The application

Muhammad Yaseen

2026-11-05

Table of contents

1 Abstract **1**

2 Starting the application **1**

3 The pages **2**

4 Reproducibility **4**

5 Deploying **5**

6 References **6**

1 Abstract

`dmai_app()` opens an application in the web browser that computes monetary aggregates without code: the data and its columns, one index with its weights and user costs, every method side by side, the user costs and the Divisia second moments. This vignette describes its pages and how it keeps its results reproducible.

2 Starting the application

```
dmai_app(
    launch.browser = TRUE
  )
```

The application needs the suggested packages **shiny**, **bslib** and **DT**. **magick** with WebP support is optional: with it the plots are sent to the browser as WebP images, smaller than the same PNG.

3 The pages

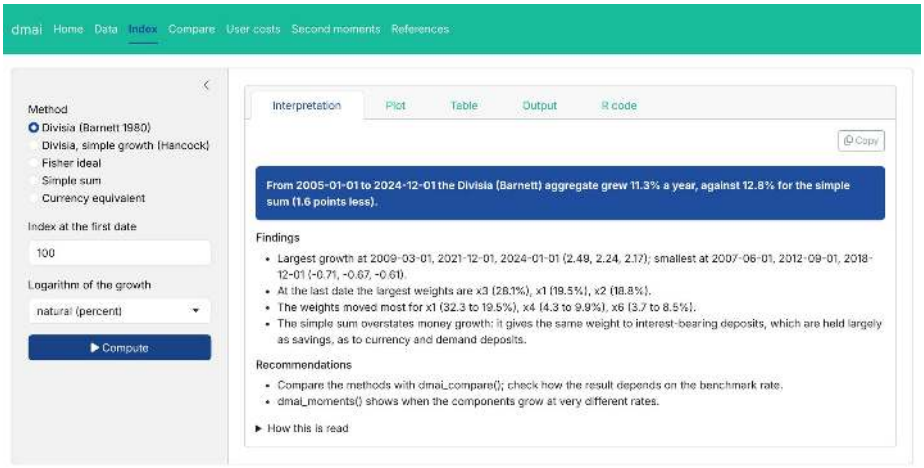
Page	What it does
Home	what the application does and the order of the pages
Data	the simulated example `money_sim` or a CSV file with one row per date; the date, quantity and rate columns (in the same order), the benchmark rate (a column, or the largest component rate) and whether the rates are in percent
Index	one method (Divisia, Divisia with simple growth, Fisher, simple sum, currency equivalent), the value at the first date and the logarithm of the growth; interpretation, plot, table, printed output and R code
Compare	the chosen methods side by side, as indexes or growth rates
User costs	the user cost and weight of every component over time
Second moments	the Divisia variance of the growth of the components
References	the articles behind the methods, as `dmai_cite()` prints them

Every result page has the same tabs:

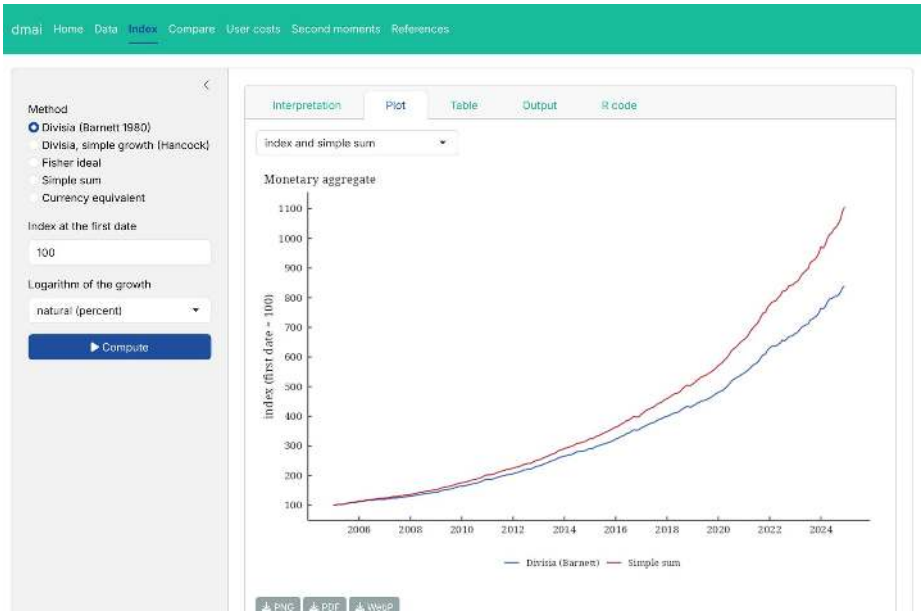
- **Interpretation** gives the plain-language reading of `dmai_interpret()`, with a button that copies it.
- **Plot** shows the figure, which can be saved as PNG, PDF or WebP.
- **Table** shows the numbers, rounded, in a sortable table with buttons to copy them or save them as CSV or Excel.
- **Output** shows the printed result as text, with a button that copies it.

The index page has a further tab, **R code**, with the code that reproduces the result outside the application.

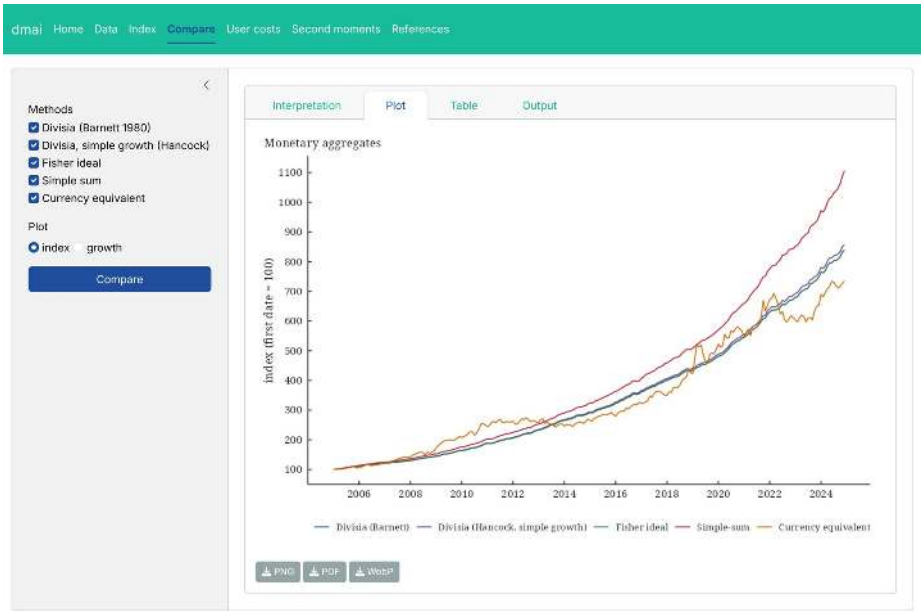
The index page after the Divisia aggregate of `money_sim` with the bond yield `rb` as benchmark:



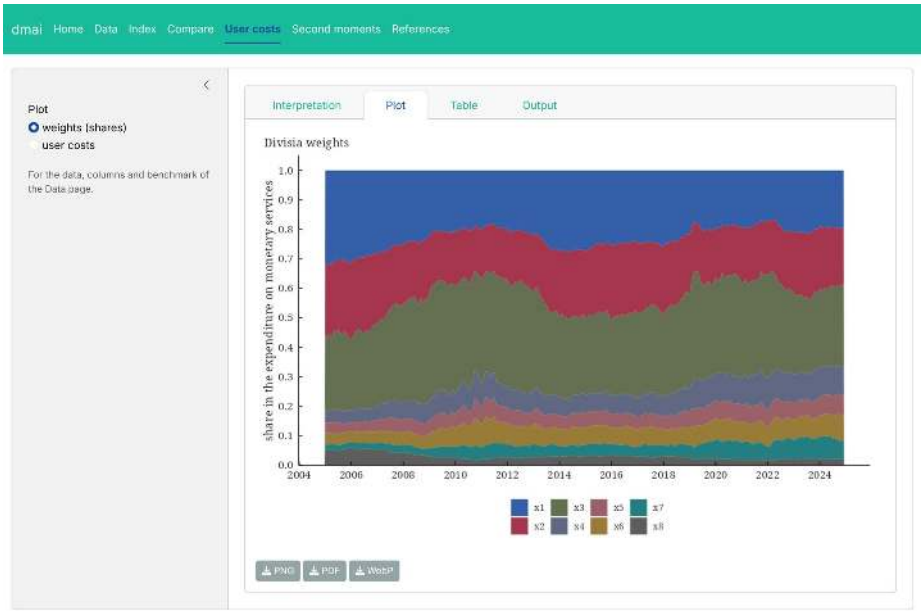
its plot, the aggregate against the simple sum:



the comparison of the methods:



and the weights of the components:



4 Reproducibility

For the settings of the screenshots, the R code tab shows:

```
settings <-
  list(
      source  = "example"
    , method  = "Barnett"
    , logbase = "e"
    , date    = "Date"
    , x       = paste0("x", 1:8)
    , r       = paste0("r", 1:8)
    , bench   = "rb"
    , percent = TRUE
    , base    = 100
    )
cat(dmai:::.dmai_app_code(input = settings))
```

```
library(dmai)
data(money_sim)
dat <- money_sim
fit <-
  dmai(
      .data     = dat
    , method    = "Barnett"
    , logbase   = NULL
    , date      = "Date"
    , x         = c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8")
    , r         = c("r1", "r2", "r3", "r4", "r5", "r6", "r7", "r8")
    , user_cost = NULL
    , benchmark = "rb"
    , percent   = TRUE
    , base      = 100
    , as_dt     = FALSE
    )
fit
plot(x = fit)
dmai_interpret(
    x   = fit
  , top = 3
  )
```

The indexes are deterministic, so the code returns the same numbers as the application.

5 Deploying

`dmai_app_object()` returns the application object without running it, for a server:

```
# app.R on a Shiny server
dmai::dmai_app_object()
```

6 References

Barnett, W. A. (1980). Economic monetary aggregates: an application of index number and aggregation theory. *Journal of Econometrics*, 14(1), 11–48.