Updated June 2022

# Getting started with TUCS

1. Get familiar with basic Unix commands: [cheatsheet](http://www.mathcs.emory.edu/~valerie/courses/fall10/155/resources/unix_cheatsheet.html)
2. Install [XQuartz](https://www.xquartz.org/) (for Mac, to view images over ssh)
3. Set up lxplus:
   1. ssh -XY [USER@lxplus.cern.ch](mailto:USER@lxplus.cern.ch)
   2. Add to your .bash\_profile (vim is a good basic text editor):

export ATLAS\_LOCAL\_ROOT\_BASE=/cvmfs/atlas.cern.ch/repo/ATLASLocalRootBase

alias setupATLAS='source ${ATLAS\_LOCAL\_ROOT\_BASE}/user/atlasLocalSetup.sh'

export ALRB\_localConfigDir=$HOME/localConfig

echo 'Setting up ATLAS'

setupATLAS > /dev/null

export AtlasSetup=/afs/cern.ch/atlas/software/dist/AtlasSetup

alias asetup='source $AtlasSetup/scripts/asetup.sh'

1. Clone TUCS from [here](https://gitlab.cern.ch/atlas-tile-offline/Tucs/-/blob/master/GIT_HOWTO.md).
   1. Manual Setup:
      1. Run: asetup Athena 22.0.46
      2. Every subsequent login, run: asetup. This will continue to use 22.0.46.
   2. Automatic setup:
      1. source setup.sh

From main TUCS folder:

[https://gitlab.cern.ch/atlas-tile-offline/Tucs](https://urldefense.com/v3/__https://gitlab.cern.ch/atlas-tile-offline/Tucs__;!!BpyFHLRN4TMTrA!6-G8fLgJNxV70AWT106M_1Xo-9tE0PTloVCejO0iVc4UkA3XUArqYS2Qu1RMJaei-sviODwB5hknZ7L2HZoR-Xr7IX0$)

1. Familiarity with Git and TUCS:
   1. Follow the ATLAS Git tutorial [here](https://atlassoftwaredocs.web.cern.ch/gittutorial/).
   2. Useful [twiki page](https://twiki.cern.ch/twiki/bin/view/Atlas/TileCalibrationTucs)
   3. Basic organization of TUCS [presentation](https://twiki.cern.ch/twiki/pub/Atlas/TileCalibrationTucs/tucs_documentation.pdf)
2. Make these changes (need to upload to git) [not resolved ; push maybe]

workers/cis/CalibDist.py: kBlue -> ROOT.kBlue & hvallo -> self.hvallo

workers/cis/CISRecalibrateProcedure.py: kTrue -> ROOT.kTrue

workers/cis/DetectorTimeStability.py: TGraphErrors -> ROOT.TGraphErrors & kYellow -> ROOT.kYellow

workers/cis/RMSPlots.py: kTrue -> ROOT.kTRUE

workers/cis/TimeStability.py: TPaveStats -> ROOT.TPaveStats

macros/laser/laserStudyFlags.py: line 103, flag\_region -> region

workers/cis/ReadCIS.py: runNumber directory threshold 343004 -> 375187

workers/cis/FlagPlots.py: ignore ES and NES flags (a few lines changed)

macros/cis/Public\_Super\_Macro.py: change print statements and has\_key to Py3

macros/cis/TimingFlag

macros/cis/StudyFlag.py change print statements

workers/cis/SetLowCISThreshold.py:

In processStop you should only calculate the avg/rms IF the list is nonempty.

This becomes relevant when you are say only running over a lowgain region

My version:

if len(hg\_old\_cis):

hi\_old\_avg = ROOT.TMath.Mean(len(hg\_old\_cis), array('f', hg\_old\_cis))

hi\_old\_rms = ROOT.TMath.RMS(len(hg\_old\_cis), array('f', hg\_old\_cis))

else:

hi\_old\_avg = 0

hi\_old\_rms = 0

if len(lg\_old\_cis):

lo\_old\_avg = ROOT.TMath.Mean(len(lg\_old\_cis), array('f', lg\_old\_cis))

lo\_old\_rms = ROOT.TMath.RMS(len(lg\_old\_cis), array('f', lg\_old\_cis))

else:

lo\_old\_avg = 0

lo\_old\_rms = 0

workers/CIS/CalibDist.py:

You should exclude the demonstrator as this causes consistent outliers. Add a line in the ProcessRegion loop to say:

if 'LBA\_m14' in region.GetHash():

continue

workers/cis/SetLowCISThreshold.py

def ProcessStop(self):

if len(hg\_old\_cis):

hi\_old\_avg = ROOT.TMath.Mean(len(hg\_old\_cis), array('f', hg\_old\_cis))

hi\_old\_rms = ROOT.TMath.RMS(len(hg\_old\_cis), array('f', hg\_old\_cis))

else:

hi\_old\_avg = 0

hi\_old\_rms = 0

if len(lg\_old\_cis):

lo\_old\_avg = ROOT.TMath.Mean(len(lg\_old\_cis), array('f', lg\_old\_cis))

lo\_old\_rms = ROOT.TMath.RMS(len(lg\_old\_cis), array('f', lg\_old\_cis))

else:

lo\_old\_avg = 0

lo\_old\_rms = 0