# Update 9/16/21

The utk branch has been superceded by the utk2 branch. This was mainly because of issues in the way utk was set up (no ability to branch off of it for code development without messing things up in subversion), so utk2 was set up with a trunk and branch directory. utk2 also includes updates from Shawn with specifying a varying absorbing boundary. The instructions below are generally the same though, just swap utk out with utk2. - SAZ

# The Situation

We got a new PC in January 2021 to run DIVIMP/3DLIM on, that way we don’t have to deal with ORNL and them locking us out of the fusion2 cluster. What follows are notes taken during the setup process to get DIVIMP installed, compiled and running. This will assume you have a working knowledge of how to navigate Linux. You can also use Putty, but I (Shawn) don’t so screw you.

# Setup and Logging In

This process is done via Ubuntu for Windows 10 (or Windows Subsystem for Linux, WSL). There are many tutorials online on how to do this, and will not be covered here. Make sure you have X windows capabilities as well. VcXsrv is a good option (Xming, the most popular option, is not updated anymore). If you’re running straight Linux then even better.

Before you can log in, you must send someone with sudo privileges your public RSA key. Steps to generate it can be found [online here](https://ubuntu.com/tutorials/ssh-keygen-on-windows#1-overview). The key will be found at ~/.ssh/is\_rsa.pub. The person with sudo access will need to [INSTRUCTIONS ON HOW TO DO THIS GOES HERE].

You also need to be on the UTK VPN to log in. Instructions [found here](https://help.utk.edu/kb/index.php?func=show&e=2712).

Once that’s all done, the log in command is, for example:

ssh -X [zamp@nec695914.nomad.utk.edu](mailto:zamp@nec695914.nomad.utk.edu)

where your name is your net ID, unless otherwise specified. No password is required since your public RSA key will be added to the system.

Bonus: Add the following to the bottom of your .bashrc file if you want a cleaner prompt (all on one line):

export PS1='${debian\_chroot:+($debian\_chroot)}\[\033[01;32m\]\u@dipsy\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '

# Downloading the DIVIMP Repository

First, subversion was installed (sudo apt install subversion). In your home directory (cd ~) create the directory utk (mkdir utk). Put the utk branch into this with:

svn checkout [http://starfire.utias.utoronto.ca/svn/divimp/branches/utk2](http://starfire.utias.utoronto.ca/svn/divimp/branches/utk) utk2

You will need a username and password from David Elder to access the repository. If you want to merge with the main trunk to get the latest and greatest, the process is:

cd utk2/trunk

svn merge [http://starfire.utias.utoronto.ca/svn/divimp/trunk](https://starfire.utias.utoronto.ca/svn/divimp/trunk)

Normally you would create a new directory like utk-merge and then checkout the utk branch into and do the merge there, so you don’t mess with your files that are in working order and break it. It is up to you to decide what to do when conflicts arise, it is a tricky process so this is best done if you know what you are doing.

# Compiling DIVIMP and 3DLIM libraries

Before getting to DIVIMP, I’ll need to compile the NetCDF4 libraries for Fortran. This only needs to be done once, and then other users who want to compile DIVIMP and can just link to this library.

## Compiling NetCDF4

First, I need to compile the netcdf libraries. This is a process, [outlined here](https://www.unidata.ucar.edu/software/netcdf/docs/getting_and_building_netcdf.html#getting). First, need to download and build zlib:

cd utk2/trunk/libsrc/

wget <http://www.zlib.net/zlib-1.2.11.tar.gz>

tar xvzf zlib-1.2.11.tar.gz

cd zlib-1.2.11

ZDIR=/home/zamp/utk2/trunk/local

./configure --prefix=${ZDIR}

make check

make install

This put some zlib stuff in utk/local/lib. Now do a similar thing for HDF5.

cd /utk2/trunk/libsrc

tar xvzf hdf5-1.8.17.tar.gz

cd hdf5-1.8.17/

H5DIR=/home/zamp/utk2/trunk/local

./configure --with-zlib=${ZDIR} --prefix=${H5DIR} --enable-hl

make check Note: Fails h5dump but whatever keep going.

make install

Next is to build netcdf from source. Note: This I believe are the C files, which are then used by the later Fortran files.

cd /utk2/trunk/libsrc/

tar xvzf netcdf-4.4.1.tar.gz

cd netcdf-4.4.1

LD\_LIBRARY\_PATH=/home/zamp/utk2/trunk/local/lib

NCDIR=/home/zamp/utk2/trunk/local

The next command is (all one line ignore the line break you see here):

CPPFLAGS='-I/home/zamp/utk2/trunk/local/include' LDFLAGS='-L/home/zamp/utk2/trunk/local/lib' ./configure --prefix=${NCDIR}

Needed to install “m4”:

sudo apt install m4

Then go back and run the previous command again. Continue with:

make check

make install

This puts a bunch of NetCDF things into bin, include, lib and share. At this point NetCDF has been installed, and we now need to install the Fortran interface.

cd /utk2/trunk/libsrc/

tar xvzf netcdf-fortran-4.4.4.tar.gz

cd netcdf-fortran-4.4.4

NCDIR and LD\_LIBRARY\_PATH were already set above, but you may need to set them again if you took a break and reset your session or something. Continuing:

NFDIR=/home/zamp/utk2/trunk/local

CC=/usr/bin/gcc

FC=/usr/bin/gcc

CPPFLAGS=-I${NCDIR}/include LDFLAGS=-L${NCDIR}/lib ./configure --prefix=${NFDIR}

make check

make install

NetCDF4 with the Fortran interface is now installed! That sucked!

# Compiling ghost

No idea what this library is for but we need it.

cd /utk2/trunk/libsrc/

tar xvzf ghostv80.adobe.tar.gz

cd ghost\_v8/g80procs

Need to change the setup file. Make a backup of the original

cp setup setup\_backup

cp setup setup.gfortran

Open up the setup.gfortran file and replace every instance of pgf90 with gfortran and cc with gcc. Then we need to modify the following link\_ files by replacing pgf90 with gfortran too:

link\_decode

link\_trand

link\_trenc

link\_postcl

link\_postsc

Then

./setup.gfortran

Recompile: y

Is your machine managed… : n

Byte ordering like… : n

Manufactured by … : n

Then when it asks for postprocessors say yes to Adobe Postscript Colour and Monochrome and GKS. Then open up move\_ghost and change all instances of /usr/local/ to /home/zamp/utk2/trunk/local. Then

./move\_ghost

Now we have all the ghost and postcl, etc. stuff in our local/lib and local/bin folders! Very not fun!

# Compiling DIVIMP

Now it’s time to actually compile DIVIMP. I had a couple of issues and had to change an instance of “iposr8” with “ipos” since iposr8 could not be found anywhere, I had to comment out the subroutine “find\_free\_unit\_number” in utility\_com.f since it was already defined in utility.f, and I had to dummy a tmp\_mass variable to 0 in sol23.f since a function “get\_bg\_mass” was also not able to be found. This effectively breaks SOL23, but we don’t use it at UTK as of the time of this writing.

Went into the Makefile.utk to change all instances of /home/zamp/utk/ with /home/zamp/utk2/trunk/ and then:

cd /utk2/trunk/div6

cp Makefile.utk Makefile

make clean

make

At this point it successfully compiled (though if it runs correctly is a task for later).

# Compiling Eirene

First go into the eirene07 folder. I need to save the Makefile.ornl\_gfortran as Makefile.utk for our purposes.

Created Makefile.utk. You can look at it at /home/zamp/utk2/trunk/eirene07/Makefile.utk, but the crucial point is changing LIBS and DIVMAIN:

DIVMAIN=/home/zamp/utk2/trunk

LIBS= -L/home/zamp/utk2/trunk/local/lib -lgks -L/usr/lib/x86\_64-linux-gnu/ -lX11 -lXt -lc

Note I coudn’t figure out how to install libgr.a, so I just compiled it without it… will see if that hurts us. (SAZ 9/1/21 Don’t think this is the case anymore?)

cp Makefile.utk Makefile

make

Compiled at least.

# Compiling OUT

Setup UTK Makefile:

cd utk/out6/

cp Makefile.ornl\_gfortran Makefile.utk

In it change:

DIVMAIN=$(HOME)/utk2/trunk

DIVLOCALINC=/home/zamp/utk2/trunk/local/include

LOCAL\_DIV\_LIB=/home/zamp/utk2/trunk/local/lib

Then we need to install the jpeg library:

sudo apt install libjpeg-dev

And then we can make it:

make

At this point OUT has been compiled.

# Compiling triangle

This is really complicated:

cd utk/triangle

make

# OPEN-ADAS

I needed to download the files Jake had already set up in the openadas folder on the Google Drive. This has things like ionization shit and whatever. I copied the folder, openadas, over into my home directory with WinSCP. Also, connecting with WinSCP just requires putting [nec695914.nomad.utk.edu](mailto:zamp@nec695914.nomad.utk.edu) in as the host name and then my username.

# Running a DIVIMP test case

First create a rundiv file:

cp rundiv.ornl rundiv.utk

Change

export ADASCENT=/home/zamp/openadas

PROGDIR=utk

export GLI\_HOME=/home/zamp/utk/local/lib

export PATH=$PATH:/home/zamp/utk/local/bin/

A reminder that the DIVIMP run syntax is:

rundiv.utk <.d6i file> <.d6o file> <grid> none none <optional background .bgp file>

The scripts such as cpc use tcsh shell, so we need to install that so they work:

sudo apt install tcsh

Somehow I made it this far being lazy not adding current directory to my path, but I’ll need it now. We may need the path to the scripts in there as well. In your .bashrc file:

export PATH=$PATH:.:/home/zamp/utk/scripts/

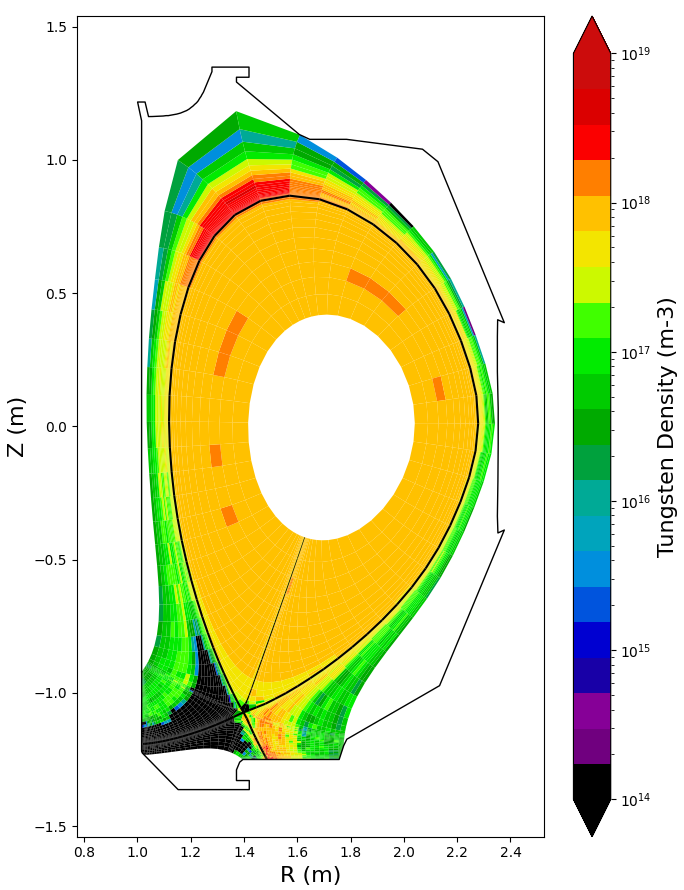
Then source it:

source .bashrc

Started with testcase10:

./rundiv.utk testcase10 testcase10-grid carre\_145672\_3000 none none none

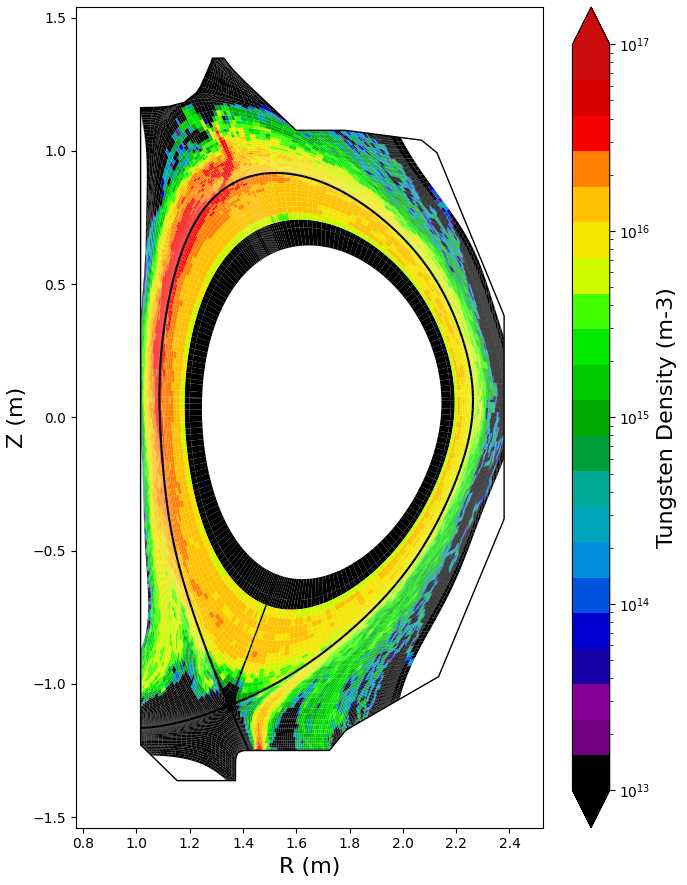
Believe it or not, this actually ran through without any mysterious errors or anything! Let’s look at it in our plotting GUI.



Wow, seems to have worked!!! Note this testcase10 isn’t W I don’t think, but whatever. Let’s try another test case with the MRC extended grid:

rundiv.utk d-167196-modE-expW-dft60-W\_test testcase10-grid grid\_167196\_3000\_hires none none d-167196-modE-shelf-bg

Incredibly, it worked! DIVIMP may at this point be up and running!



# Compiling DG and EQUTRN

First, in dg\_carre\_repo run the setup script:

setup.bash

Again copy over for a Makefile:

cp Makefile.ornl Makefile.utk

Change:

LIBS=-L/usr/lib/x86\_64-linux-gnu/ -lXm -lXt -lX11 -lm -lpthread

INCLUDES=-I/usr/include/

Need libmotif (for libXm.a) and ksh shell:

sudo apt install libxm4 (maybe unneeded)

sudo apt install libmotif-dev (probably just need this one)

sudo apt install ksh

Then to make we actually use:

make all

The DG executable is then the one under LINUX:

cd LINUX

dg &

EQUTRN is a collection of tools that can modify equilibriums needed for DG. To compile nothing needs be changed:

cd DG/equtrn

cp Makefile.ornl Makefile.utk

cp Makefile.utk Makefile

make clean

make

# Compiling MSCL and Carre

First go into mscl and compile. No changes needed really.

cp Makefile.utk Makefile (It’s the same as Makefile.ornl, just renamed to .utk for consistency)

make

Then go to the Carre directory, and make our Makefile:

cp Makefile.ornl Makefile.utk

In the Makefile change:

LDFLAGS=$(NCAR\_LIBS) -L/usr/lib/x86\_64-linux-gnu -lX11 -lm -lmscl

Then:

cp Makefile.utk Makefile

make

Testing Carre will require a bit more effort since I gotta make a grid and all that, but it at least compiles.

# Compiling 3DLIM

My baby. Currently the working version of 3DLIM is under lim3:

cd utk/lim3

cp Makefile.shawn Makefile.utk

Change:

DIVMAIN=/home/zamp/utk

DIVLOCALINC=/home/zamp/utk/local/include

LOCAL\_LIB=/home/zamp/utk/local/lib/

LOCAL\_DIV\_LIB=/home/zamp/utk/local/lib

LIBS= -Wl,-rpath=$(LOCAL\_DIV\_LIB) -L$(LOCAL\_DIV\_LIB) -lnetcdff -lnetcdf -lghost$(LIB\_VER) -lpostcl$(LIB\_VER)

Then:

cp Makefile.utk Makefile

Make

I received the following error:

make: \*\*\* No rule to make target 'mod\_diagvel\_unstruc\_lim.o', needed by 'lim3Ogcc'. Stop.

This in fact was a file David Elder had forgot to include. He has since added it, so now we must merge with the trunk again. First check in everything we’ve done so far:

svn update

Seems David went ahead and did something with the utk branch maybe, so just take every incoming file (tf) or merge (m) when asked. This for some reason gets caught up on “Checking r939…” and I have no idea why. It seems to be an issue with the fact that mod\_sol22\_input\_lim.f90 had been deleted and something something I don’t really know. But I found a solution online:

touch lim3/comsrc/f90/mod\_sol22\_input.f90

svn revert lim3/comsrc/f90/mod\_sol22\_input.f90

rm lim3/comsrc/f90/mod\_sol22\_input.f90

And then I can go through the resolve process again, and just use the merge option on everything in regards to adding/deleting new files:

svn resolve # (m) for every prompt

Now let’s add all the Makefile.utk and rundiv.utk to the branch:

svn add out6/Makefile.utk div6/Makefile.utk eirene07/Makefile.utk lim3/lim/Makefile.utk rundiv.utk

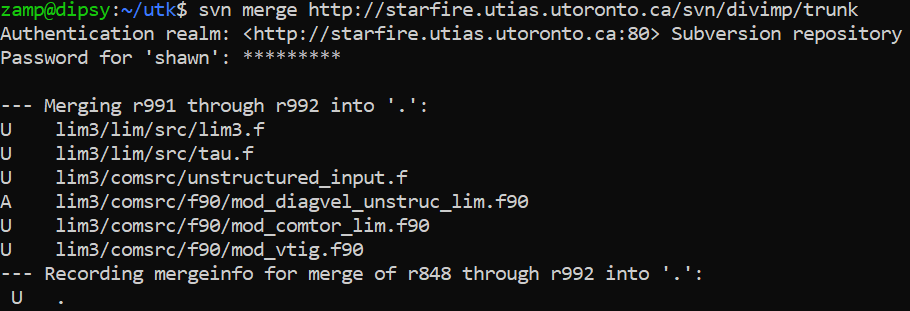
Check in these additions:

svn ci -m "Initial commit from dipsy. Added Makefile.utk and rundiv.utk."

svn update

Now we can merge with the trunk:

svn merge http://starfire.utias.utoronto.ca/svn/divimp/trunk



We can see the files David has added (A) and updated (U). Now when we go back to compile 3dlim, we run into a conflict that is in mod\_comxyt\_lim.f90, where it has the .working and .merge\_left and what not put into the file as a result of a conflict. Let’s go in and choose the most updated version, .merge-right.r988 (literally go into the file and manually delete the .working and .merge-left code). Save the file again.

cd utk/lim3/lim

cp Makefile.utk Makefile

make

At this point it successfully compiles!

Also need to compile the 3DLIM version of OUT (don’t really use it at all but it’s generally needed to run).

cd utk/lim3/out

cp Makefile.ornl\_gfortran Makefile.utk

Change:

DIVMAIN=/home/zamp/utk

DIVLOCALINC=/home/zamp/utk/local/include

LOCAL\_LIB=/home/zamp/utk/local/lib

LOCAL\_DIV\_LIB=/home/zamp/utk/local/lib

Then compile:

cp Makefile.utk Makefile

Make

Compiled! Next is to test it.

# Testing 3DLIM

Need to update the run script. So open up runlim.ornl, and change:

VER=gcc

export ADASCENT=/home/zamp/openadas

export RUNROOT=/home/zamp/utk/lim3

export DATAROOT=/home/zamp/utk/lim3

export EXEROOT=/home/zamp/utk/lim3

export BATDIR=/home/zamp/utk/scripts

export LOCALDIR=/home/zamp/utk/local/bin

Save as runlim.utk. Give executable permissions to the run script if needed

chmod 777 runlim.utk

Then run a test file:

./runlim.utk colprobe-a8 colprobe

And the test seems to have worked! In the results/ folder the .nc .lim.gz and .dat file can be pulled out for the lim\_plots\_gui.py plotting GUI. Don’t know if OUT was compiled correctly but eh, who cares. This should officially end the setup of DIVIMP and 3DLIM on the dipsy computer.

# HEAT Installation

Instructions Last Updated on 02/12/2021

HEAT Documentation:

<https://heat-flux-engineering-analysis-toolkit-heat.readthedocs.io/en/latest/#>

Installation Video:

<https://youtu.be/mDui3_z_2oM>

Make and move into directory for storing HEAT AppImage:

mkdir HEAT

cd HEAT

Download the AppImage:

wget https://github.com/plasmapotential/HEAT/releases/download/v1.2-beta/HEAT\_AppImage-v1.2-beta-x86\_64.AppImage

Make the file executable:

chmod +x HEAT\_AppImage-v1.2-beta-x86\_64.AppImage

Run HEAT as executable:

./HEAT\_AppImage-v1.2-beta-x86\_64.AppImage

Alternative: run HEAT as executable when not on LINUX

./HEAT\_AppImage-v1.2-beta-x86\_64.AppImage -a <address> -p <port>

GUI access is then possible in the internet browser. Simply navigate to the terminal output address and run HEAT or field line traces following the documentation instructions.