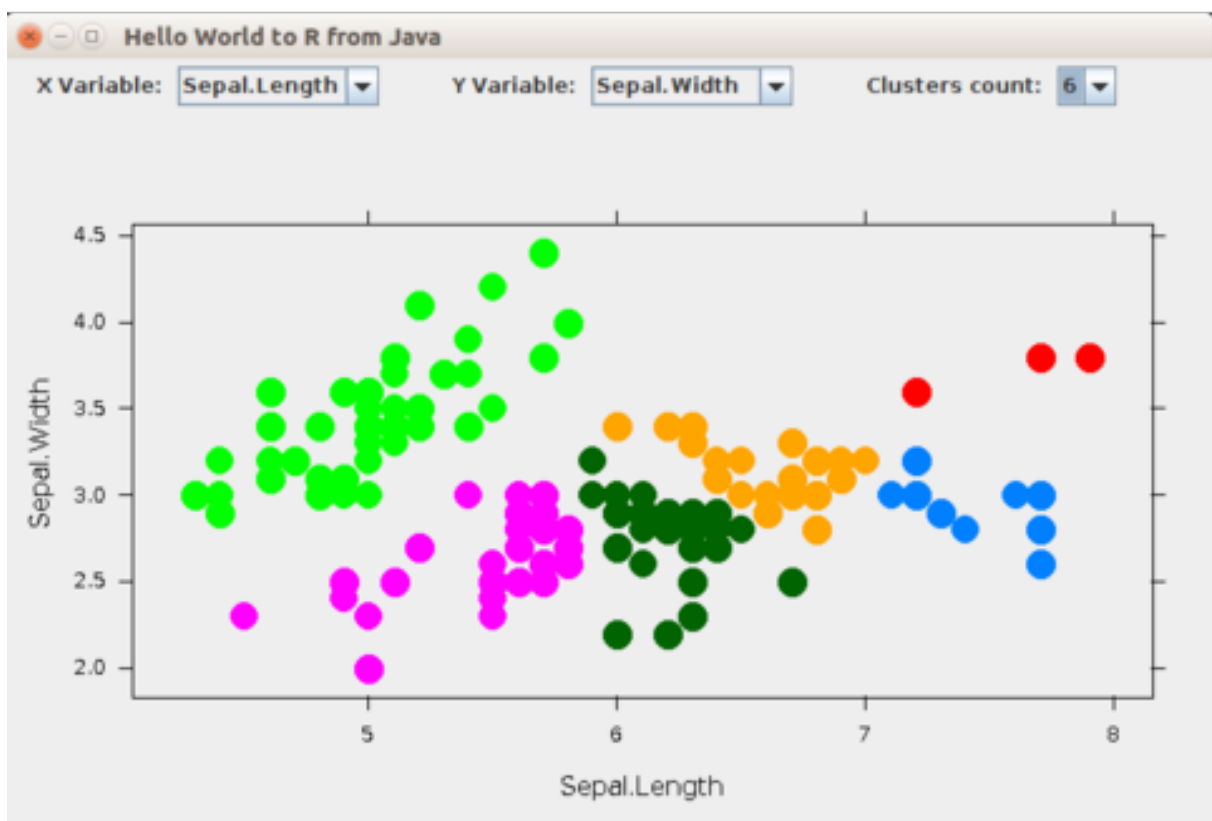

FastR

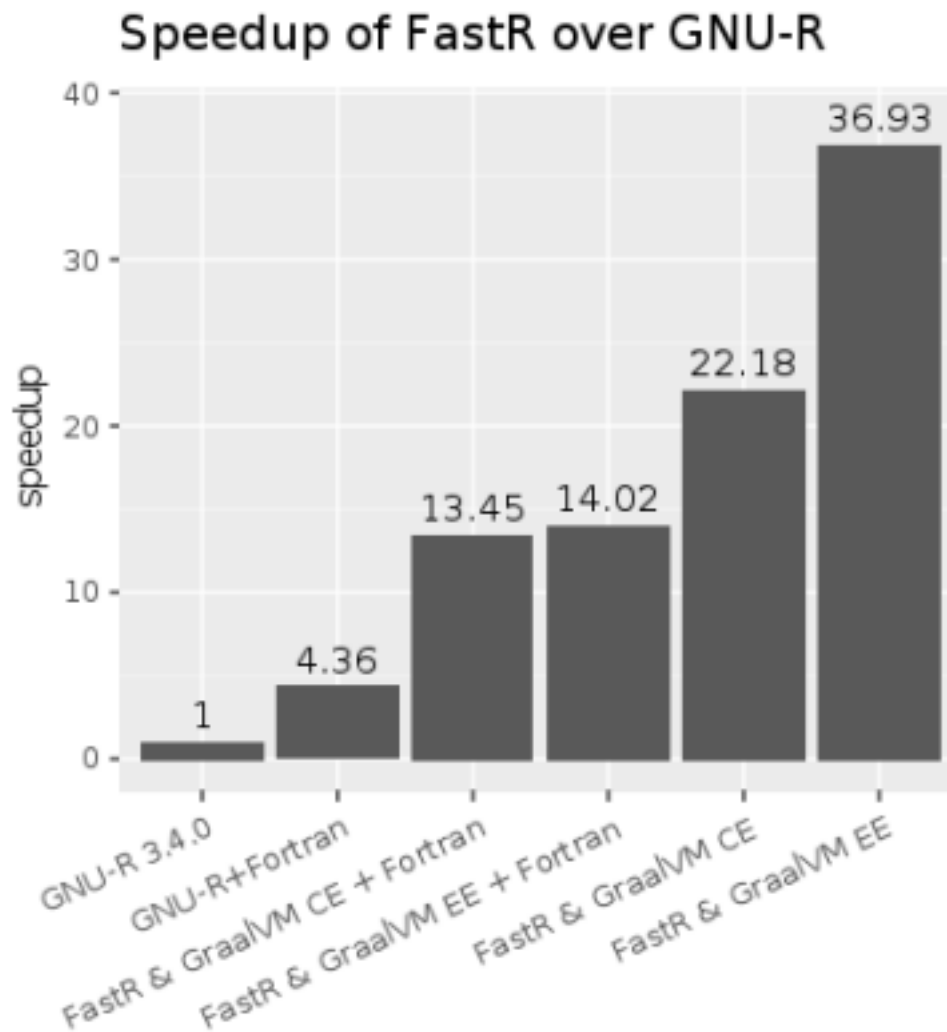
FastR is a high-performance implementation of the R programming language, built on GraalVM.

FastR aims to be:

- * efficient: executing R language scripts faster than any other R runtime and as fast as [Rcpp](#)
- * polyglot: allowing fast polyglot interoperability with other languages in the GraalVM ecosystem.
- * compatible: with the reference R implementation including the R extensions C API
- * embeddable: allowing integration using the R embedding API or the GraalVM polyglot embedding SDK for Java

The screenshot below shows Java application with embedded FastR engine. The plot below was generated by [ggplot2](#) running on FastR and it shows peak performance of the raytracing example. The measurements were reproduced independently.





Getting Started

See the documentation on the GraalVM website on how to get GraalVM and install and use FastR.

```
1 $JAVA_HOME/bin/R
2 Type 'q()' to quit R.
3 > print("Hello R!")
4 [1] "Hello R!"
5 >
```

Current Status

The goal of FastR is to be a drop-in replacement for GNU-R, the reference implementation of the R language, including the R extensions C API. FastR faithfully implements the R language, and any dif-

ference in behavior is considered to be a bug.

CRAN Packages

FastR can currently install and run basic examples of many of the popular R packages, such as `ggplot2`, `jsonlite`, `testthat`, `assertthat`, `dplyr`, `knitr`, `Shiny`, `Rcpp`, `quantmod`, and more. However, one should take into account **the experimental state of FastR**, there can be packages that are not compatible yet, and if you try FastR on a complex R application, it can stumble on those. If this happens, please submit an issue on GitHub.

To provide better stability, FastR uses by default a fixed snapshot of CRAN (via MRAN). Function `install.packages` therefore does not install the latest versions. This can be overridden by passing `repos` argument to `install.packages` pointing to CRAN.

FastR provides its own replacements for `rJava` and `data.table` packages, which can be installed with `install.fastr.packages(c("rJava", "data.table"))`.

Native Extensions Performance

Packages that use the R extensions C API in hot paths, especially via `Rcpp`, **may exhibit slower performance** on FastR due to the high cost of transitions between the native and managed code. This can be mitigated by using the GraalVm LLVM runtime. Preview of the support is available via the `--R.BackEnd=llvm` option. Note that most of the times FastR running R code equivalent to given `Rcpp` code is as fast as GNU-R/`Rcpp` and sometimes even faster because of the advanced optimizations of the Graal dynamic compiler.

Documentation

FastR reference documentation which explains its advantages, its current limitations, compatibility, and additional functionality is available on the GraalVM website.

Further documentation, including contributor and developer-oriented information, is in the documentation folder of this repository.

Stay Connected with the Community

See graalvm.org/community on how to stay connected with the development community. The discussion on Slack is a good way to get in touch with us.

We would like to grow the FastR open-source community to provide a free R implementation atop the Truffle/Graal stack. We encourage contributions, and invite interested developers to join in. Prospective contributors need to sign the Oracle Contributor Agreement (OCA). The access point for contributions, issues and questions about FastR is the GitHub repository.

Authors

FastR is developed by Oracle Labs and is based on the GNU-R runtime. It contains contributions by researchers at Purdue University ([purdue-fastr](#)), Northeastern University, JKU Linz, TU Dortmund and TU Berlin.

License

FastR is available under a GPLv3 license.