The **gpusim** package
Installation and Compilation Guide

Marius Appel*
marius.appel@uni-muenster.de

August 6, 2012

*Institute for Geoinformatics, University of Muenster, Germany.

1 Installation and Compilation

Since the gpusim package is developed for using NVIDIA's CUDA technology und thus uses some NVIDIA libraries, the installation and compilation is more complex than just using R and run `install.package()`. This document shows how to install and build the package for Linux and Microsoft Windows platforms.

1.1 Linux Platforms

At first, you have to get the latest source code of gpusim. Using the Rforge svn, make a checkout by typing: `svn checkout svn://scm.r-forge.r-project.org/svnroot/gpusim/` You should receive two folders `pkg` and `www`. Ignore the `www` folder and rename the `pkg` directory to `gpusim`. Alternatively, download the tar.gz file and unpack the file.

You should now have a folder named gpusim, which is the package root directory that contains several sub folders (e.g. `src,man,R`).

Now, you have to compile the package using `gcc`, which is automatically called by `R CMD INSTALL gpugeo`. Since our package makes use of third party libraries from NVIDIA, you must have the following tools installed on your machine:

- Latest NVIDIA Graphics Drivers
- Latest version of NVIDIA's CUDA toolkit containing the libraries `libcuda.so`, `libcublas.so`, `libcudart.so`, `libcurand.so`, and `libcufft.so` and their corresponding header files.

We assume that your CUDA installation is located under the default path `/usr/local/cuda` where the sub folder `/include` contains the headers and `/lib64` contains the libraries. If you did not use the default installation path, open the package's makefile under `gpusim/src/Makefile` and change the value of `CUDA_HOME` to the right directory. If you are using a 32bit R version, you have to change the `CUDA_LIB_PATH` value to `$(CUDA_HOME)/lib`.

Typing `R CMD INSTALL gpugeo` finally installs the package.

1.2 Windows platforms

For Microsoft Windows platforms, binary releases are available for download. After downloading the .zip file, you can simply install the package in the RGui by selecting `packages -> install packages from local zip files` in the menu. Notice that there are different versions for x86 and x64 systems. However, the binaries only work with the latest CUDA versions. If there is a different CUDA version running on your workstation, you will need to compile the package on your own. This is quite intricately and you will need Microsoft's Visual Studio 2010 compiler. Run the following steps for the compilation:

1. Make sure you have installed the latest versions of R, Rtools and Microsoft Visual Studio 2010. You should also have set needed R environment variables (Rtools usually does it during installation) and you need a working CUDA toolkit with environment variable `CUDA_PATH`. Your `Path` variable should also contain the path of the `nvcc` compiler.

2. Download the latest source code of gpusim either using the Rforge svn or downloading the tar.gz file. Using svn, make a checkout in your favorite svn tool (e.g. Tortoise SVN) using the URL `svn://scm.r-forge.r-project.org/svnroot/gpusim/`. You should receive two folders `pkg` and `www`. Ignore the `www` folder and rename the `pkg` directory to `gpusim`. Alternatively, download the tar.gz file and unpack the file using your favorite archiving software (e.g. 7-Zip). You should now have a folder named gpusim, which is the package root directory that contains several sub folders (e.g. `src,man,R`).

3. Check, whether you want to compile the package for x32 or x64 R.

4. Open a command line window.

   a) If x86, load the Visual C++ environment variables by typing `"C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\bin\vcvars32.bat"`. This is the default path of the Visual Studio installation, change it if necessary. Notice that the quotation marks are necessary if the path contains space characters.

   b) If x64, load the Visual C++ environment variables by typing `"C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\bin\x86_amd64\vcvarsx86_amd64.bat"`. This is the default path of the Visual Studio installation, change it if necessary. Notice that the quotation marks are necessary if the path contains space characters.

5. Jump to the folder, where the package directory resides, e.g. type `cd "C:\Users\%USERNAME%\Desktop\"`, if there is the gpusim folder on your desktop.

6. Let `%RPATH%` be the absolute path of your R executable (either x86 or x64). Then type `%RPATH% CMD INSTALL --build gpusim` to run the compilation. Using the default R installation, `%RPATH%` will be `"C:\Program Files\R\R-2.15.0\bin\i386\R"` for x86 or `"C:\Program Files\R\R-2.15.0\bin\x64\R"` for x64 R versions. Again, notice that the quotation marks are necessary if the path contains space characters. You could also simply use `R`, but you have to know, whether this is the x64 or x86 executable.

7. A gpusim .zip file should now have been created.

The procedure above only creates a package binary for either x64 or x86 machines.