```
################################################################################
#Step 1 - Mount/obtain data

#cop-out on this one


################################################################################
#Step 2 -  Filelists

#First ssh over to -> po and then -> rc18

        ssh barrettj@po
        ssh barrettj@rc18

#demonstrate that the Westford data is mounted in slot #1

        da-client
        >> mstat?1
        >> quit
        ls /mnt/disks/1/1/data/

#make sure we have access to the latest version of vsum

        source ./setup-DiFX-2.6.bash

#demonstrate vsum use/behavior

        stdbuf -o0 vsum -s --mark6slot 1 | tee ./vt9105_wf.filelist

#kill this early with ^C,  and exit back to po

        exit

#now copy in the pre-prepared file lists into our TOWDATA directory
        cd $TOWDATA
        pwd
        cp ../prep/*filelist ./

################################################################################
#Steps 3 & 4 - VEX pt. 1

#Now we want to start putting together the correlator .vex file.
#We need to do this on demi (b/c of sked).

        ssh barrettj@demi

#cd over to $TOWDATA dir

        cd $TOWDATA
        pwd
        ls

#now grab the .skd file
```

```
    wget ftp://cddis.gsfc.nasa.gov/vlbi/ivsdata/aux/2019/vt9105/vt9105.skd

#convert it to a .vex file

    sked ./vt9105.skd
    ? VWC vt9105.vex.obs
    ? quit
    ls # show our new vex file

###############################################################################
#Steps 3 & 4 - VEX pt. 2

#exit back to po now, and cd to TOWDATA dir

    exit
    cd $TOWDATA

#now we need to copy in the vgos template

    cp ../prep/template.vex ./template.vex.tmp

#now cut and insert the parts of the obs. vex file that we need:
#(EXPER, SCHED, SOURCE, SITES):

    source ../prep/tweak_vex.sh ./vt9105.vex.obs ./vt9105.vex.corr

#demonstrate that the script isn't doing anything more complicated than
#cut/paste, and find/replace GEOSX.SX with VGOS:

    cat ../prep/tweak_vex.sh

#now open and edit the .vex file to add correlator specific information

    nano ./vt9105.vex.corr

#now insert the following lines into the $EXPER section
#(time range is the 1st 2 scans):

exper_num = 1234;
exper_nominal_start=2019y104d00h00m00s;
exper_nominal_stop=2019y105d23h59m59s;

#Next copy in the EOP/CLOCKS section:
#Make note of the fact that setting CLOCK section is often an iterative process:

    cat ../prep/vt9105_clocks.vex

#copy and paste into vt9105.vex.corr, via nano, insert between $ANTENNA and
#$BBC sections (this is 1 page-down ^V )

    nano /vt9105.vex.corr

###############################################################################
```

#STEP 5 - check for calcserver

#now check that the env, and CALC_SERVER is set and running, to do this
#first source in the DiFX environment, comment on versions:

```
source ~/scripts/haystack-difx-production-env.sh
echo $CALC_SERVER
checkCalcServer $CALC_SERVER
```

##
#STEP 6 - vex2difx

#now we move on to the h.v2d file, so copy it in from the prep folder.
#We will not edit it in real-time, but should explain the various parts:

```
cp ../prep/h.v2d ./
nano ./h.v2d
```

##
#Step 7

#now we can go ahead and run vex2difx:

```
vex2difx h.v2d
```

#output should be:
#start date: 2019y104d00h00m00s
#stop date: 2019y105d23h59m59s
#2 job(s) created.

##
#Step 8 - DiFX

#now open up a second window and ssh over to po, in order to run errormon2

```
ssh barrettj@po
source ~/scripts/haystack-difx-production-env.sh
errormon2 5
```

#now switch back to the original window and fire up DiFX:

```
startdifx -f -n h.joblist
```

#wait with bated breath...this should take around 1m30s for the first scan
#and another 1m for the second.

##
#STEP 9 - difx2mark4

#now that we are done, exit the errormon2 window, and log-back in,
#but this time set-up a clean DiFX install
#to work around the difx2mark4 segfault

```
    exit
    ssh barrettj@po
    source ~/scripts/haystack-difx2mark4-env.sh
    cd $TOWDATA
    ls

#copy in the station codes file and show them the contents:

    cp ../prep/station.codes
    cat ./station.codes

#now run difx2mark4 (shouldn't take much time at all)

    difx2mark4 -v -d -b X 2300 14000 -e 1234 -s station.codes


#########################################################################################
#STEP 10 - fourfit

#now we can take look with fourfit.
#first exit po, and re-login to get a clean environment,
#now and make sure we have hops 3.20 in our path
#should point out to them that if they want to use any of the post-processing
#tools we will distribute they must not have another version of HOPS in
#their path (this can happen if they have source'd in DiFX with HOPS enabled)

    ssh -XY barrettj@po
    source /swc/hops/x86_64-3.20/bin/hops.bash

#this can be done with the control file from the last processed session
#brief demo with one particular baseline (GV) in -I mode

    cd $TOWDATA
    cp ../prep/cf_3686_GEHSVY_pstokes ./
    fourfit -pt -c ./cf_3686_GEHSVY_pstokes -b GV -P I ./1234/105-1800

#maybe also fourfit without -P I option to show the full set of pol-products
#note that most x-power is in the cross-pols, note that the dPAR is near
#90-degrees cycle through XX,YY,XY,YX:

    fourfit -pt -c ./cf_3686_GEHSVY_pstokes -b GV ./1234/105-1800

#########################################################################################
#STEP 11 - Construct control file for pStokes-I

#now we are going to change over to curie to do the post-correlation processing,
#using the 1-hour of data we correlated ahead of time

    ssh -XY barrettj@curie
    cd $TOWDATA
    ls

#copy over control file from last session
```

```
cp /media/barrettj/data/geodesy/3686/cf_3686_GEHSVY_pstokes ./cf_1234_firstpass

#edit to change label to VT9105, strip out Y-X offsets to avoid clutter

        nano ./cf_1234_firstpass

#show help

        vgoscf_generate.py -h

#now we can fire up vgoscf_generate.py, we should probably do this for only
#one baseline (GV) in demo (alternatively, use the pre-fitted data in the
#scratch directory), the following took about 36 minutes on curie:

        vgoscf_generate.py -n 12 -p ./cf_1234_firstpass G EHSVY ./

#could run just the GV baselines, this only takes about 4 minutes

        vgoscf_generate.py -n 12 -p ./cf_1234_firstpass G V ./

#The following takes about 1 minutes: <---- DO THIS!!
#run in the already fourfit-ed scratch directory (using the -w option).

        cd ./scratch/20190507-144536/1234
        cp ../../../cf_1234_firstpass ./
        vgoscf_generate.py -n 12 -p -w ./cf_1234_firstpass G EHSVY ./

#retrieve the resulting control files from scratch directory

        cd $TOWDATA
        cp ./scratch/20190507-144536/1234/cf_1234_* ./

################################################################################
#STEP 12 - quality check pc_phases

#now run summarize_report.py to take a look at ffres2pcpc
#show V_X.png and V_Y.png

        cd ./scratch/20190507-144536/1234
        summarize_report.py ./ffres2pcp-report-*
        eog ./V_X.png
        eog ./V_Y.png

################################################################################
#STEP 12 - quality check on phase/delay offsets

#now run summarize_report.py on fourphase output, and show distributions
#for station V (reasonable for 1 hour of data)

        summarize_report.py ./fourphase-report-*
        eog ./yx_delay_phase_offsets_V.png

#if time (now or later point) out the issues with station E
```

```
#and show fourfit plots for -PI , -P YY and -P XX on example
#fourfit -pt -b GE -P I -c ./cf_1234_GEHSVY_pstokes./105-1800/


################################################################################
#STEP 13 - Production fourfit

#now execute batch_fourfit.py to produce the pseudo-stokes fringe files
#the following took about 7 minutes for all stations the first time around:

    cd $TOWDATA
    batch_fourfit.py -n 12 -p ./cf_3686_GEHSVY_pstokes GEHSVY I ./

################################################################################
#STEP 14 - Inspect the output with phase_resid.py

#And create the phase residual plots with phase_resid.py
#(only do one baseline, takes too long to plot otherwise), show help:

    phase_resid.py -h
    phase_resid.py -n 12 -p -z dtec ./cf_1234_GEHSVY_pstokes GV I ./
    eog ./phresid_GV_I_cf_1234_GEHSVY_pstokes_1234_dtec.png

#now look at a plot from a full/complete experiment on GV baseline

    eog $VT9063/diagnostics/phresid_GV_I_cf_3685_GEHSVY_pstokes_3685_dtec.png

################################################################################
#STEP 15 - Generate proxy cable cal

#now generate the proxy cable calibration for station G (takes about 1 minutes)

    cd $TOWDATA
    pcc_generate.py -v 3 -f G ./

#now display the resulting meanbandelay plot

    eog ./pcc_datfiles/meanbanddelay.G.png

#show what one from a full experiment looks like

    eog $VT9063/pcc_datfiles/meanbanddelay.G.png


################################################################################
#STEP 16 - Select proxy cable cal

#run pcc_select.py to average delays for BCD:XY band-pols, show head of file

    pcc_select.py -e vt9105 -d ./pcc_datfiles/ -s G:BCD:XY
    head -10 ./pcc_datfiles/vt9105gs.pcmt.BCD.XY.dat

################################################################################
#STEP 17 - generate the database
```

```
#create a fake correlator report

        cd $VGOSDB
        echo "This is my report." > ./vt9105.corr
        vgosDbMake -t $VGOSDB/vt9105.corr -d "19APR15VG" $TOWDATA

#or create Db without report:

        vgosDbMake -d "19APR15VG" $TOWDATA


################################################################################
#STEP 18 - add the delay mode with vgosDbCalc

#check the status of the calcserver:

        sudo service calcserver status

#run vgosDbCalc

        vgosDbCalc ./19APR15VG/19APR15VG_V001_iMHC_kall.wrp

################################################################################
#STEP 19 - vgosDbProcLogs

#now do the vgosDbProcLogs first step to attach the station logs

        cd $STATION_LOGS
        mkdir -p ./2019/vt9105 && cd ./2019/vt9105
        wget ftp://cddis.gsfc.nasa.gov/vlbi/ivsdata/aux/2019/vt9105/*.log
        cd $VGOSDB
        vgosDbProcLogs -k log ./19APR15VG/19APR15VG_V002_iMHC_kall.wrp

#now append the proxy-cable calibration data

        cp $TOWDATA/pcc_datfiles/*pcmt* $STATION_LOGS/2019/vt9105/

        vgosDbProcLogs -zc -s GGAO12M -k pcmt \
        ./19APR15VG/19APR15VG_V003_iMHC_kall.wrp

#normal command would be something like below, but we have only
#generated PCC for GGAO12M

        vgosDbProcLogs -zc -s KOKEE12M -s WETTZ13S -s RAEGYEB -s GGAO12M \
        -k pcmt ./19APR15VG/19APR15VG_V003_iMHC_kall.wrp

################################################################################
#STEP 20 - Database ready!
```