SMDO group environment setup guideline

Yicong Fu

August 2021

This document covers steps to install the software developed by the group on a linux machine (Ubuntu 20.04 LTS).

1 Prerequisite software and libraries

The following list contains the basic software and libraries you'll need to have in your machine. Note that for lab's desktops you will not have the sudo (superuser) privilege, so you'll need to contact IT support (support@ae.gatech.edu) to install them for you.

- GNU Make (`sudo apt install make`)
- CMake (`sudo apt install cmake`)
- GNU C++ compiler (`sudo apt install g++`)
- MPI C++ compiler (`sudo apt install openmpi-bin`)
- Linear algebra libraries: BLAS/LAPACK
  (`sudo apt install libblas-dev liblapack-dev`)
- Python package installer: (`sudo apt install python3-pip`)
- numpy, mpi4py, Cython: (`pip3 install numpy mpi4py Cython`)
- Paraview (`sudo apt install paraview`)

2 TACS

This sections shows how to install the finite element code TACS.

- Repository: https://github.com/smdogroup/tacs
- Documentation: https://smdogroup.github.io/tacs/

2.1 Installation

1. Clone the repository. The group's convention is to download it to `$HOME/git/`:
   `git clone git@github.com:smdogroup/tacs.git`
2. Switch to develop branch:
   `git checkout develop`

3. Download and install METIS:
   ```
   cd $HOME/git/tacs/extern
   wget http://glaros.dtc.umn.edu/gkhome/fetch/sw/metis/metis-5.1.0.tar.gz
   tar -xzf metis-5.1.0.tar.gz
   cd metis-5.1.0
   make config prefix=$HOME/git/tacs/extern/metis/ CFLAGS="-fPIC"
   make install
   ```
4. Create `Makefile.in` file (and customize it, if needed):
   ```
   cd $HOME/git/tacs && cp Makefile.in.info Makefile.in
   ```
5. Compile TACS and python wrapper:
   ```
   make && make interface
   ```
6. Set up `$PYTHONPATH` for the python wrapper, add the following line to `~/.bashrc`:
   ```
   export PYTHONPATH=${PYTHONPATH}:~/git/tacs
   ```
7. Then, run `source ~/.bashrc`

2.2 Test

Examples in `tacs/examples` can be used to test if TACS (and python wrapper) was built successfully. For example, you can run

```
cd ~/git/tacs/examples/triangle && python3 triangle.py
```

if it runs, then TACS has been installed properly.

3 ParOpt

This section shows how to intall ParOpt, the parallel gradient-based optimizer.

- Repository: https://github.com/smdogroup/paropt
- Documentation: https://smdogroup.github.io/paropt/

3.1 Dependencies

ParOpt would need MPI, numpy, mpi4py, cython and BLAS/LAPACK. But if you already installed TACS then no extra packages are needed.

3.2 Installation

1. Clone the repository. The group's convention is to download it to `$HOME/git/`:
   ```
   git clone git@github.com:smdogroup/paropt.git
   ```
2. Create `Makefile.in` file (and customize it, if needed):
   ```
   cp Makefile.in.info Makefile.in
   ```
3. Compile ParOpt and python wrapper:
   ```
   cd $HOME/git/paropt && make && make interface
   ```
4. Set up `$PYTHONPATH` for the python wrapper, add the following line to `~/.bashrc`:
   ```
   export PYTHONPATH=${PYTHONPATH}:~/git/paropt
   ```
5. Then, run `source ~/.bashrc`

3.3 Test

You could run `~/git/paropt/examples/rosenbrock/rosenbrock.py` to test if the installation succeeded.

4 TMR

This section covers steps to install TMR, the mesh refinement tool.

- Repository: https://github.com/smdogroup/tmr
- Documentation: https://smdogroup.github.io/tmr/

4.1 Download the code

1. Clone the repository. The group's convention is to download it to `$HOME/git/`:
   ```
   git clone git@github.com:smdogroup/tmr.git
   ```
2. Create and modify `Makefile.in` file:
   ```
   cp Makefile.in.info Makefile.in
   ```
   Then comment out the last three lines regarding the netgen libraries.

4.2 Dependencies

You need to install the following dependencies before installing TMR.

- TACS, ParOpt
- Blossom V
  1. Download the package to `tmr/extern`:
     ```
     cd $HOME/git/tmr/extern
     wget https://pub.ist.ac.at/~vnk/software/blossom5-v2.05.src.tar.gz
     tar -zxf blossom5-v2.05.src.tar.gz
     ```
  2. Compile:
     ```
     cd blossom5-v2.05.src
     cp ../Makefile-blossom5 Makefile && make && make lib
     ```
- OpenCASCADE
  1. Create the directory for packages:
     ```
     mkdir $HOME/packages && cd $HOME/packages
     ```
  2. Download OpenCASCADE:
     ```
     wget https://acdl.mit.edu/ESP/OCC741lin64.tgz
     tar -zxf OCC741lin64.tgz
     mv OpenCASCADE-7.4.1/ OpenCASCADE
     ```
  3. Add the following lines to `~/.bashrc`
     ```
     export CASROOT=$HOME/packages/OpenCASCADE
     export CASARCH=
     export PATH=$CASROOT/bin:$PATH
     export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CASROOT/lib
     ```
  4. Then, run `source ~/.bashrc`

5. Add the following values to `TMR_DEBUG_FLAGS` and `TMR_FLAGS` in `$HOME/git/tmr/Makefile.in`:
   `-DTMR_HAS_OPENCASCADE -DTMR_HAS_EGADS -DTMR_HAS_PAROPT`

- egads4py

1. Clone the repository to `$HOME/git`:
   `cd $HOME/git && git clone git@github.gatech.edu:gkennedy9/egads4py.git`
2. Copy over `Makefile.in.info` to `Makefile.in`
3. Compile:
   `make make interface`
4. Set up `$PYTHONPATH` for the python wrapper, add the following line to `~/.bashrc`:
   `export PYTHONPATH=${PYTHONPATH}:~/git/egads4py`
5. Then, run `source ~/.bashrc`

4.3 Installation

1. Add the following lines to `$~/.bashrc`:
   ```
   export TACS_DIR=$HOME/git/tacs
   export PAROPT_DIR=$HOME/git/paropt
   export EGADS_DIR=$HOME/git/egads4py
   ```
2. Then, run `source ~/.bashrc`
3. Compile the code and python wrapper:
   `cd $HOME/git/tmr && make && make interface`
4. Set up `$PYTHONPATH` for the python wrapper, add the following line to `~/.bashrc`:
   `export PYTHONPATH=${PYTHONPATH}:~/git/tmr`
5. Then, run `source ~/.bashrc`

4.4 Test

You could run `~/git/tmr/examples/egads/cylinder/cylinder.py` to test if the installation succeeded.

5 Miscellaneous

Some commands and resources that might be useful.

- Remote access to lab's desktop

  You can use the following command to ssh into your PC in the office:

  `ssh -J your-GTID@ssh.ae.gatech.edu your-GTID@AE-SMDO-XXXXXX.ae.gatech.edu`

  where `XXXXXX` is the machine number that can be found on your PC case.

- PACE training series

  PACE has a series of training sessions on git, linux, and PACE cluster (that has thousands of nodes and is we run very large jobs). If you want to get yourself familiar with those topics, you could check the following website for more details:

  https://pace.gatech.edu/training

- The-Missing-Semester-Of-Your-CS-Education

  A short course by MIT that covers many useful topics about the tools for developers, including shell, command line environment, git, and so on:

  https://missing.csail.mit.edu/