University of Texas at El Paso
Computational Science (CPS)

A short tutorial

Install petsc-3.5.3 on my Desktop
Show how to use PETSC on my Desktop and STAMPEDE

1 References

```
http://www.mcs.anl.gov/petsc/index.html
http://www.mcs.anl.gov/petsc/index.html  -> Features
http://www.mcs.anl.gov/petsc/documentation/installation.html
http://acts.nersc.gov/petsc/
http://hpc.ucla.edu/hoffman2/software/petsc.php#cpp
http://charlesmartinreid.com/wiki/Petsc
http://parallel-programming-quickstart.blogspot.com/2007/10/installing-petsc.html
http://www.cise.ufl.edu/research/sparse/codes/
```

2 Introduction

PETSC stand for **P**ortable, **E**xtensible **T**oolkit for **S**cientific **C**omputation. it is pronounced PET-see, the S is silent. PETSC library is designed to work in both parallel and sequential codes, it is object orientated in design and is available for Linux (or Unix) and Windows. It consists of both data structures and functions that are intended for building scientific applications. **C, C++, Fortran** and **Python** are supported. In fact, PETSc includes a large suite of parallel linear, nonlinear equation solvers and ODE integrators that are easily used in application codes written in **C, C++, Fortran** and now **Python**. Also, PETSc is a suite of data structures and routines for the scalable (parallel) solution of scientific applications modeled by partial differential equations. It supports MPI, shared memory pthreads, and GPUs through CUDA or OpenCL, as well as hybrid MPI-shared memory pthreads or MPI-GPU parallelism.

3 PETSC Important Features

PETSc has built in profiling for both memory usage and floating point calculation this accompanied with progress reporting features common in many of the solvers mean that one can get a reasonable picture of a codes execution profile merely by supplying a few additional command line arguments.

One of the strengths of the PETSc library is the large number of code examples it ships with. These examples are cross-referenced throughout the documentation so one can readily see an example of most significant functions along with related functions.

PETScs key feature is the number of highly regarded solvers etc. it brings under one roof: parallel timestepping ODE solvers, parallel preconditioners, Krylov subspace methods. parallel Newton-based nonlinear solvers and interfaces to numerous other 3rd party packages.

4 Features include

- Parallel vectors
  - includes code for communicating ghost points
- Parallel matrices
  - several sparse storage formats
  - easy, efficient assembly

- Scalable parallel preconditioners
- Krylov subspace methods
- Parallel Newton-based nonlinear solvers
- Parallel timestepping (ODE) solvers
- Support for Nvidia GPU cards
- Complete documentation
- Automatic profiling of floating point and memory usage
- Consistent user interface
- Intensive error checking
- Portable to UNIX and Windows
- Over one hundred examples
- PETSc is supported and will be actively enhanced for many years

5 Find out if PETSC is installed already on your PC

You mat install already petsc on your PC using `Synaptic Package Manager` or `sudo apt-get install petsc`. If this is the case, you may want to know where is your PC petsc folder

```
>> echo petcs
petcs

 >>  whereis petsc
petsc: /usr/lib/petsc /usr/include/petsc

 >> dpkg -l | grep petsc
ii  libpetsc3.1                 3.1.dfsg-11ubuntu1               Shared libraries for version 3.1 of PETSc
ii  libpetsc3.1-dbg             3.1.dfsg-11ubuntu1               Static debugging libraries for PETSc
ii  libpetsc3.1-dev             3.1.dfsg-11ubuntu1               Static libraries, shared links, header files for PETSc
ii  petsc-dev                   3.1.dfsg-11ubuntu1               Meta-package depending on latest PETSc development package
ii  petsc3.1-doc                3.1.dfsg-11ubuntu1               Documentation and examples for PETSc
```

6 Install directly using apt-get or Synaptic Package Manager

There many ways to install PETSC. Here are two direct way, NO RECOMMENDED. Since PETSC requiere requiere to be configure to full fill your programming needs. These two way will not help to acomplish that.

- Open a terminal

  ```
  >> sudo apt-get install petsc-dev
  ```

- Open Synaptic Package Manager, see figure (**??**)

7 Get PETSC on the follow wedsite

Open a terminal and create a folder where do you want to download PETSC. For example

```
>> cd Desktop/
>> mkdir PETSC
>> cd PETSC
```

Donwload petsc into the folder PETSC

```
henry@bluebottle:~/Desktop/PETSC$ wget http://ftp.mcs.anl.gov/pub/petsc/release-snapshots/petsc-3.5.3.tar.gz
```

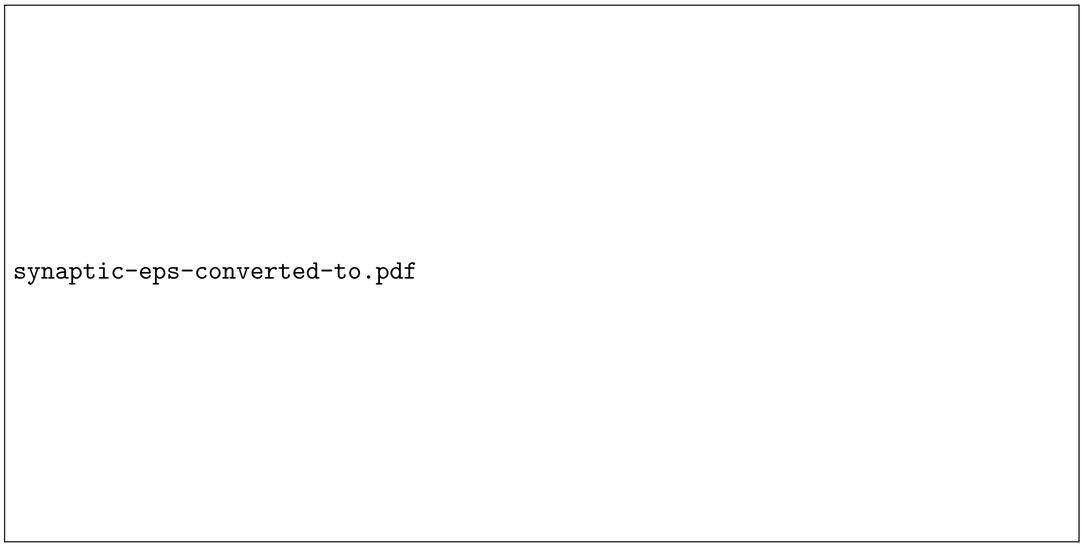


Figure 1: Synaptic Package Manager

Unpack PETSC in one step

```
henry@bluebottle:~/Desktop/PETSC$ gunzip -c petsc-3.5.3.tar.gz | tar -xof -
```

or

```
henry@bluebottle:~/Desktop/PETSC$ tar zxvf petsc-3.5.3.tar.gz
```

Unpack into two steps

```
henry@bluebottle:~/Desktop/PETSC$ gunzip petsc-3.5.3.tar.gz
henry@bluebottle:~/Desktop/PETSC$ tar xvf petsc-3.5.3.tar
henry@bluebottle:~/Desktop/PETSC$ cd petsc-3.5.3
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$
```

8 Get Examples for PETSC

You use these examples to check if your PETSC installation and learn how to programming PETSC

```
>> wget http://www.mcs.anl.gov/petsc/petsc-3.4/src/ksp/ksp/examples/tutorials/
```

9 PETSC_DIR and PETSC_ARCH

Before run PETSc based program you must set two environment variables.

- Set PETSC_DIR, first environment variable to the path of the PETSc directory. Within this directory there is a lib directory which will have at least one subdirectory corresponding to a set of PETSc libraries built with a given configuration.

  ```
  export PETSC_DIR=~/Desktop/PETSC/petsc-3.5.3
  ```

  Set PETSC_ARCH, second environment variable is used to specify which library build within the PETSC_DIR to use. This allows you to prepare a variety of PETSc builds e.g. optimised, debug differing MPI libraries etc. and create and run the corresponding executables while only changing the PETSC_ARCH variable.

  ```
  export PETSC_ARCH=linux-gnu-complex
  ```

- `PETSC_DIR` and `PETSC_ARCH` are a couple of variables that control the configuration and build process of PETSc. These variables can be set as environment variables or specified on the command line `[to both configure and make]`.

- `PETSC_DIR`: this variable should point to the location of the PETSc installation that is used. Multiple PETSc versions can coexist on the same file-system. By changing `PETSC_DIR` value, one can switch between these installed versions of PETSc.
- `PETSC_ARCH`: this variable gives a name to a `configuration/build`. Configure uses this value to stores the generated config makefiles in `${PETSC_DIR}/${PETSC_ARCH}/conf`. And make uses this value to determine this location of these makefiles [which intern help in locating the correct include and library files].
- Thus one can install multiple variants of PETSc libraries - by providing different `PETSC_ARCH` values to each configure build. Then one can switch between using these variants of libraries [from make] by switching the `PETSC_ARCH` value used.
- If configure doesn't find a `PETSC_ARCH` value [either in env variable or command line option], it automatically generates a default value and uses it. Also - if make doesn't find a `PETSC_ARCH` env variable - it defaults to the value used by last successful invocation of previous configure.
- Build Complex version of PETSc [using c++ compiler] (add the option `--with-fortran-kernels=generic` to get possibly faster complex number performance on some systems):

  ```
  henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$./configure --with-cc=gcc --with-fc=gfortran --with-cxx=g++ --with-clanguage=cxx --download-fblaslapack --download-mpich --with-scalar-type=complex
  ```

  Note that `--with-clanguage=cxx` means that the PETSc source code is compiled with the C++ compiler. This is not normally needed and we don't recommend it. One can use 'c' build of PETSc from both C and C++. One can also have a complex build with C99.
- Install 2 variants of PETSc. Specify different `PETSC_ARCH` for each build.
  - With gnu

    ```
    henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$./configure PETSC_ARCH=linux-gnu --with-cc=gcc --with-cxx=g++ --with-fc=gfortran --download-mpich
    henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$make PETSC_ARCH=linux-gnu all test
    ```
  - With intel compilers (intel use mkl instead of blas and lapack).

    ```
    henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$./configure PETSC_ARCH=linux-gnu-intel --with-cc=icc --with-cxx=icpc --with-fc=ifort --download-mpich --with-blas-lapack-dir=/usr/local/mkl
    henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$make PETSC_ARCH=linux-gnu-intel all test
    ```
  - BLAS/LAPACK : These packages provide some basic numeric kernels used by PETSc.
    - Configure will automatically look for `blas/lapack` in certain standard locations, on most systems you should not need to provide any information about `BLAS/LAPACK` in the `./configure` command.
    - One can use the following options to let configure download/install blas/lapack automatically.
      - `--download-fblaslapack` [when fortran compiler is present]
      - `--download-f2cblaslapack` [when configuring without a fortran compiler - i.e `--with-fc=0`]
    - Alternatively one can use other options like one of the following.
      - `--with-blas-lapack-lib=libsunperf.a`
      - `--with-blas-lib=libblas.a --with-lapack-lib=liblapack.a`
      - `--with-blas-lapack-dir=/soft/com/packages/intel/13/079/mkl`
- Specify enviornment variable for bash `[can be specified in ~/.bashrc]`

  ```
  export PETSC_DIR=~/Desktop/PETSC/petsc-3.5.3
  export PETSC_ARCH=linux-gnu-c-debug
  ```

10 Find out if compiler names are within the $PATH

If this is the case. No need for explicit specification on the configuration.

```
henry@bluebottle:~/Desktop$ which gcc
/usr/bin/gcc
henry@bluebottle:~/Desktop$ which g++
/usr/bin/g++
henry@bluebottle:~/Desktop$ which gfortran
/usr/bin/gfortran
henry@bluebottle:~/Desktop$ which mpicc
/usr/bin/mpicc
henry@bluebottle:~/Desktop$ which mpicxx
/usr/bin/mpicxx
henry@bluebottle:~/Desktop$ which mpif90
/usr/bin/mpif90
```

11 FFTW

`--download-PACKAGENAME=/PATH/TO/package.tar.gz`: If `./configure` cannot automatically download the package [due to network/firewall issues], one can download the package by alternaive means [perhaps wget or scp via some other machine]. Once the tarfile is downloaded, the path to this file can be specified to configure with this option. Configure will proceed to install this package and then configure PETSc with it.

Since FFTW need to configure for a MPI. We download fftw and set the $PATH to be install it.

```
--download-fftw=/home/henry/Desktop/FFTW/downloads_fftw/fftw-3.5.3.tar.gz
```

12 Install PETSC Manually

12.1 Before you start your installation

You made want to know how your petsc folder look like because it is going to change a little.

```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ ll
total 8756
drwxr-xr-x 12 henry henry    4096 May 23 17:42 ./
drwxrwxr-x  7 henry henry    4096 Aug 10 10:57 ../
drwxr-xr-x  6 henry henry    4096 May 23 17:42 bin/
drwxr-xr-x  2 henry henry    4096 May 23 17:42 conf/
drwxr-xr-x  5 henry henry    4096 May 23 17:42 config/
-rwxr-xr-x  1 henry henry     340 Sep  8  2014 configure*
-rw-r--r--  1 henry henry    1751 Sep  8  2014 CONTRIBUTING
-rw-r--r--  1 henry henry 6336652 May 23 17:42 CTAGS
-rw-r--r--  1 henry henry    6844 Sep  8  2014 .dir-locals.el
drwxr-xr-x  4 henry henry    4096 May 23 17:42 docs/
-rw-r--r--  1 henry henry    8798 May 23 10:57 gmakefile
drwxr-xr-x  6 henry henry    4096 May 23 17:42 include/
-rw-r--r--  1 henry henry     815 May 23 17:42 index.html
drwxr-xr-x  3 henry henry    4096 May 23 17:42 interfaces/
-rw-r--r--  1 henry henry    1526 Sep  8  2014 LICENSE
-rw-r--r--  1 henry henry   27891 May 23 17:42 makefile
-rw-r--r--  1 henry henry   34168 May 23 17:42 makefile.html
-rwxr-xr-x  1 henry henry    9775 Jan 30  2015 setup.py*
drwxr-xr-x  3 henry henry    4096 May 13  2013 share/
drwxr-xr-x 12 henry henry    4096 May 23 17:42 src/
drwxr-xr-x  3 henry henry    4096 May 13  2013 systems/
-rw-r--r--  1 henry henry 2458233 May 23 17:42 TAGS
drwxr-xr-x  3 henry henry    4096 May 23 17:42 tutorials/
```

Let's start with the installation

```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ export PETSC_DIR=~/Desktop/PETSC/petsc-3.5.3

henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ export PETSC_ARCH=linux-gnu-complex

henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ ./configure
===============================================================================
             Configuring PETSc to compile on your system
===============================================================================
TESTING: alternateConfigureLibrary from PETSc.packages.mpi4py(config/PETSc/packages/mpi4py.py:56)
Compilers:
  C Compiler:         mpicc  -fPIC -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragmas -g3 -O0
  C++ Compiler:       mpicxx  -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragmas -g -O0   -fPIC
  Fortran Compiler:   mpif90  -fPIC -Wall -Wno-unused-variable -ffree-line-length-0 -Wno-unused-dummy-argument -g -O0
```

```
Linkers:
  Shared linker:   mpicc  -shared  -fPIC -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragmas -g3 -O0
  Dynamic linker:   mpicc  -shared  -fPIC -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragmas -g3 -O0
make:
MPI:
  Includes: -I/usr/lib/openmpi/include -I/usr/lib/openmpi/include/openmpi
BLAS/LAPACK: -llapack -lblas
X:total 21312
drwxr-xr-x 14 henry henry     4096 Apr 27 13:43 ./
drwxrwxr-x  6 henry henry     4096 May  1 13:28 ../
drwxr-xr-x  6 henry henry     4096 Apr 24 10:40 bin/
-rw-------  1 henry henry    62332 Apr 27 13:40 CMakeLists.txt
drwxr-xr-x  2 henry henry     4096 Apr 24 10:40 conf/
drwxr-xr-x  5 henry henry     4096 Apr 24 10:40 config/
-rwxr-xr-x  1 henry henry      340 Sep  8  2014 configure*
lrwxrwxrwx  1 henry henry       28 Apr 27 13:40 configure.log -> linux-dbg/conf/configure.log
lrwxrwxrwx  1 henry henry       32 Apr 27 13:40 configure.log.bkp -> linux-dbg/conf/configure.log.bkp
-rw-rw-r--  1 henry henry  1963859 Apr 24 23:03 configure_log.txt
-rw-r--r--  1 henry henry     1751 Sep  8  2014 CONTRIBUTING
-rw-r--r--  1 henry henry  6330591 Jan 31 00:14 CTAGS
drwxrwxr-x  6 henry henry     4096 Apr 24 10:40 -dbg/
-rw-r--r--  1 henry henry     6844 Sep  8  2014 .dir-locals.el
drwxr-xr-x  4 henry henry     4096 Jan 31 00:14 docs/
-rw-r--r--  1 henry henry     8681 Sep  8  2014 gmakefile
drwxr-xr-x  6 henry henry     4096 Jan 31 00:14 include/
-rw-r--r--  1 henry henry      816 Jan 31 00:14 index.html
drwxr-xr-x  3 henry henry     4096 Jan 31 00:14 interfaces/
-rw-r--r--  1 henry henry     1526 Sep  8  2014 LICENSE
drwxrwxr-x  7 henry henry     4096 Apr 25 19:45 linux-dbg/
-rw-r--r--  1 henry henry    27795 Jan 31 00:14 makefile
-rw-r--r--  1 henry henry    34073 Jan 31 00:14 makefile.html
lrwxrwxrwx  1 henry henry       23 Apr 27 13:40 make.log -> linux-dbg/conf/make.log
-rw-rw-r--  1 henry henry        0 Apr 28 18:52 .nagged
-rw-rw-r--  1 henry henry 10825568 Apr 27 13:40 RDict.log
-rwxr-xr-x  1 henry henry     9775 Jan 30 23:23 setup.py*
drwxr-xr-x  3 henry henry     4096 May 13  2013 share/
drwxr-xr-x 12 henry henry     4096 Jan 31 00:14 src/
drwxr-xr-x  3 henry henry     4096 May 13  2013 systems/
-rw-r--r--  1 henry henry  2454792 Jan 31 00:14 TAGS
drwxr-xr-x  3 henry henry     4096 Jan 31 00:14 tutorials/

  Library:  -lX11
  Arch:
pthread:
  Library:  -lpthread
ssl:
  Library:  -lssl -lcrypto
valgrind:
PETSc:
  PETSC_ARCH: linux-dbg
  PETSC_DIR: /home/henry/Desktop/PETSC/petsc-3.5.3
  Clanguage: C
  shared libraries: enabled
  Scalar type: real
  Precision: double
  Memory alignment: 16
xxx=========================================================================xxx
 Configure stage complete. Now build PETSc libraries with (gnumake build):
   make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg all
xxx=========================================================================xxx
```

Creat the object files

```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ make
.
.
.
          CC linux-dbg/obj/src/tao/interface/taosolver_bounds.o
          CC linux-dbg/obj/src/tao/interface/dlregistao.o
          CC linux-dbg/obj/src/tao/interface/fdiff.o
          CC linux-dbg/obj/src/tao/interface/fdtest.o
          CC linux-dbg/obj/src/tao/interface/ftn-auto/taosolver_boundsf.o
          CC linux-dbg/obj/src/tao/interface/ftn-auto/taosolver_fgf.o
          CC linux-dbg/obj/src/tao/interface/ftn-auto/taosolver_hjf.o
          CC linux-dbg/obj/src/tao/interface/ftn-auto/taosolverf.o
          CC linux-dbg/obj/src/tao/interface/ftn-custom/ztaosolverf.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/nls/nls.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/neldermead/neldermead.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/ntr/ntr.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/cg/taocg.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/lmvm/lmvm.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/bmrm/bmrm.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/ntl/ntl.o
          CC linux-dbg/obj/src/tao/unconstrained/impls/owlqn/owlqn.o
          CC linux-dbg/obj/src/tao/constrained/impls/ipm/ipm.o
          CC linux-dbg/obj/src/tao/linesearch/interface/taolinesearch.o
          CC linux-dbg/obj/src/tao/linesearch/interface/dlregis_taolinesearch.o
          CC linux-dbg/obj/src/tao/linesearch/interface/ftn-auto/taolinesearchf.o
          CC linux-dbg/obj/src/tao/linesearch/interface/ftn-custom/ztaolinesearchf.o
          CC linux-dbg/obj/src/tao/linesearch/impls/armijo/armijo.o
          CC linux-dbg/obj/src/tao/linesearch/impls/morethuente/morethuente.o
          CC linux-dbg/obj/src/tao/linesearch/impls/owarmijo/owarmijo.o
          CC linux-dbg/obj/src/tao/linesearch/impls/unit/unit.o
          CC linux-dbg/obj/src/tao/linesearch/impls/gpcglinesearch/gpcglinesearch.o
          CC linux-dbg/obj/src/tao/leastsquares/impls/pounders/pounders.o
          CC linux-dbg/obj/src/tao/leastsquares/impls/pounders/gqt.o
     CLINKER /home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg/lib/libpetsc.so.3.5.3
make[2]: Leaving directory '/home/henry/Desktop/PETSC/petsc-3.5.3'
=========================================
make[1]: Leaving directory '/home/henry/Desktop/PETSC/petsc-3.5.3'
Now to check if the libraries are working do:
make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg test
=========================================
```

12.2 A new Folder is create

Folder `linux-dbg` is created

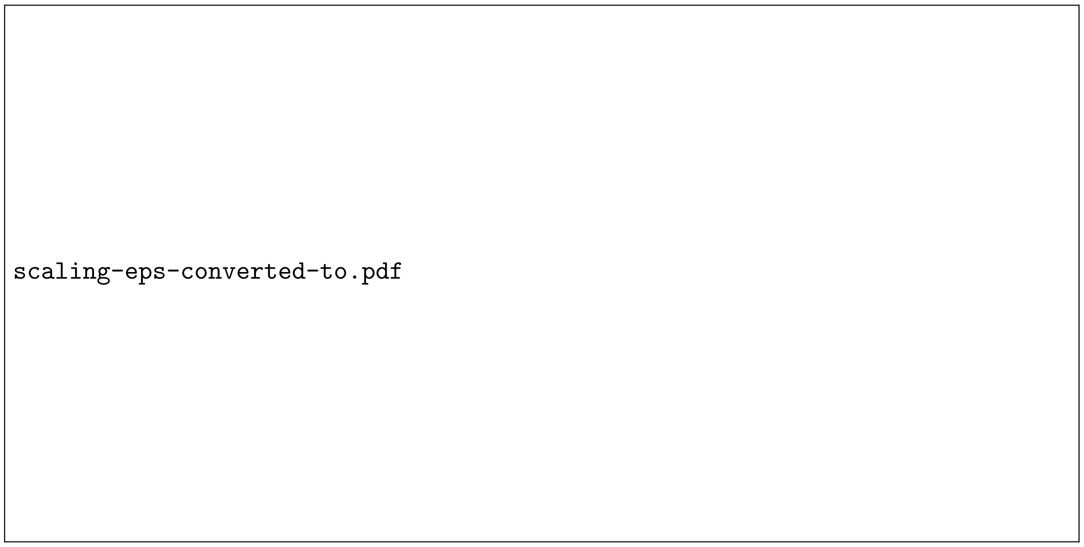
```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ ll
total 21312
drwxr-xr-x 14 henry henry     4096 Apr 27 13:40 ./
drwxrwxr-x  5 henry henry     4096 Apr 24 18:20 ../
drwxr-xr-x  6 henry henry     4096 Apr 24 10:40 bin/
-rw-------  1 henry henry    62332 Apr 27 13:40 CMakeLists.txt
drwxr-xr-x  2 henry henry     4096 Apr 24 10:40 conf/
drwxr-xr-x  5 henry henry     4096 Apr 24 10:40 config/
-rwxr-xr-x  1 henry henry      340 Sep  8  2014 configure*
lrwxrwxrwx  1 henry henry       28 Apr 27 13:40 configure.log -> linux-dbg/conf/configure.log
lrwxrwxrwx  1 henry henry       32 Apr 27 13:40 configure.log.bkp -> linux-dbg/conf/configure.log.bkp
-rw-rw-r--  1 henry henry  1963859 Apr 24 23:03 configure_log.txt
-rw-r--r--  1 henry henry     1751 Sep  8  2014 CONTRIBUTING
-rw-r--r--  1 henry henry  6330591 Jan 31 00:14 CTAGS
drwxrwxr-x  6 henry henry     4096 Apr 24 10:40 -dbg/
-rw-r--r--  1 henry henry     6844 Sep  8  2014 .dir-locals.el
drwxr-xr-x  4 henry henry     4096 Jan 31 00:14 docs/
-rw-r--r--  1 henry henry     8681 Sep  8  2014 gmakefile
drwxr-xr-x  6 henry henry     4096 Jan 31 00:14 include/
-rw-r--r--  1 henry henry      816 Jan 31 00:14 index.html
drwxr-xr-x  3 henry henry     4096 Jan 31 00:14 interfaces/
-rw-r--r--  1 henry henry     1526 Sep  8  2014 LICENSE
drwxrwxr-x  7 henry henry     4096 Apr 25 19:45 linux-dbg/
-rw-r--r--  1 henry henry    27795 Jan 31 00:14 makefile
-rw-r--r--  1 henry henry    34073 Jan 31 00:14 makefile.html
lrwxrwxrwx  1 henry henry       23 Apr 27 13:40 make.log -> linux-dbg/conf/make.log
-rw-rw-r--  1 henry henry        0 Apr 27 13:31 .nagged
-rw-rw-r--  1 henry henry 10825568 Apr 27 13:40 RDict.log
-rwxr-xr-x  1 henry henry     9775 Jan 30 23:23 setup.py*
drwxr-xr-x  3 henry henry     4096 May 13  2013 share/
drwxr-xr-x 12 henry henry     4096 Jan 31 00:14 src/
drwxr-xr-x  3 henry henry     4096 May 13  2013 systems/
-rw-r--r--  1 henry henry  2454792 Jan 31 00:14 TAGS
drwxr-xr-x  3 henry henry     4096 Jan 31 00:14 tutorials/
```

12.3 Now to check if the libraries are working

```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg test
Running test examples to verify correct installation
Using PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 and PETSC_ARCH=linux-dbg
C/C++ example src/snes/examples/tutorials/ex19 run successfully with 1 MPI process
C/C++ example src/snes/examples/tutorials/ex19 run successfully with 2 MPI processes
Fortran example src/snes/examples/tutorials/ex5f run successfully with 1 MPI process
Completed test examples
=========================================
Now to evaluate the computer systems you plan use - do:
make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg streams NPMAX=<number of MPI processes you intend to use>
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$
```

Now to evaluate the computer systems you plan use-do:

```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg streams NPMAX=4
cd src/benchmarks/streams; /usr/bin/make  --no-print-directory streams
mpicc -o MPIVersion.o -c -fPIC -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragmas -g3 -O0   -I/home/henry/Desktop/PETSC/petsc-3.5.3/include
/home/henry/Desktop/PETSC/petsc-3.5.3/src/benchmarks/streams/MPIVersion.c: In function main:
/home/henry/Desktop/PETSC/petsc-3.5.3/src/benchmarks/streams/MPIVersion.c:99:7: warning: value computed is not used [-Wunused-value]
/home/henry/Desktop/PETSC/petsc-3.5.3/src/benchmarks/streams/MPIVersion.c:103:4: warning: value computed is not used [-Wunused-value]
Number of MPI processes 1
Process 0 bluebottle
Function      Rate (MB/s)
Copy:       11785.8911
Scale:      11347.4576
Add:        13004.7537
Triad:      12298.5090
Number of MPI processes 2
Process 0 bluebottle
Process 1 bluebottle
Function      Rate (MB/s)
Copy:       12171.7356
Scale:      12082.4353
Add:        13361.2045
Triad:      13562.3699
Number of MPI processes 3
Process 0 bluebottle
Process 1 bluebottle
Process 2 bluebottle
Function      Rate (MB/s)
Copy:       12192.0058
Scale:      12130.3003
Add:        13479.2845
Triad:      13570.7497
Number of MPI processes 4
Process 0 bluebottle
Process 1 bluebottle
Process 2 bluebottle
Process 3 bluebottle
Function      Rate (MB/s)
Copy:       12592.1116
Scale:      12085.5584
Add:        13892.1390
Triad:      13938.5530
------------------------------------------------
np  speedup
1 1.0
2 1.1
3 1.1
4 1.13
Estimation of possible speedup of MPI programs based on Streams benchmark.
It appears you have 1 node(s)
See graph in the file src/benchmarks/streams/scaling.png
```



PetscScalar: Evaluate the Computer System using Real Numbers

13 Complex Configuration

```
henry@bluebottle:~/Desktop/PETSC$ cd petsc-3.5.3/
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ export PETSC_DIR=~/Desktop/PETSC/petsc-3.5.3
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ export PETSC_ARCH=linux-dbg-Complex
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ ./configure --with-cc=gcc --with-fc=gfortran --with-cxx=g++ --with-clanguage=cxx
--download-fblaslapack --download-mpich --with-scalar-type=complex

===============================================================================
             Configuring PETSc to compile on your system
===============================================================================
===============================================================================
                                      ===============================================================================
                                                                            ===============================================================================
  C Compiler:         /home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/bin/mpicc  -fPIC -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-p
  C++ Compiler:       /home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/bin/mpicxx   -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragm
  Fortran Compiler:   /home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/bin/mpif90  -fPIC  -Wall -Wno-unused-variable -ffree-line-length-0 -Wno-un
Linkers:
  Shared linker:   /home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/bin/mpicxx  -shared
  Dynamic linker:   /home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/bin/mpicxx  -shared
make:
MPI:
  Includes: -I/home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/include
BLAS/LAPACK: -Wl,-rpath,/home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/lib -L/home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/lib -lflap
fblaslapack:
X:
  Library:  -lX11
  Arch:
pthread:
  Library:  -lpthread
ssl:
  Library:  -lssl -lcrypto
valgrind:
PETSc:
  PETSC_ARCH: linux-dbg-Complex
  PETSC_DIR: /home/henry/Desktop/PETSC/petsc-3.5.3
  Clanguage: Cxx
  shared libraries: enabled
  Scalar type: complex
  Precision: double
  Memory alignment: 16
xxx=========================================================================xxx
 Configure stage complete. Now build PETSc libraries with (gnumake build):
   make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg-Complex all
xxx=========================================================================xxx
```

Creat the objects Files

```
 henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ make
.
.
.
         CXX linux-dbg-Complex/obj/src/ts/impls/implicit/theta/theta.o
         CXX linux-dbg-Complex/obj/src/ts/impls/implicit/theta/ftn-auto/thetaf.o
         CXX linux-dbg-Complex/obj/src/ts/impls/implicit/alpha/alpha.o
         CXX linux-dbg-Complex/obj/src/ts/impls/implicit/alpha/ftn-auto/alphaf.o
         CXX linux-dbg-Complex/obj/src/ts/interface/ts.o
         CXX linux-dbg-Complex/obj/src/ts/interface/tscreate.o
         CXX linux-dbg-Complex/obj/src/ts/interface/tsreg.o
         CXX linux-dbg-Complex/obj/src/ts/interface/tsregall.o
         CXX linux-dbg-Complex/obj/src/ts/interface/dlregists.o
         CXX linux-dbg-Complex/obj/src/ts/interface/tseig.o
         CXX linux-dbg-Complex/obj/src/ts/interface/ftn-auto/tsf.o
         CXX linux-dbg-Complex/obj/src/ts/interface/ftn-custom/ztscreatef.o
         CXX linux-dbg-Complex/obj/src/ts/interface/ftn-custom/ztsf.o
         CXX linux-dbg-Complex/obj/src/ts/interface/ftn-custom/ztsregf.o
         CXX linux-dbg-Complex/obj/src/ts/adapt/interface/tsadapt.o
         CXX linux-dbg-Complex/obj/src/ts/adapt/interface/ftn-auto/tsadaptf.o
         CXX linux-dbg-Complex/obj/src/ts/adapt/impls/cfl/adaptcfl.o
         CXX linux-dbg-Complex/obj/src/ts/adapt/impls/none/adaptnone.o
         CXX linux-dbg-Complex/obj/src/ts/adapt/impls/basic/adaptbasic.o
          FC linux-dbg-Complex/obj/src/ts/f90-mod/petsctsmod.o
         CXX linux-dbg-Complex/obj/src/tao/matrix/lmvmmat.o
         CXX linux-dbg-Complex/obj/src/tao/matrix/adamat.o
         CXX linux-dbg-Complex/obj/src/tao/matrix/submatfree.o
         CXX linux-dbg-Complex/obj/src/tao/util/tao_util.o
         CXX linux-dbg-Complex/obj/src/tao/util/ftn-auto/tao_utilf.o
         CXX linux-dbg-Complex/obj/src/tao/interface/taosolver.o
         CXX linux-dbg-Complex/obj/src/tao/interface/taosolver_fg.o
         CXX linux-dbg-Complex/obj/src/tao/interface/taosolverregi.o
         CXX linux-dbg-Complex/obj/src/tao/interface/taosolver_hj.o
         CXX linux-dbg-Complex/obj/src/tao/interface/taosolver_bounds.o
         CXX linux-dbg-Complex/obj/src/tao/interface/dlregistao.o
         CXX linux-dbg-Complex/obj/src/tao/interface/fdiff.o
         CXX linux-dbg-Complex/obj/src/tao/interface/fdtest.o
         CXX linux-dbg-Complex/obj/src/tao/interface/ftn-auto/taosolver_boundsf.o
         CXX linux-dbg-Complex/obj/src/tao/interface/ftn-auto/taosolver_fgf.o
         CXX linux-dbg-Complex/obj/src/tao/interface/ftn-auto/taosolver_hjf.o
         CXX linux-dbg-Complex/obj/src/tao/interface/ftn-auto/taosolverf.o
         CXX linux-dbg-Complex/obj/src/tao/interface/ftn-custom/ztaosolverf.o
         CXX linux-dbg-Complex/obj/src/tao/linesearch/interface/taolinesearch.o
         CXX linux-dbg-Complex/obj/src/tao/linesearch/interface/dlregis_taolinesearch.o
         CXX linux-dbg-Complex/obj/src/tao/linesearch/interface/ftn-auto/taolinesearchf.o
         CXX linux-dbg-Complex/obj/src/tao/linesearch/interface/ftn-custom/ztaolinesearchf.o
     CLINKER /home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/lib/libpetsc.so.3.5.3
make[2]: Leaving directory '/home/henry/Desktop/PETSC/petsc-3.5.3'
=========================================
make[1]: Leaving directory '/home/henry/Desktop/PETSC/petsc-3.5.3'
Now to check if the libraries are working do:
make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg-Complex test
=========================================
```

Check if the library are working

```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg-Complex test
Running test examples to verify correct installation
Using PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 and PETSC_ARCH=linux-dbg-Complex
C/C++ example src/snes/examples/tutorials/ex19 run successfully with 1 MPI process
C/C++ example src/snes/examples/tutorials/ex19 run successfully with 2 MPI processes
Fortran example src/snes/examples/tutorials/ex5f run successfully with 1 MPI process
Completed test examples
=========================================
Now to evaluate the computer systems you plan use - do:
make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg-Complex streams NPMAX=<number of MPI processes you intend to use>
```

Now we evaluate the computer system you plan to use:

```
henry@bluebottle:~/Desktop/PETSC/petsc-3.5.3$ make PETSC_DIR=/home/henry/Desktop/PETSC/petsc-3.5.3 PETSC_ARCH=linux-dbg-Complex streams NPMAX=4
cd src/benchmarks/streams; /usr/bin/make  --no-print-directory streams
/home/henry/Desktop/PETSC/petsc-3.5.3/linux-dbg-Complex/bin/mpicxx -o MPIVersion.o -c -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragmas -g
Number of MPI processes 1
Process 0 bluebottle
Function      Rate (MB/s)
Copy:       11076.8118
Scale:      10638.6912
Add:        12522.6468
Triad:      12336.1882
Number of MPI processes 2
Process 0 bluebottle
Process 1 bluebottle
Function      Rate (MB/s)
Copy:       12778.3848
Scale:      12660.8554
Add:        14196.9249
Triad:      14360.4703
Number of MPI processes 3
Process 0 bluebottle
Process 1 bluebottle
Process 2 bluebottle
Function      Rate (MB/s)
Copy:       12539.6763
Scale:      12434.2251
Add:        13975.9454
Triad:      14031.1417
Number of MPI processes 4
Process 0 bluebottle
Process 1 bluebottle
Process 2 bluebottle
Process 3 bluebottle
Function      Rate (MB/s)
Copy:       12412.4117
Scale:      12379.4968
Add:        13801.3317
Triad:      13867.5576
------------------------------------------------
np  speedup
1 1.0
2 1.16
3 1.14
4 1.12
Estimation of possible speedup of MPI programs based on Streams benchmark.
It appears you have 1 node(s)
See graph in the file src/benchmarks/streams/scaling.png
```

PetscScalar: Evaluate the Computer System using Complex Numbers

14 Use BASH file to untar, install and configure

```
#!/bin/bash
henry@bluebottle:~$ tar zxvf /Desktop/PETSC/petsc-3.5.3.tar.gz  # untar petsc on a particular folder
cd Desktop/PETSC/petsc-3.5.3                                    # move to petsc folder
export PETSC_DIR=~/Desktop/PETSC/petsc-3.5.3
export PETSC_ARCH=linux-gnu-complex
./configure PETSC_ARCH=linux-gnu-complex  --with-scalar-type=complex --download-fftw --with-debugging=1
make
make all test  # In one step:
```

In two steps:

```
make all
make test
```

On this configuration:

- `PETSC_ARCH=linux-gnu-complex` give a name to configuration/build
- Complex number configutationis using: `--with-scalar-type=complex`
- If BLAS, LAPACK, MPI are install already. The default system/compiler locations are availab via PATH. No need for these:

  ```
  --with-blas-lapack-dir=/usr/local/blaslapack
  --with-mpi-dir=/usr/local/mpich
  --with-cc=mpicc --with-cxx=mpicxx --with-fc=mpif90
  ```

- fftw is install but it is not setup for mpi. We hope the download of fftw will configure fftw for mpi, No need to include the PATH to fftw: `--with-fftw-dir=/usr/include/`

15 Compile and Execute Hello World Example

15.1 On my Desktop PC

My Hello world c code: `example_hello_0_C.c`

```
#include <petsc.h>

int main ( int argc, char *argv[] ){
   PetscErrorCode ierr;
   PetscMPIInt    rank, size;

   PetscInitialize(&argc, &argv, PETSC_NULL,PETSC_NULL);
   ierr = MPI_Comm_size(PETSC_COMM_WORLD,&size);
   CHKERRQ(ierr);  /* Checks error code, if non-zero it calls the error handler and then returns */
   ierr = MPI_Comm_rank(PETSC_COMM_WORLD,&rank);
   CHKERRQ(ierr);

/* Prints to standard out, only from the first processor in the communicator. Calls from other processes are ignored.
   Specifically designed to print the message once for all the processes */

   //PetscPrintf(PETSC_COMM_WORLD,"Number of processors = %d, rank = %d\n",size,rank);
   //PetscPrintf(PETSC_COMM_WORLD, "Hello World from [%d] rank\n",rank);

/* Prints to standard out, from all processor in the communicator. Specifically designed to print the message from each of the processes*/
   PetscPrintf(PETSC_COMM_SELF,"Hello World from [%d] rank\n",rank);
   PetscPrintf(PETSC_COMM_SELF,"Number of processors = %d, rank = %d\n",size,rank);

   PetscFinalize();
   return 0;
}
```

Makefile file : `Makefile`

```
CFLAGS          =
FFLAGS          =
CPPFLAGS         =
FPPFLAGS         =
LOCDIR           = home/Desktop/PETSC/Examples/example_0/
EXAMPLESC        = example_hello_0_C.c example_hello_1_C.c
EXAMPLESF        = example_hello_0_F.f
MANSEC           = example_0

include ${PETSC_DIR}/conf/variables
include ${PETSC_DIR}/conf/rules

example_hello_0_C: example_hello_0_C.o  chkopts
-${CLINKER} -o out example_hello_0_C.o  ${PETSC_LIB}
${RM} example_hello_0_C.o

example_hello_1_C: example_hello_1_C.o  chkopts
-${CLINKER} -o out example_hello_1_C.o  ${PETSC_LIB}
${RM} example_hello_1_C.o

example_hello_0_F: example_hello_0_F.o  chkopts
-${CLINKER} -o example_hello_0_F example_hello_0_F.o  ${PETSC_LIB}
${RM} example_hello_0_F.o
```

Bash file: `compile_and_execute.sh`

```
#!/bin/bash
export PETSC_DIR=~/Desktop/PETSC/petsc-3.5.3
export PETSC_ARCH=linux-dbg
make example_hello_0_C
mpirun -np 4 out
```

Compile with using just the makefile

```
henry@bluebottle:~/Desktop/PETSC/Examples/example_0$ export PETSC_DIR=~/Desktop/PETSC/petsc-3.5.3
henry@bluebottle:~/Desktop/PETSC/Examples/example_0$ make example_hello_0_C
mpicc -fPIC -Wall -Wwrite-strings -Wno-strict-aliasing -Wno-unknown-pragmas -g3 -O0  -o example_hello_0_C example_hello_0_C.o  -Wl,-rpath,/home/henry/De
lutil -lgcc_s -lpthread -ldl
/bin/rm -f example_hello_0_C.o
```

Execute Hello example

```
henry@bluebottle:~/Desktop/PETSC/Examples/example_0$ mpirun -np 4 example_hello_0_C
/bin/rm -f example_hello_0_C.o
Hello World from [0] rank
Number of processors = 4, rank = 0
Hello World from [1] rank
Number of processors = 4, rank = 1
Hello World from [2] rank
Number of processors = 4, rank = 2
Hello World from [3] rank
Number of processors = 4, rank = 3
```

Compile and Execute using a bash file on my Desktop. Maybe you will need to change the permission of your bash file

```
henry@bluebottle:~/Desktop/PETSC/Examples/example_0$ ./compile_and_execute.sh
bash: ./compile_and_execute.sh: Permission denied
henry@bluebottle:~/Desktop/PETSC/Examples/example_0$ chmod 777 compile_and_execute.sh
henry@bluebottle:~/Desktop/PETSC/Examples/example_0$ ./compile_and_execute.sh
```

15.2 On STAMPEDE

Since petsc is already installe and compile on Stapede. We just neeed to check how we can submit a job. of cource always is easy to start with a small program

`example_hello_0_C.c`

```
#include <petsc.h>

int main ( int argc, char *argv[] ){
   PetscErrorCode ierr;
   PetscMPIInt    rank, size;

   PetscInitialize(&argc, &argv, PETSC_NULL,PETSC_NULL);
   ierr = MPI_Comm_size(PETSC_COMM_WORLD,&size);
   CHKERRQ(ierr);  /* Checks error code, if non-zero it calls the error handler and then returns */
   ierr = MPI_Comm_rank(PETSC_COMM_WORLD,&rank);
   CHKERRQ(ierr);

/* Prints to standard out, only from the first processor in the communicator. Calls from other processes
are ignored. Specifically designed to print the message once for all the processes */

   //PetscPrintf(PETSC_COMM_WORLD,"Number of processors = %d, rank = %d\n",size,rank);
   //PetscPrintf(PETSC_COMM_WORLD, "Hello World from [%d] rank\n",rank);

/* Prints to standard out, from all processor in the communicator. Specifically designed to print the message from each of the processes*/
   PetscPrintf(PETSC_COMM_SELF,"Hello World from [%d] rank\n",rank);
   PetscPrintf(PETSC_COMM_SELF,"Number of processors = %d, rank = %d\n",size,rank);

   PetscFinalize();
   return 0;
}
```

Makefile

```
CFLAGS          =
FFLAGS          =
CPPFLAGS         =
FPPFLAGS         =
LOCDIR           = /home1/02817/hmoncada/CPS_5310/example_1
EXAMPLESC        = example_hello_0_C.c  example_hello_1_C.c
EXAMPLESF        =
MANSEC           = example_1

include ${PETSC_DIR}/conf/variables
include ${PETSC_DIR}/conf/rules

example_hello_0_C: example_hello_0_C.o  chkopts
-${CLINKER} -o out example_hello_0_C.o  ${PETSC_LIB}
${RM} example_hello_0_C.o

example_hello_1_C: example_hello_1_C.o  chkopts
        -${CLINKER} -o out example_hello_1_C.o  ${PETSC_LIB}
        ${RM} example_hello_1_C.o
```

Batch

```
#!/bin/bash
#SBATCH -A TG-ASC140011           # account name
#SBATCH -J example_hello_0_C      # job name
#SBATCH -o example_out.%j         # output file
#SBATCH -e example_err.%j         # error file
#SBATCH -N 1                      # total nodes requested
#SBATCH -n 4                      # total MPI tasks requested
#SBATCH -p serial                 # queue name
#SBATCH -t 00:02:00               # total time requested <hh:mm:ss>
```

```
module load petsc
module list
export PETSC_DIR=/opt/apps/intel13/mvapich2_1_9/petsc/3.5/
export PETSC_ARCH=sandybridge
make example_hello_0_C
ibrun ./out > log.txt
```

Compile and execute the hello example

1. Open TERMINAL 1. On your laptop or desktop open a first Terminal. Login into stampede:

```
>> ssh user_name@stampede.tacc.utexas.edu
or
>> ssh user_name@login.xsede.org
```

2. Set your workspace

```
>> mkdir CPS_3510
>> cd CPS_3510
>> mkdir example
>> cd example
```

3. Open TERMINAL 2. Copy all the file on this email into the folder example. On your laptop or desktop open a second Terminal. Next, Go to the folder where you have or save this files.

3.1 On that folder you call sftp

```
>> sftp  user_name@stampede.tacc.utexas.edu
or
>> sftp  user_name@login.xsede.org
```

3.2 Look for the folder where you want to save this files

```
>> cd CPS_3510
>> cd example
>> put *
>> ls
>> exit
```

How to use `put` and `get`. Please see **Transferring Files with SFTP** below

4. ON TERMINAL 1. Compile and execute

```
>> sbatch job
-----------------------------------------------------------------
              Welcome to the Stampede Supercomputer
-----------------------------------------------------------------

--> Verifying valid submit host (login4)...OK
--> Verifying valid jobname...OK
--> Enforcing max jobs per user...OK
--> Verifying availability of your home dir (/home1/02817/hmoncada)...OK
--> Verifying availability of your work dir (/work/02817/hmoncada)...OK
--> Verifying availability of your scratch dir (/scratch/02817/hmoncada)...OK
--> Verifying valid ssh keys...OK
--> Verifying access to desired queue (serial)...OK
--> Verifying job request is within current queue limits...OK
--> Checking available allocation (TG-ASC140011)...OK
Submitted batch job 5396653
```

check your job

```
>> squeue -u  5396653
             JOBID   PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
```

or

```
>> squeue -u hmoncada
             JOBID   PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           5396653      serial fdder_Pe hmoncada PD       0:00      1 (Resources)
```

5. Wait for around 5 min. Next `job.txt` is the final output

```
>> vi job.txt
```

16 Transferring Files with SFTP

16.1 Transferring Remote Files to the Local System

If we would like download files from our remote host, we can do so by issuing the following command:

```
get remoteFile
```

```
Fetching /home/demouser/remoteFile to remoteFile
/home/demouser/remoteFile                          100%   37KB  36.8KB/s   00:01
```

As you can see, by default, the "get" command downloads a remote file to a file with the same name on the local file system. We can copy the remote file to a different name by specifying the name afterwards:

```
get remoteFile localFile
```

The "get" command also takes some option flags. For instance, we can copy a directory and all of its contents by specifying the recursive option:

```
get -r someDirectory
```

We can tell SFTP to maintain the appropriate permissions and access times by using the "-P" or "-p" flag:

```
get -Pr someDirectory
```

16.2 Transferring Local Files to the Remote System

Transferring files to the remote system is just as easily accomplished by using the appropriately named "put" command:

```
put localFile

Uploading localFile to /home/demouser/localFile
localFile                                     100% 7607     7.4KB/s   00:00
```

The same flags that work with "get" apply to "put". So to copy an entire local directory, you can issue:

```
put -r localDirectory
```

One familiar tool that is useful when downloading and uploading files is the "df" command, which works similar to the command line version. Using this, you can check that you have enough space to complete the transfers you are interested in:

```
df -h

    Size     Used    Avail   (root)    %Capacity
  19.9GB   1016MB   17.9GB   18.9GB           4%
```

Please note, that there is no local variation of this command, but we can get around that by issuing the "!" command.

The "!" command drops us into a local shell, where we can run any command available on our local system. We can check disk usage by typing:

```
!
df -h

Filesystem      Size   Used  Avail Capacity  Mounted on
/dev/disk0s2   595Gi   52Gi  544Gi     9%    /
devfs          181Ki  181Ki    0Bi   100%    /dev
map -hosts       0Bi    0Bi    0Bi   100%    /net
map auto_home    0Bi    0Bi    0Bi   100%    /home
```

Any other local command will work as expected. To return to your SFTP session, type:

```
exit
```

You should now see the SFTP prompt return.