrent les avantages du « formula-masking » ici : https://www.sciviews.org/svTidy/.