

# Simplifiez, fiabilisez et accélérez vos workflows R complexes avec {targets}

Vincent Guyader\*

## Résumé

La gestion de workflows R complexes peut vite devenir laborieuse et source d’erreurs, surtout si vous devez tout relancer manuellement à chaque modification. Le package {targets} propose une solution efficace pour automatiser ces workflows tout en assurant un suivi reproductible. Il vous suffit de définir les étapes cibles (ou « targets ») de votre analyse et leurs dépendances. {targets} orchestre alors l’exécution : seules les étapes nécessaires sont recalculées en cas de modification des données ou du code, évitant ainsi tout calcul redondant.

Par exemple, vous définissez une cible pour importer un jeu de données, une pour le nettoyer et une pour générer un graphique. Si les données changent, {targets} ne recalculera que les étapes concernées (nettoyage, graphique) sans toucher au reste. Ce principe rappelle le package {memoise}, qui met en cache les résultats de fonctions pour éviter les recalculs inutiles ; de même, {targets} conserve les résultats intermédiaires de chaque étape. Cette mise en cache évite de répéter des calculs coûteux et améliore les performances. Vous gagnez du temps et profitez d’un workflow plus robuste et reproductible, avec moins de risques d’oubli ou d’erreur – vous pouvez ainsi vous concentrer sur l’essentiel : votre analyse.

**Mots-clés** : Automatisation - Reproductibilité - Workflow

## Développement

Dans cette présentation, nous commencerons par introduire le concept de mise en cache en nous appuyant sur le package {memoise} comme modèle théorique. Ce package démontre comment le stockage temporaire des résultats d’une fonction permet d’éviter des recalculs inutiles, améliorant ainsi l’efficacité des analyses répétitives.

Nous passerons ensuite à {targets}, un outil qui étend ce principe à l’automatisation des workflows en R. À l’aide d’un exemple simple, nous illustrerons comment définir un pipeline de calcul composé de cibles interdépendantes : depuis l’importation des données, leur traitement, jusqu’à la génération d’une visualisation. L’idée est de suivre pas à pas l’exécution de ce workflow, en mettant en évidence comment {targets} détecte automatiquement les modifications et ne recalcule que les étapes concernées, garantissant ainsi une exécution rapide et reproductible.

La démonstration live permettra de voir concrètement le fonctionnement de ces mécanismes, en observant en temps réel le recalcul sélectif des cibles suite à une modification. Pour conclure, nous ouvrirons la discussion sur des cas d’usage plus complexes, afin de montrer que cette approche, bien que présentée via un exemple simplifié, est entièrement adaptable à des projets d’envergure nécessitant une gestion fine des dépendances et une optimisation des temps de calcul.

## Références

1. Landau W. et al. (2023). *targets: A Pipeline Toolkit for Reproducible Workflows in R*. R package version 0.1.6. Disponible sur CRAN : <https://CRAN.R-project.org/package=targets>.

---

\*ThinkR, [vincent@thinkr.fr](mailto:vincent@thinkr.fr)

2. Landau W. et al. (2023). *targets* – code source et documentation. Dépôt GitHub : <https://github.com/ropensci/targets>.
3. Wickham H. (2023). *memoise: Memoization for R*. R package version 2.0.1. Disponible sur CRAN : <https://CRAN.R-project.org/package=memoise>.
4. Wickham H. (2023). *memoise* – code source et documentation. Dépôt GitHub : <https://github.com/r-lib/memoise>.