# Installation Notes

February 26, 2020

Kylie Tailin Zhu

## 1 Installation - MacOX 10.14

### 1.1 uproot

Install with conda:

```
conda config --add channels conda-forge
conda install uproot
```

The conda installer also installs optional dependencies (except for Pandas).

### 1.2 pandas

Install with conda:

```
conda install -c anaconda pandas
```

### 1.3 ROOT

Install with conda:

```
conda install -c conda-forge root
```

version:

```
ROOT Version: 6.16/00
Built for macosx64 on Jun 23 2019, 15:10:00
From @
```

where PyROOT is enabled.

### 1.4 matplotlib

Install with conda:

```
conda install -c conda-forge matplotlib
```

### 1.5 rootpy

Clone the repository with git:

```
git clone git://github.com/rootpy/rootpy.git
```

But I have to do this again after opening a new terminal window.

Hence instead, using conda:

```
conda install -c nlesc rootpy
```

But rootpy only works for Python 2.6 and 2.7.

### 1.6 Python2 (this env cannot enable PyROOT. )

Create a Python 2 Environment:

```
conda create --name python2 python=2 #build python 2.7.15
```

To activate this environment, use

```
conda activate python2
```

To deactivate an active environment, use

```
conda deactivate
```

### 1.7 Python2 with ROOT/PyROOT

To solve this, build a env with root:

```
conda create -n myrootenv python=2 root -c conda-forge
```

And use

```
pip install uproot
pip install rootpy
```

(Warning: The previous method with 'conda install' doesn't work for 'uproot' and 'rootpy' in this env. ROOT was disabled after installing rootpy and exetremely long process for installing uproot. But other packages are fine.)

'conda install' should be used for 'matplotlib' and 'pandas', as described above.

To activate this environment, use

```
conda activate myrootenv
```

To deactivate an active environment, use

```
conda deactivate
```

#### 1.7.1 Troubleshooting

For some reason, using 'matplotlib' can be given several DEBUG messages and can not show the graph window and terminate the programme. Hence, ROOT was used to plot the graphs rather than 'matploylib' in some case.

# 2 Simulation

## 2.1 test_uproot_script.py

The script runs follow the installation steps in the myrootenv environment and a histogram was plotted. See test_root.py for a improved version using ROOT instead of matplotlib.

# 3 AmpGen

website: https://github.com/GooFit/AmpGen#getting-started

## 3.1 Installation and Compilation

Get the source:

```
git clone http://github.com/GooFit/AmpGen/ --recursive
```

With ROOT already built:

```
source /Applications/root_v6.18.04//bin/thisroot.sh
```

Go to the AmgGen directory:

```
cd ~/AmpGen
mkdir build
cd build
```

Set up the compiler:

```
cmake .. -Dcxx17=On -DCMAKE_C_COMPILER=llvm-gcc -DCMAKE_CXX_COMPILER=llvm-g++ -DCMAKE_Fortran_C
```

Compile

```
make -j 8
```

lastly set AMPGENROOT

```
export AMPGENROOT=/Users/zhutailin/AmpGen/
```

Try an example:

```
cd bin
./Generator DtoKKpipi_v2.opt --nEvents=1000 --Output=output_DtoKKpipi_v2.root
```

generates a .root file.

### 3.1.1 Information on debugging:

a. change the conditions in CompilerWrapper.cpp in the AmpGen directory

   open -a TextWrangler src/CompilerWrapper.cpp

add // before line 34 and 43.

b. to show the compiling source used, add

   CompilerWrapper::Verbose true

at the start of the .opt file of the decay. command it using #.

c. To get rid of the warnings (optional), change the line in CompilerWrapper.cpp:

if( m_cxx.find(“clang”) = std::string::npos)

to

```
if( m_cxx.find("clang") = std::string::npos || m_cxx.find("llvm-g++") = std::string::npos)
```

[ ]: