Wrattler: Interactive, smart and polyglot notebooks

Tomas Petricek, James Geddes, Charles Sutton

Next generation notebook system for data science

Using programming language techniques to make notebook systems for data science richer, more powerful and less error-prone.

Interactive

You often cannot know what will work until you try. We give quick feed-back to help you find out as early as possible.

Provenance

Running cells out of order can break reproducibility. We use provenance track-ing to make sure output is always up-to-date.

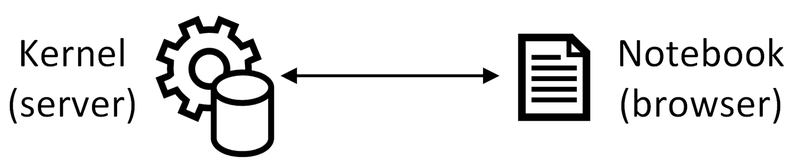
Polyglot

Data analysts need to use many tools, languages and packages. We makes it easy to mix JavaScript, Python, R and more.

Extensible

In traditional notebooks, all state is hidden. We make it public to allow tools like AI assistants help with tedious tasks.

Figure 1. In Jupyter, Notebook sends commands to Kernel, which is responsible for maintaining state and executing code. Wrattler separates state management from execution and lets Notebook control how to recalculate results after a code change. Multiple language runtimes are responsible for eval-uating code in different languages and they share data through data store.



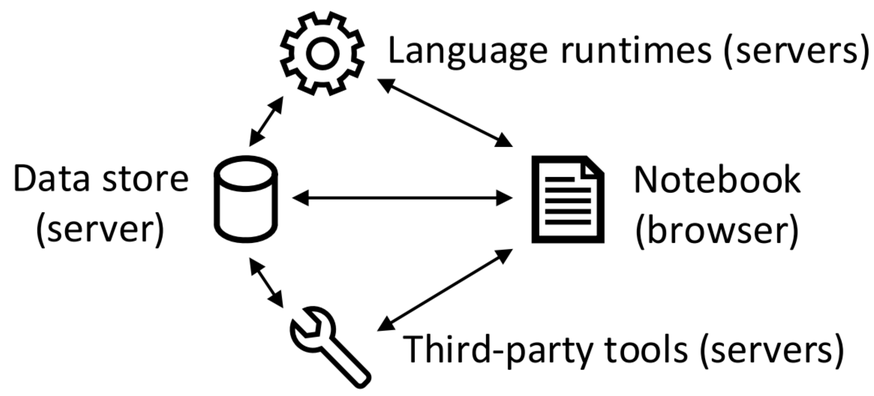


Figure 2. Wrattler supports tools and languages that evaluate code in the browser and can provide hints when writing code, such as auto-complete and also rapid feedback such as live previews.

```
let avgs2014 =
  web.loadTable("https://www.ofcom.org.uk/__data/assets/excel_doc/0014/74120/data_november_2014.csv.xls")
    .explore.'group data'.'by Urban/rural'.'average Download speed (Mbit/s) 24 hrs'.then
    .'filter data'.'Urban/rural is not'.''.then
    .'get series'.'with key Urban/rural'.'and value Download speed (Mbit/s) 24 hrs'

let avgs2015 =
  web.loadTable("https://www.ofcom.org.uk/__data/assets/excel_doc/0015/50073/panellist-data.csv.xls")
    .explore.'group data'.'by Urban/rural'.'average Download speed (Mbit/s) 24 hrs'.then
    .'filter data'.'Urban/rural is not'.

compost.charts.bar(avgs2014)
compost.charts.bar(avgs2015)
```

Rural
Urban

(current) avgs2014 avgs2015 (output) (output)

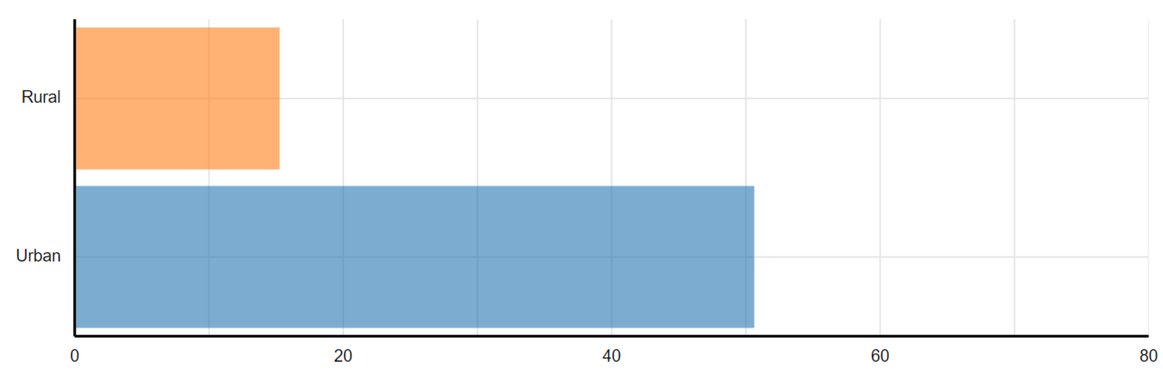


Figure 3. Wrattler inte-grates with standard data science tools such as R and Python, giving data scientists easy access to all the tools they know and love.

```
bball <- rbind(bb2014, bb2015fix)
bball <- bball %>%
  mutate(IsRural = ifelse(Urban == "Urban", 0, 1),
         YearAfter = ifelse(Year == 2014, 0, 1))
did_model <-
  lm(Down ~ IsRural + YearAfter +
     IsRural*YearAfter, data = bball)
print(summary(did_model))
```

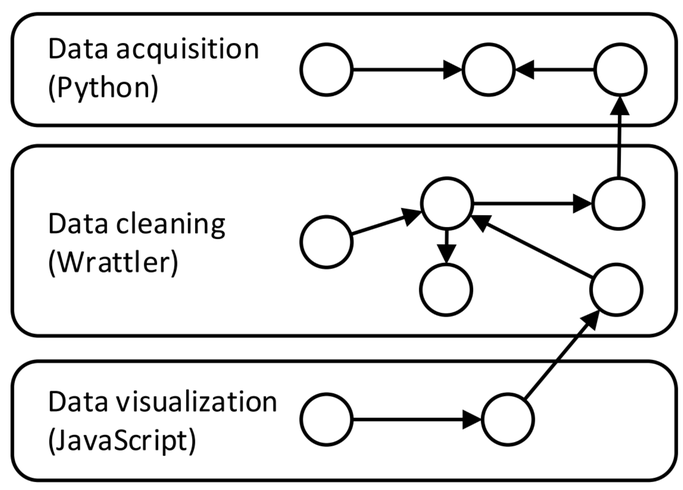
(console) bball

```
Call:
lm(formula = Down ~ IsRural + YearAfter + IsRural * YearAfter,
    data = bball)

Residuals:
   Min     1Q Median     3Q    Max
-55.91 -26.81 -12.30  15.36 238.42

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)        50.622      1.019  49.687  < 2e-16 ***
IsRural           -29.023      2.453 -11.832  < 2e-16 ***
YearAfter           6.063      1.432   4.233 2.34e-05 ***
IsRural:YearAfter   5.777      2.920   1.978    0.048 *
```

Figure 4. Notebook uses a dependency graph between cells to efficiently update previews after code change and to guarantee that re-evaluating notebook from scratch will reproduce the currently displayed results.



Tools and features enabled by Wrattler architecture

Separating execution from state management and dependency tracking allows features and tools that are (almost) impossible in other systems.

Browser-based languages

Evaluation is controlled by the notebook running in a browser and plugins can choose to run all code directly in the browser.

Smart AI assistants

AI assistants can look at data in the data store and make data wrangling recommendations without understanding R or Python

Notebook refactoring tools

Dependency graph let us extract code needed for a given output and produce a “clean” notebook without experimental code.

Advanced notebook versioning

Data store caches previous results and can keep past versions of code, letting users easily see previous steps in their research.

The Alan Turing Institute