Step 0: Setup

0.0 Set up proxy:

```
# Check whether voms proxy exists and is up-to-date; if it doesn't
exist or has expired, set it again
checkAndSet_voms() {
    NEEDS_TO_BE_SET="true"
    if [ -e ${X509_USER_PROXY} ]; then
        VOMS_TIME_LEFT=`voms-proxy-info --all | grep timeleft | head -
1 | sed "s|^timeleft *: *\([0-9:]*\)*$|\1|"`
        VOMS_HOURS_LEFT=$(echo ${VOMS_TIME_LEFT} | sed "s|\([0-
9]*\):[0-9]*:[0-9]*|\1|")
        echo "Hours left on grid certificate: ${VOMS_HOURS_LEFT}"
        if [ "${VOMS_TIME_LEFT}" == "00:00:00" ]; then
            echo "Proxy no longer valid; needs to be reset"
        elif [ "${VOMS_TIME_LEFT}" == "0:00:00" ]; then
            echo "Proxy no longer valid; needs to be reset"
        elif [ "${VOMS_HOURS_LEFT}" -lt 18 ]; then # less than 18
hours left on voms certificate
            rm ${X509_USER_PROXY}
            voms-proxy-destroy
            echo "Proxy valid for less than 18 hours; needs to be
reset"
        else
            echo "Proxy still valid"
            NEEDS_TO_BE_SET="false"
        fi
    else
        echo "Proxy file not found; needs to be set"
    fi
    if [ "${NEEDS_TO_BE_SET}" == "true" ]; then
        echo "setting voms proxy"
        voms-proxy-init --rfc --voms cms -valid 192:00
    fi
}
export X509_USER_PROXY=${HOME}/private/x509up_u$(id -u)
checkAndSet_voms
```

0.1 Clone this repository, suggested path: `~/private/tmPyUtils:` https://github.com/tanmaymudholkar/tmPyUtils/

0.2 Clone this repository, suggested path: `~/private/tmCPPUtils:` https://github.com/tanmaymudholkar/tmCPPUtils/tree/master

0.3 Clone this repository, suggested path: `~/private/STEALTH:` https://github.com/cmu-stealth-analysis/STEALTH

0.4 Set up an empty `CMSSW 10_2_10` environment, suggested path: `~/nobackup/cmssw/CMSSW_10_2_10`

0.5 Set the following environment variables in your `bashrc`:

```
# If not running interactively, stop sourcing here
case $- in
    *i*) ;;
      *) return;;
esac


export TMPYUTILS=/uscms/homes/t/tmudholk/private/tmPyUtils/ # replace
export TMCPPUTILS=/uscms/homes/t/tmudholk/private/tmCPPUtils/ # replace
export EOSPREFIX=root://cmseos.fnal.gov/
export XROOT_REDIRECTOR=root://cms-xrd-global.cern.ch/
export EOSTMPAREA=/uscms/home/tmudholk/nobackup/eos_tmp_area # create a
similar area in your nobackup directory and replace
export TM_UTILS_PARENT=/uscms/home/tmudholk/private # replace
export STEALTH_ROOT=/uscms/home/tmudholk/private/stealth/STEALTH #
replace
export STEALTH_EOS_ROOT=/store/user/lpcsusystealth
export
STEALTH_CMSSW_BASE=/uscms/home/tmudholk/nobackup/cmssw/CMSSW_10_2_10 #
replace
export STEALTH_ARCHIVES=/uscms/home/tmudholk/nobackup/archives # create a
similar area in your nobackup directory and replace
export CONDORWORKAREAROOT=/uscms/home/tmudholk/nobackup/condorWorkAreas
# create a similar area in your nobackup directory and replace
export ANALYSISROOT=/uscms/home/tmudholk/nobackup/analysisAreas # create
a similar area in your nobackup directory and replace
export SCRATCHAREA=/uscms/home/tmudholk/cmslpc_scratch # replace with
your scratch area
export PYTHONPATH=${HOME}/private/tmPyUtils:${PYTHONPATH} # change the
path to tmPyUtils if needed
```

0.6 (Important) Make sure to do "`cmsenv`" inside your CMSSW setup before you proceed. Once that is done, compile everything inside your copy of `tmCPPUtils`. For example, "`cd ~/private/tmCPPUtils/ROOTUtils && make`". **You also have to run "make" inside the event selection directory.**

0.6.5 You need to have a script that creates a tarball with only the source code (not compiled output) from tmPyUtils and tmCPPUtils, in order to upload it to EOS. You also need a script that extracts this tarball and compiles it on the platform that it runs on when you submit a condor job.

Practically, here's what you have to do. Assuming you've saved the util folders into ~/private/tmCPPUtils and ~/private/tmPyUtils, you need to copy the attached two scripts into ~/private/update_tmUtilsTarball.sh and ~/private/extract_tmUtilsTarball.sh. (And remember to flag them both as executables with chmod +x ...)

0.7 Extract the two attached tarballs somewhere, and then copy the folders "`fileLists`" and "`xSecLumiInfo`" to your `${STEALTH_ROOT}`. (Note: Copy the folders themselves, the contents should be one level down; e.g. "`${STEALTH_ROOT}/xSecLumiInfo/lumi_notes.txt`" should be a valid path.

0.8 Here's an extract from my bashrc:

```
stealth_setup ()
{
    cd private/stealth/STEALTH
    source setupEnv.sh
}
```

That way, after I log on to LPC, if I'm going to work on the Stealth analysis, I call `"stealth_setup".` Modify this recipe to your taste (or use it as it is): you will have to call `setupEnv.sh,` which in turn requires the environment variables above to be set.

0.9 Set up the combine command for the 10-2-X release using the commands in (be sure to match version v8.1.0 or else bugs can come up):

https://cms-analysis.github.io/HiggsAnalysis-CombinedLimit/#combine-v8-cmssw_10_2_x-release-series

0.9.1 Set up the combine tool using

https://cms-analysis.github.io/HiggsAnalysis-CombinedLimit/#combine-tool

Once the set up steps are done, next are the steps needed to run the actual analysis:

Step 1: Ntuplizer (you can skip Step 1 for the time being, because I'm sending over my input files folder which has paths to already processed data)

Ntuplizer link: https://github.com/cmkuo/ggAnalysis/tree/110X

Ntuplizer input: MiniAOD
Ntuplizer output: ROOT n-tuples

Step 2: Event selection.

Input: newline-separated list of file paths to the ggNtuplizer outputs
Output: submits jobs via LPC condor to run event selection on all outputs
The commands to run the selections are at the bottom of: https://github.com/cmu-stealth-analysis/STEALTH/blob/master/submitEventSelectionJobs.py

Between the first four and the last two scripts in submitEventSelectionJobs.py, remember to change the jet threshold to 50 GeV

i.e. change this line: https://github.com/cmu-stealth-analysis/STEALTH/blob/master/eventSelection/include/parameters.h#L22

Step 3: run "merge" scripts: bottom of: https://github.com/cmu-stealth-analysis/STEALTH/blob/master/runSelectionMerge.py

Input: results of event selection (typically thousands of small files.)
Output: merged selections (complete selections grouped by year.)

Step 4: Run full analysis: bottom of:
https://github.com/cmu-stealth-analysis/STEALTH/blob/master/runAnalysis.py

Potential bug fixes/ extra steps for special cases:

1. If custom ~/.bash_profile is not sourced automatically, add to ~/.bash_profile

```
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi
```

2. To use the scratch area in the LPC:

```
mkdir -p /uscmst1b_scratch/lpc1/3DayLifetime/<your-user-name>
cd ${HOME}
ln -sf /uscmst1b_scratch/lpc1/3DayLifetime/<your-user-name>
cmslpc_scratch
```

You should also create a folder named “merged” inside your LPC area, as the code expects this path to exist.