

# **Installation Instructions for the R-package**

## ***cpgen***

Claas HEUER

July, 2014

# Contents

<b>1</b>	<b>General requirements</b>	<b>1</b>
<b>2</b>	<b>Linux - Debian and Debian-derived Linux distributions (e.g. Ubuntu)</b>	<b>1</b>
2.1	Updating the GNU-Compiler (g++) . . . . .	1
2.2	Updating R . . . . .	2
2.3	Install the package from Repository . . . . .	2
<b>3</b>	<b>Windows</b>	<b>2</b>
3.1	Install the package from Repository . . . . .	3
<b>4</b>	<b>Mac</b>	<b>3</b>
4.1	Installation from Repository . . . . .	3
4.2	Compiling the source package with OpenMP . . . . .	3
4.2.1	Using the default Tool-Chain with <i>clang/clang++</i> . . . . .	3
4.2.2	Using the GNU-Compiler <i>gcc/g++</i> . . . . .	4

# 1 General requirements

Generally the package requires the following:

- R ( $\geq 3.1.0$ )
- C++11 - only supported from R-3.1.0 onwards
- GNU-Compiler g++ ( $\geq 4.6.3$ )
- OpenMP - comes with g++
- R-packages:
  - Rcpp
  - RcppEigen
  - Matrix

## 2 Linux - Debian and Debian-derived Linux distributions (e.g. Ubuntu)

Most Linux distributions come with the needed tool-chain for developing and compiling source packages.

Due to the general requirements of *cgen* some updating might be necessary. If you run Debian  $\geq 7.0$  (Wheezy) or Ubuntu  $\geq 12.04$  only updating R will be necessary.

### 2.1 Updating the GNU-Compiler (g++)

If some essential tools are missing for some reason:

```
sudo apt-get install build-essential
```

1. Obtain the sources from: <https://gcc.gnu.org/>

e.g.: <ftp://ftp.fu-berlin.de/unix/languages/gcc/releases/gcc-4.9.0/>

2. Compile the GNU-Compiler (<https://gcc.gnu.org/wiki/InstallingGCC>:

Move to the downloaded source package and run as root:

```
tar xzf gcc-4.9.0.tar.gz
cd gcc-4.6.2
./contrib/download_prerequisites
cd ..
```

```
mkdir objdir
cd objdir
$PWD/../../gcc-4.6.2/configure --prefix=$HOME/gcc-4.9.0
make
make install
```

3. Make sure to include the custom path of the compiled g++ in your PATH variable:

```
PATH=$HOME/gcc-4.9.0:$PATH
```

## 2.2 Updating R

1. Obtain the R-sources from: <http://www.r-project.org/>  
e.g.: <http://cran.rstudio.com/src/base/R-3/R-3.1.1.tar.gz>
2. Install all required packages to build R from source:

```
sudo apt-get build-dep r-base
```

3. Compile the source package. Move to the extracted source folder and run:

```
./configure
make
sudo make check install
```

## 2.3 Install the package from Repository

Start R-3.1.x

```
# installation of the package using the R-Forge repository
install.packages("cpgen", repos=c("http://R-Forge.R-project.org",
"http://cran.at.r-project.org"), dependencies=TRUE)
```

## 3 Windows

For a Windows installation of R you need to install these two binary packages:

1. **Rtools - 3.1**

<http://cran.r-project.org/bin/windows/Rtools/>

2. **R - 3.1.x**

e.g.: <http://cran.rstudio.com/bin/windows/base/R-3.1.1-win.exe>

## 3.1 Install the package from Repository

Start R-3.1.x

```
# installation of the package using the R-Forge repository
install.packages("cpgen", repos=c("http://R-Forge.R-project.org",
"http://cran.at.r-project.org"), dependencies=TRUE)
```

## 4 Mac

On a Mac the development tool-chain has to be installed.

1. **Xcode:** Go to: <https://developer.apple.com/xcode/> and get Xcode. A free developer registration is necessary
2. **Command-line Tools:** According to: <http://hpc.sourceforge.net/>
  - You will find the option to download the command-line tools in XCode's Preferences
  - On 10.9 Mavericks, you can get the command-line tools by simply typing:

```
xcode-select --install
```

R-3.1.x can be obtained as binary package from: <http://www.r-project.org/>

### 4.1 Installation from Repository

Start R-3.1.x

```
# installation of the package using the R-Forge repository
install.packages("cpgen", repos=c("http://R-Forge.R-project.org",
"http://cran.at.r-project.org"), dependencies=TRUE, type="source")
```

This is the easiest way to install the package, but it does not support OpenMP. Everything will run though, but not parallelized.

### 4.2 Compiling the source package with OpenMP

There are two options available in order to compile the package with OpenMP support. The GNU-Compiler way is recommended!

#### 4.2.1 Using the default Tool-Chain with *clang/clang++*

The first option is to use the standard compiler of Apples Xcode, which is *clang/clang++*. But as of now (July, 2014) this compiler does not support OpenMP by default. Make sure to have *Xcode* and *command-line tools* installed as described above. Follow the instructions from the Clang-OpenMP project: <http://clang-omp.github.io/>.

## 4.2.2 Using the GNU-Compiler *gcc/g++*

Make sure to have *Xcode* and *command-line tools* installed as described above. In order to use the GNU-Compiler rather than *clang* for compiling R-packages do the following:

### 1. Get the GNU Compiler

- Download from: <http://hpc.sourceforge.net/>
- e.g: <http://prdownloads.sourceforge.net/hpc/gcc-4.8-bin.tar.gz?download>
- Installation from terminal:

```
### switch to the download directory of gcc/g++
# unzip
gunzip gcc-4.8-bin.tar.gz

# install
sudo tar -xf gcc-4.8-bin.tar -C /

# add the g++ location to your PATH variable
export PATH=/usr/local/bin:$PATH
echo PATH=/usr/local/bin:$PATH >> ${HOME}/.bash_profile
```

### 2. Install the cpgen source package with g++

Download the most recent version of the package, set-up everything for compiling with the GNU-Compiler and install the package by simply pasting this into your terminal:

```
URL=https://gist.githubusercontent.com/cheuerde\
/6bd537175fa6fad3b16f/raw/
curl "$URL" | sh
```