Stockholm University

Department of Physics

Using **GADGET-2** on '**Sunrise**'

Application Note for GADGET-2, v2.0.7

GAlaxies with **D**ark matter and **G**asint**E**rac**T**
A code for collisionless and gasdynamical cosmological simulations

http://www.mpa-garching.mpg.de/gadget

Mikica Kocic

Document Revision: v0.2 (2019-10-10)
Fysikum, Stockholm University

Contents

1 Introduction

This document contains usage notes for the GADGET-2 application on the 'Sunrise' HPC cluster. Besides a standard ANSI C compiler, GADGET-2 v2.0.7 (from 2005) requires specific versions of MPI, GSL, FFTW, HDF5 libraries. Such a software environment is available on Sunrise by using:

```
module load gnu8/8.3.0 gsl/2.5 hdf5/1.6.10 fftw/2.1.5
```

The required (older) versions of `hdf5` and `fftw` modules are built in Appendix A. To verify the loaded modules, issue `module list`.

2 Building Gadget-2

2.1 Obtain the Gadget-2 tarball

Download and unpack the GADGET-2 tarball:

```
wget https://wwwmpa.mpa-garching.mpg.de/gadget/gadget-2.0.7.tar.gz
tar xvfz gadget-2.0.7.tar.gz

cd Gadget-2.0.7/Gadget2
```

2.2 Modify Makefile

First, edit `Makefile` and **disable** all compile-time options in `Makefile` by commenting the lines with "`OPT +=`". These options will be dealt with in §2.3. Then, add the "Sunrise" cluster to Makefile:

```
SYSTYPE="Sunrise"

ifeq ($(SYSTYPE),"Sunrise")
CC        =  mpicc
OPTIMIZE  =  -O3 -Wall -m64
GSL_INCL  =  -I${GSL_INC}
GSL_LIBS  =  -L${GSL_LIB}
FFTW_INCL =  -I${FFTW_INC}
FFTW_LIBS =  -L${FFTW_LIB}
MPICHLIB  =
HDF5INCL  =  -I${HDF5_INC}
HDF5LIB   =  -L${HDF5_LIB} -lhdf5
endif
```

Finally, fix the following lines that references `FFTW_LIB`:

```
ifeq (NOTYPEPREFIX_FFTW,$(findstring NOTYPEPREFIX_FFTW,$(OPT))) # fftw installed with type prefix?
  FFTW_LIBS += -lrfftw_mpi -lfftw_mpi -lrfftw -lfftw
else
ifeq (DOUBLEPRECISION_FFTW,$(findstring DOUBLEPRECISION_FFTW,$(OPT)))
  FFTW_LIBS += -ldrfftw_mpi -ldfftw_mpi -ldrfftw -ldfftw
else
  FFTW_LIBS += -lsrfftw_mpi -lsfftw_mpi -lsrfftw -lsfftw
endif
endif

LIBS = $(HDF5LIB) -g  $(MPICHLIB)  $(GSL_LIBS) -lgsl -lgslcblas -lm $(FFTW_LIBS)
```

2.3 Makefile options for the examples

GADGET-2 comes with a number of examples. In order to run GADGET-2, one needs to do both compile-time changes in the Makefile and the run-time changes in the param file for the dataset. Here we create include files for Make which contain the compile-time options for all the examples.

```
# Makefile customization for the "cluster" example:

cat > Makefile-cluster.mk <<'EOF'
OPT += -DUNEQUALSOFTENINGS
OPT += -DPEANOHILBERT
OPT += -DWALLCLOCK
OPT += -DPMGRID=128
OPT += -DSYNCHRONIZATION
OPT += -DHAVE_HDF5
EOF

# Makefile customization for the "galaxy" example:

cat > Makefile-galaxy.mk <<'EOF'
OPT += -DUNEQUALSOFTENINGS
OPT += -DPEANOHILBERT
OPT += -DWALLCLOCK
OPT += -DSYNCHRONIZATION
OPT += -DHAVE_HDF5
EOF

# Makefile customization for the "gassphere" example:

cat > Makefile-gassphere.mk <<'EOF'
OPT += -DPEANOHILBERT
OPT += -DWALLCLOCK
OPT += -DSYNCHRONIZATION
OPT   += -DHAVE_HDF5
EOF

# Makefile customization for the "lcdm_gas" example:

cat > Makefile-lcdm_gas.mk <<'EOF'
OPT += -DPERIODIC
OPT += -DPEANOHILBERT
OPT += -DWALLCLOCK
OPT += -DPMGRID=128
OPT += -DSYNCHRONIZATION
OPT += -DHAVE_HDF5LI
EOF
```

2.4 Compile the code

Add line at the top of `Makefile` to include the specific compile-time options. For example:

```
SIM = galaxy
include Makefile-$(SIM).mk
```

Modify the line with "`EXEC =`" so the executable is put in the "`Sim`" directory with `$(SIM)` suffix:

```
EXEC = ../Sim/Gadget2-$(SIM)
```

Before compiling, load the necessary modules (needed only once after login):

```
module load gsl hdf5/1.6.10 fftw/2.1.5
```

To verify the modules environment, issue `module list`:

```
Currently Loaded Modules:
  1) autotools   3) gnu8/8.3.0       5) ohpc      7) hdf5/1.6.10
  2) prun/1.3    4) openmpi3/3.1.4   6) gsl/2.5   8) fftw/2.1.5
```

Finally, compile GADGET-2:

```
make clean
make
```

In the above example you will get "`../Sim/Gadget2-galaxy`". The following example will create an executable "`../Sim/Gadget2-`cluster" using the options from "`Makefile-cluster.mk`":

```
make clean
make SIM=cluster
```

To rebuild executables for all the examples, run the script "`build-all`":

```
#!/bin/bash

for sim in galaxy cluster gassphere lcdm_gas test_large ; do
        make clean
    make SIM=$sim
done

make clean
```

3 Build N-GenIC

Download and unpack the N-GenIC tarball:

```
wget https://wwwmpa.mpa-garching.mpg.de/gadget/n-genic.tar.gz
tar xvfz n-genic.tar.gz

cd N-GenIC
```

Edit `Makefile`, define the "Sunrise" cluster (this is the same as for GADGET2):

```
SYSTYPE="Sunrise"

ifeq ($(SYSTYPE),"Sunrise")
CC        =  mpicc
OPTIMIZE  =  -O3 -Wall -m64
GSL_INCL  =  -I${GSL_INC}
GSL_LIBS  =  -L${GSL_LIB}
FFTW_INCL =  -I${FFTW_INC}
FFTW_LIBS =  -L${FFTW_LIB}
MPICHLIB  =
HDF5INCL  =  -I${HDF5_INC}
HDF5LIB   =  -L${HDF5_LIB} -lhdf5
endif
```

Edit `Makefile`, fix the following lines that references `FFTW_LIB`. Ensure to have:

```
FFTW_LIBS += -ldrfftw_mpi -ldfftw_mpi -ldrfftw -ldfftw

LIBS = -lm  $(MPICHLIB)  $(FFTW_LIBS)  $(GSL_LIBS)  -lgsl -lgslcblas
```

Be sure to have loaded necessary modules (needed only once after login):

```
module load gsl hdf5/1.6.10 fftw/2.1.5
```

Finally, to build `N-GenIC`, issue the `make` command.

4 Running Gadget-2

4.1 Setup the simulation

Here we run a simulation for the "galaxy" example. First, create a directory for the simulations and the output subdirectory (here we assume that the current directory is `Gadget-2.0.7`):

```
# Assume we are in "Gadget-2.0.7"
mkdir -p Sim
cd Sim
```

Copy the default parameter file `galaxy.param`:

```
cp ../Gadget2/parameterfiles/galaxy.param .
```

then customize:

```
InitCondFile        ../../ICs/galaxy_littleendian.dat
OutputDir           .
OutputListFilename  ../outputs_galaxy.txt
SnapFormat          3   % = HDF5
```

Create `job.sh` script. The following script is suitable to run any simulation, which can be customized by setting the job name using `-J` option:

```
cat > job.sh <<'EOF'
#!/bin/bash -l

#SBATCH -J galaxy
#SBATCH -A Fysikum
#SBATCH -p fermi
#SBATCH -n 32
#SBATCH -t 24:00:00
#SBATCH -o %x-%j.log

################################################################################
# Load necessary modules

module load gsl hdf5/1.6.10 fftw/2.1.5

################################################################################
# Print the separation line

for i in {1..80}; do echo -n '-'; done; echo

################################################################################
# Report the cluster name and the used compute nodes"

echo "# Job running on '${SLURM_CLUSTER_NAME}' nodes: ${SLURM_NODELIST}"

################################################################################
# Print the separation line

for i in {1..80}; do echo -n '-'; done; echo; echo

################################################################################
# Show (this) script

cat $0
```

```
##############################################################################
# Print the separation line

echo; for i in {1..80}; do echo -n '-'; done; echo; echo

##############################################################################
# Show used modules

module list

##############################################################################
# Start measuring elapsed time

startT=$(date +'%s')

##############################################################################
# Print the separation line

for i in {1..80}; do echo -n '-'; done; echo; echo

##############################################################################
# Create a directory for the results and make it the current directory

mkdir ${SLURM_JOB_NAME}-${SLURM_JOB_ID}
cd ${SLURM_JOB_NAME}-${SLURM_JOB_ID}

##############################################################################
# Run the simulation

prun ../Gadget2-${SLURM_JOB_NAME} ../${SLURM_JOB_NAME}.param

##############################################################################
# Print the separation line

echo; for i in {1..80}; do echo -n '-'; done; echo; echo

##############################################################################
# Report the elapsed time

echo -e "\nElapsed:  $(($(date +'%s') - $startT)) s"

EOF
```

For example, to run a "cluster" simulation on "solar" partition using 16 cores, modify:

```
#SBATCH -J cluster
#SBATCH -p solar
#SBATCH -n 16
```

4.2 Start the simulation

To start the simulation, issue:

```
sbatch job.sh
```

You can use either `squeue` or `qtop` utility to monitor the job progress:

```
squeue -u mikica             TOTAL: 1 job(s)            sol-login   2019-10-05 22:47:55

   JOBID USER     ACCOUNT           NAME  ST REASON          TIME  TIME_LEFT NODES CPUS
     144 mikica   Fysikum         galaxy   R None            3:53   23:56:07     4   48
```

4.3 Interrupt the simulation

To interrupt the simulation, generate a file named `stop` in the output-directory of a simulation:

```
echo > galaxy-144/stop
```

The code will then write a restart-file and terminate itself after the current timestep is completed, and the file 'stop' will be erased automatically. Restart files are also generated when the last timestep of a simulation has been completed. They can be used if one later wants to extend the simulation beyond the original final time.

4.4 Simulation results

The GADGET-2 simulation snapshots can be visualized using Gadget Viewer (whose static build can be found in `/opt/ohpc/pub/bin/gadgetviewer`). An example of the "galaxy" simulation snapshots is shown in Figure 1.

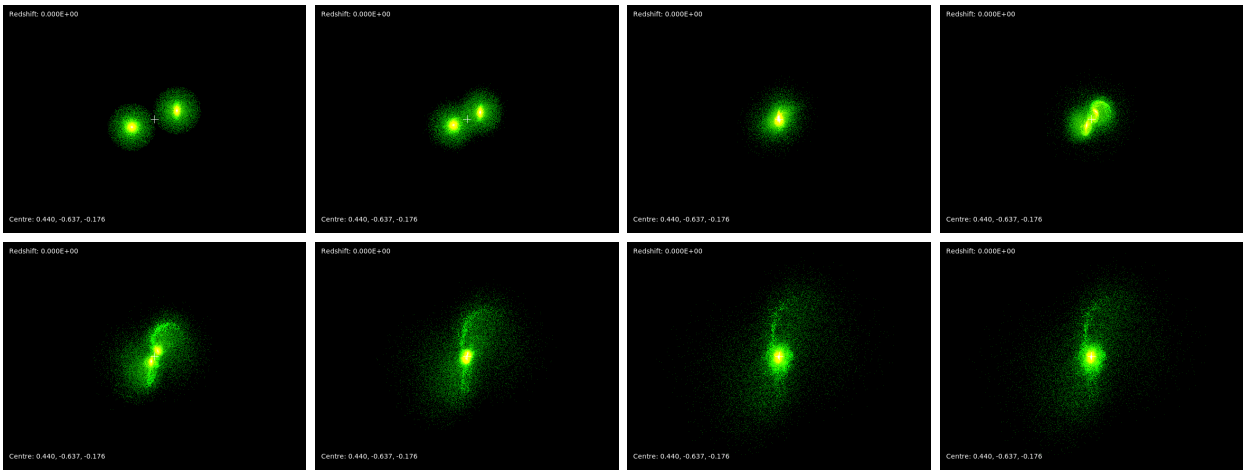

Figure 1: An example of the "galaxy" simulation snapshots.

Appendices

A Software Environment

A.1 Build `hdf5/1.6.10` module (done by the system administrator)

Get the software:

```
cd ~ohpc/tarballs
wget https://support.hdfgroup.org/ftp/HDF5/releases/hdf5-1.6/hdf5-1.6.10/src/hdf5-1.6.10.tar.bz2
cd ~ohpc/build
tar xvfj ~ohpc/tarballs/hdf5-1.6.10.tar.bz2
```

Configure, make, and install the library:

```
cd hdf5-1.60.10
./configure --prefix=/opt/ohpc/pub/libs/gnu8/hdf5/1.6.10
time make  # 3m30s
sudo make install
```

Create the module:

```
cat > /opt/ohpc/pub/moduledeps/gnu8/hdf5/1.6.10 <<'EOF'
#%Module1.0#####################################################################

proc ModulesHelp { } {
  puts stderr " "
  puts stderr "This module loads the HDF5 library built with the gnu8 compiler toolchain."
  puts stderr "\nVersion 1.6.10\n"
}
module-whatis "Name: HDF5 built with gnu8 toolchain"
module-whatis "Version: 1.6.10"
module-whatis "Category: runtime library"
module-whatis "Description: A general purpose library and file format for storing scientific data"
module-whatis "http://www.hdfgroup.org/HDF5"

set     version                     1.6.10

prepend-path    PATH                /opt/ohpc/pub/libs/gnu8/hdf5/1.6.10/bin
prepend-path    INCLUDE             /opt/ohpc/pub/libs/gnu8/hdf5/1.6.10/include
prepend-path    LD_LIBRARY_PATH     /opt/ohpc/pub/libs/gnu8/hdf5/1.6.10/lib

setenv          HDF5_DIR        /opt/ohpc/pub/libs/gnu8/hdf5/1.6.10
setenv          HDF5_BIN        /opt/ohpc/pub/libs/gnu8/hdf5/1.6.10/bin
setenv          HDF5_LIB        /opt/ohpc/pub/libs/gnu8/hdf5/1.6.10/lib
setenv          HDF5_INC        /opt/ohpc/pub/libs/gnu8/hdf5/1.6.10/include

family "hdf5"
EOF
```

Create the module version info:

```
cat > /opt/ohpc/pub/moduledeps/gnu8/hdf5/.version.1.6.10 <<'EOF'
#%Module1.0#####################################################################
##
## version file for hdf5-1.6.10
##
set     ModulesVersion      "1.6.10"
EOF
```

A.2 Build `fftw/2.1.5` module (done by the system administrator)

Get the software:

```
cd ~ohpc/tarballs;  wget http://www.fftw.org/fftw-2.1.5.tar.gz
cd ~ohpc/build;     tar xvfz ~ohpc/tarballs/fftw-2.1.5.tar.gz
```

Configure, make, and install the double precision library.

```
cd fftw-2.1.5
./configure --prefix=/opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5  \
            --enable-mpi --enable-type-prefix
time make  # 0m56s
sudo make install
```

Configure, make, and install the single precision library.

```
make distclean
./configure --prefix=/opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5  \
            --enable-mpi --enable-type-prefix --enable-float
time make  # 0m56s
sudo make install
```

Create the module:

```
cat > /opt/ohpc/pub/moduledeps/gnu8-openmpi3/fftw/2.1.5 <<'EOF'
#%Module1.0#####################################################################

proc ModulesHelp { } {
  puts stderr " "
  puts stderr "This module loads the fftw library built with the gnu8 compiler"
  puts stderr "toolchain and the openmpi3 MPI stack."
  puts stderr "\nVersion 3.3.8\n"
}
module-whatis "Name: fftw built with gnu8 compiler and openmpi3 MPI"
module-whatis "Version: 2.1.5"
module-whatis "Category: runtime library"
module-whatis "Description: A Fast Fourier Transform library"
module-whatis "URL http://www.fftw.org"

set     version                     2.1.5

prepend-path    PATH                /opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5/bin
prepend-path    MANPATH             /opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5/share/man
prepend-path    INCLUDE             /opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5/include
prepend-path    LD_LIBRARY_PATH     /opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5/lib

setenv          FFTW_DIR        /opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5
setenv          FFTW_LIB        /opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5/lib
setenv          FFTW_INC        /opt/ohpc/pub/libs/gnu8/openmpi3/fftw/2.1.5/include
EOF
```

Create the module version info:

```
cat > /opt/ohpc/pub/moduledeps/gnu8-openmpi3/fftw/.version.2.1.5 <<'EOF'
#%Module1.0#####################################################################
##
## version file for fftw-2.1.5
##
set     ModulesVersion      "2.1.5"
EOF
```