Installing RStan from Source

Jump to bottom

Jonah Gabry edited this page on Dec 19, 2020 · 3 revisions

Windows

First, ensure that you have configured your system to be able to compile C++ by following the instructions in Windows - Configuring C++ Toolchain.

First, remove any existing installations and configurations:

```
remove.packages("rstan")
if (file.exists(".RData")) file.remove(".RData")
```

and then restart R and set the desired number of cores to use during installation

```
Sys.setenv(MAKEFLAGS = "-j4") # four cores used
```

Finally, to install the CRAN version of RStan from source you can run:

```
install.packages("rstan", type = "source")
```

Or to install the development version of RStan from GitHub:

```
remotes::install_github("stan-dev/rstan", ref = "develop", subdir = "rstan/rstan")
```

Finally, you can test that your installation is working by running:

```
example(stan_model, package = "rstan", run.dontrun = TRUE)
```

The model should then compile and sample. You may also see the warning:

```
Warning message:
In system(paste(CXX, ARGS), ignore.stdout = TRUE, ignore.stderr = TRUE) :
  'C:/rtools40/usr/mingw_/bin/g++' not found
```

- Then proceed to How to Use RStan

Mac

First, ensure that you have configured your system to be able to compile C++ by following the instructions in Mac - Configuring C++ Toolchain.

You are now ready to install RStan from source. Execute in R

```
remove.packages("rstan")
if (file.exists(".RData")) file.remove(".RData")
```

and then restart R and set the desired number of cores to use during installation

```
Sys.setenv(MAKEFLAGS = "-j4") # four cores used
```

Install the main dependencies with the same compiler settings

```
install.packages(c("Rcpp", "RcppEigen", "RcppParallel", "StanHeaders"), type = "source")
```

Finally, either do

```
install.packages("rstan", type = "source")
```

to install the CRAN version of RStan from source or

```
remotes::install_github("stan-dev/rstan", ref = "develop", subdir = "rstan/rstan")
```

to install the development version of RStan from GitHub.

Linux

First, ensure that you have configured your system to be able to compile C++ by following the instructions in Linux - Configuring C++ Toolchain.

You are now ready to install RStan from source. Execute in R

```
remove.packages("rstan")
remove.packages("StanHeaders")
if (file.exists(".RData")) file.remove(".RData")
```

and then restart R and set the desired number of cores to use during installation

```
Sys.setenv(MAKEFLAGS = "-j4") # four cores used
```

Finally, either do

```
install.packages("rstan", type = "source")
```

to install the CRAN version of RStan from source or

```
remotes::install_github("stan-dev/rstan", ref = "develop", subdir = "rstan/rstan")
```

to install the development version of RStan from GitHub.

Special Note: CentOS 7.0

When installing rstan from source on CentOS 7, even if you have a compatible gcc compiler installed, you may have an error like

```
rstan /lib64/libstdc++.so.6: version 'GLIBCXX_3.4.20' not found (required by
/usr/lib64/R/library/rstan/libs/rstan.so)
```

pop up and terminate your install (or, after the install, your library load). This is a known issue on CentOS, and can often be worked around by ensuring that the LD_LIBRARY_PATH is set properly. To do this as a one-time fix, run

```
export LD_LIBRARY_PATH=/usr/local/lib:/usr/lib:/usr/local/lib64:/usr/lib64
```

before launching R and running one of the above commands. This can be setup as a permanent fix in the usual fashion. If you are using RStudio Server and want rstan to work for all your users, you can set the LD_LIBRARY_PATH in /etc/rstudio/rserver.conf, as

```
rsession-ld-library-path=/usr/local/lib:/usr/lib:/usr/local/lib64:/usr/lib64
```

which will ensure each session launched has appropriate access.

Clone this wiki locally

```
https://github.com/stan-dev/rstan.wiki.git
```