Rパッケージ環境の保存と復元

鈴木 了太
suzuki@ef-prime.com
株式会社 ef-prime

2025年11月23日
統計数理研究所共同利用研究集会「データ解析環境Rの整備と利用」

目的：R実行環境の保存と復元

R実行環境の保存と復元はさまざまな場面で求められる

- 再現可能研究（reproducible research）
- スクリプトの開発・保守

目標：Rパッケージ環境の保存と復元

ハードウェア環境の再現は必ずしも容易ではないが、
ソフトウェア環境は比較的再現しやすい

オペレーティングシステム
　OSのインストーラ（ISOイメージまたはメディア）、
　Dockerコンテナなどの仮想環境
R
　インストーラパッケージ、ソースコード、仮想環境

上記が再現できる前提で、**Rパッケージ環境の保存と復元**にフォーカス

理想

研究・開発の段階から再現可能な環境を整備。例えば：

- コンテナ環境を構築し、OS、R、パッケージを一括で保管
- `renv`パッケージでプロジェクトごとにパッケージを管理
- `miniCRAN`パッケージでポータブルなローカルレポジトリを作成

⇒ これらを最初から利用することが望ましい

現実

Rスクリプトの再現可能性は必ずしも十分ではない（特に古いもの）

実行環境レベルの再現可能研究はこの10年程度で普及：

- `Sweave`（Leisch, 2002）は「文芸的プログラミング」の系譜
  「ひとつのソースにコードとドキュメントを含む統計レポート」
- 「パッケージを含む実行環境そのもの」の再現が普及したのは
  おそらく2014年以降

  `packrat`
  : Packrat: A Dependency Management System for R（Allaire, 2014）

  `miniCRAN`
  : Introducing miniCRAN: an R package to create a private CRAN repository（Vries, 2014）

よくある問題

- 10年前に書いたRスクリプトを動かしたい
- 現行の環境ではエラーで停止
- 当時のバージョンのRをインストールしても問題が生じる
  - CRANから取得するパッケージは現行版のため、エラーで停止
  - 当時のバージョンのパッケージを取得しても、依存関係の解決を`install.packages()`関数に任せると現行版を取得してしまい、互換性がなくエラーで停止

依存関係も含めて、当時のパッケージ環境を再現する必要がある

⇒ あの頃に戻りたい...

かつての解決策

CRAN (MRAN) Time Machine

- Microsoft社が提供していたCRANスナップショット機能
  - バイナリ、ソースコードいずれも対応
- 日付を含むURLを指定すると、その時点のバージョンをインストール可能
  - 2014/9/17以降をサポート
    https://www.r-bloggers.com/2019/05/mran-snapshots-and-you-2/
  - 2023/7/1にサービス停止
    https://www.r-bloggers.com/2023/01/mran-time-machine-will-be-retired-on-july-1/

例：2023/3/1時点の`ranger`パッケージをインストール（現在は動作しない）

```
install.packages("ranger",
  repos = "https://cran.microsoft.com/snapshot/2023-03-01/")
```

現在のおすすめ

Posit Package Manager (PPM)

https://packagemanager.posit.co

- CRAN Time Machineと同様にスナップショットが利用可能
  - バイナリ、ソースコードに対応（ただし一部バイナリは対象外）
  - 2017/10/10以降をサポート
    https://posit.co/blog/migrating-from-mran-to-posit-package-manager/
  - Posit社がソースからバイナリを独自ビルド
    https://posit.co/blog/the-road-to-building-ten-million-binaries/

例：2023/3/1時点の`ranger`パッケージをインストール

```
install.packages("ranger",
  repos = "https://packagemanager.posit.co/cran/2023-03-01")
```

もうひとつの選択肢

`dateback`パッケージ

https://github.com/r-suzuki/dateback

仮想CRANスナップショットのように動作

- CRANから特定の日付のパッケージを依存関係も含めて取得
- 外部サービスに依存せず、R上で動作するオープンソースソフトウェア
- **ソースコードのみ**対応、原理上は日付の制限なし（詳細は後述）

例：2023/3/1時点の`ranger`パッケージをインストール

```
dateback::install("ranger", date = "2023-03-01",
  repos = "https://cloud.r-project.org")
```

使用例

準備

datebackパッケージをCRANからインストール

```
install.packages("dateback")
```

パッケージのインストール

例：2024/9/1時点のrangerパッケージを依存関係も含めてインストール

```
dateback::install(pkgs = "ranger", date = "2024-09-01")
```

> library(dateback)の後にinstall()でも動作するが、基本的には
> dateback::install()の形式を推奨

使用例

ローカルレポジトリの作成

ディレクトリ`local_repo`を指定してパッケージを収集：

```
dateback::collect(pkgs = "ranger", date = "2024-09-01",
                  outdir = "local_repo")
```

指定したディレクトリに**ローカルレポジトリ**が構築される：

```
local_repo/src/contrib/
  PACKAGES  PACKAGES.gz  ranger_0.16.0.tar.gz
  Rcpp_1.0.13.tar.gz     RcppEigen_0.3.4.0.2.tar.gz
```

ローカルレポジトリからのインストール。別マシンでも動作：

```
install.packages(pkgs = "ranger", repos = "file:local_repo")
```

使用例

取得パッケージの一覧

2025/11/18時点での実行ログ（抜粋）：

```
    package                          file       date  status
1      Rcpp          Rcpp_1.0.13.tar.gz 2024-07-17 archive
2 RcppEigen RcppEigen_0.3.4.0.2.tar.gz 2024-08-24  latest
3    ranger        ranger_0.16.0.tar.gz 2023-11-12 archive
```

- `ranger`は`Rcpp`および`RcppEigen`に依存
- `RcppEigen`のみ最新版、他のふたつはアーカイブから旧版を取得

使用例

より複雑な例

依存関係が多い、または複雑な場合に特に有用。以下は
rstanパッケージについて、2024/9/1時点のファイルを取得した例：

```
          package                          file       date  status
1    RcppParallel   RcppParallel_5.1.9.tar.gz 2024-08-19 archive
2            Rcpp           Rcpp_1.0.13.tar.gz 2024-07-17 archive
3      RcppEigen  RcppEigen_0.3.4.0.2.tar.gz 2024-08-24  latest

...

45        ggplot2        ggplot2_3.5.1.tar.gz 2024-04-23 archive
46             BH          BH_1.84.0-0.tar.gz 2024-01-10 archive
47          rstan         rstan_2.32.6.tar.gz 2024-03-05 archive
```

当時のrstanは14パッケージに依存（base/recommendedを除く）。
再帰的な依存関係も含めると、計**47パッケージ**を取得する必要があった

動作の概要

CRANにアクセスし、以下に相当する処理を実施：

1. パッケージを検索。見つからなければ`/src/contrib/Archive/`も参照
2. パッケージの最新リリース日を確認（例：`2024-11-08`）
3. リリース日が指定日（例：`2024-09-01`）以前であれば最新版を取得。
   そうでなければ指定日以前の最新のアーカイブを取得
4. 取得した`tar.gz`ソースを展開し、依存パッケージを確認。
   これらについても再帰的に上記の手順を適用

> 依存パッケージについて、既にダウンロード済みのパッケージは再処理を行わない。そのほか既定の挙動は以下の通り（オプションで変更可能）：
>
> - `install()`では実行時にインストール済みのパッケージはスキップ
> - `collect()`ではインストール済みのパッケージも取得するが、`recommended`パッケージはスキップ

仕様の詳細

動作要件

- 現時点では`R >= 3.2.0`（2015/4/16）が必要
  - 少なくとも10年前までは比較的容易に対応可能
  - `R >= 3.2.0`において`collect()`でローカルレポジトリを作成し、`R < 3.2.0`から利用することも可能
- CRANの仕様に依存。特に過去バージョンの取得はWebスクレイピングを用いているため、将来的にサイト構造が変わると改修が必要
- なるべく多くの環境で動作させるため、外部パッケージに依存しない設計

> 開発当初はWebスクレイピングのために`curl, desc, httr, rvest`などのパッケージを利用していたが、上記方針のためbase-Rで書き直した。同様に`dplyr`や`%>%`パイプも不使用

仕様の詳細

対象パッケージ

原理上はCRAN上にあるすべてのパッケージおよびバージョンが対象。
現時点では以下に注意：

- `R-1.9.0`（2004/4/12）でパッケージの再編があり、統合前のパッケージが`DESCRIPTION`ファイルに含まれる場合、アーカイブからの取得に失敗する場合がある
  - 例：2003/8/1時点の`Hmisc`は`mva`に依存しており、これが原因で失敗
  - R-1.9.0では`ctest, eda, lqs, modreg, mva, nls, stepfun, ts`が`stats`に統合（ https://developer.r-project.org/190update.txt）

活用例

Windows/Macおよびコンテナ以外のLinux環境

- 2017/10/10以降の日付
  - Posit Package Managerを利用（ただし一部バイナリは要ビルド）
- それより前の日付（**パッケージをビルドできる環境が必要**）
  - `R >= 3.2.0`で`dateback::install()`
  - `R >= 3.2.0`で`dateback::collect()`によって
    ローカルレポジトリを構築し、`R < 3.2.0`にコピーして利用

Linuxコンテナ環境の構築

`dateback::install()`でコンテナからインストール、または
`dateback::collect()`でローカルレポジトリをあらかじめ用意

補足

- Windows上のRで`collect()`したローカルレポジトリのファイルを、Docker DesktopのLinuxコンテナからマウントして利用することも可能
- 2017/10/10以降の日付が対象であれば、PPM + `miniCRAN`の組み合わせでも特定の日付におけるローカルレポジトリを構築可能

まとめ

- 再現可能研究やスクリプトの開発・保守において、
  R実行環境の保存と復元が重要
- 研究・開発の初期段階から、コンテナ環境や`renv`、`miniCRAN`
  などを利用すると効果的
- 特定の日付におけるパッケージ環境を再現する方法として、
  Posit Package Manager（PPM）や`dateback`パッケージが利用可能
  - 2017/10/10以降の日付はPPM、それより前は`dateback`
  - `dateback`によるローカルレポジトリはコンテナ環境構築に便利