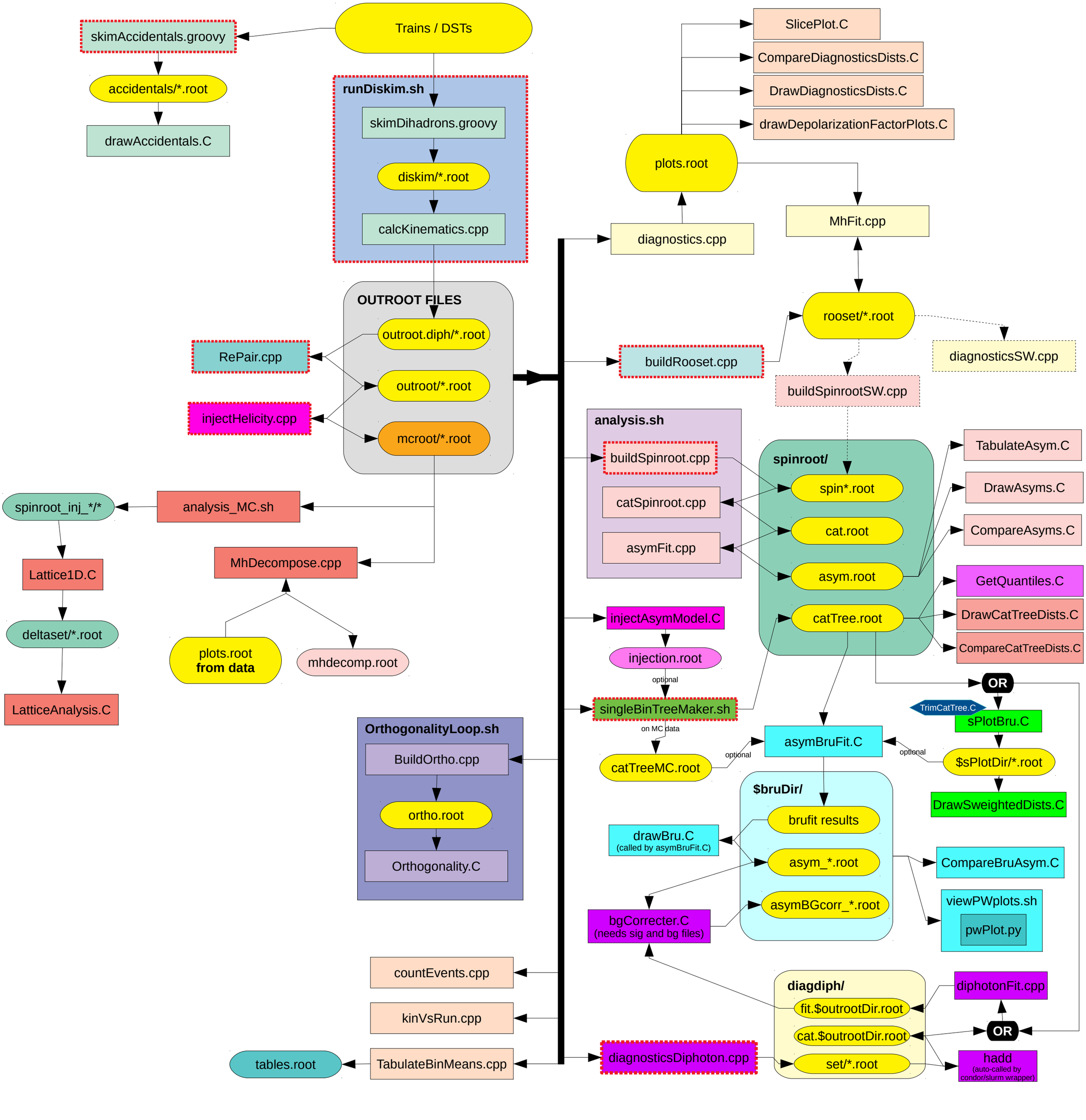

Trains / DSTs
skimAccidentals.groovy
accidentals/*.root
drawAccidentals.C
runDiskim.sh
skimDihadrons.groovy
diskim/*.root
calcKinematics.cpp
OUTROOT FILES
outroot.diph/*.root
outroot/*.root
mcroot/*.root
RePair.cpp
injectHelicity.cpp
SlicePlot.C
CompareDiagnosticsDists.C
DrawDiagnosticsDists.C
drawDepolarizationFactorPlots.C
plots.root
diagnostics.cpp
MhFit.cpp
rooset/*.root
diagnosticsSW.cpp
buildRooset.cpp
buildSpinrootSW.cpp
analysis.sh
buildSpinroot.cpp
catSpinroot.cpp
asymFit.cpp
spinroot/
spin*.root
cat.root
asym.root
catTree.root
TabulateAsym.C
DrawAsyms.C
CompareAsyms.C
GetQuantiles.C
DrawCatTreeDists.C
CompareCatTreeDists.C
spinroot_inj_*/*
analysis_MC.sh
Lattice1D.C
deltaset/*.root
LatticeAnalysis.C
MhDecompose.cpp
plots.root
from data
mhdecomp.root
injectAsymModel.C
injection.root
optional
singleBinTreeMaker.sh
on MC data
catTreeMC.root
OR
TrimCatTree.C
sPlotBru.C
$sPlotDir/*.root
DrawSweightedDists.C
asymBruFit.C
optional
optional
$bruDir/
brufit results
asym_*.root
asymBGcorr_*.root
drawBru.C
(called by asymBruFit.C)
CompareBruAsym.C
viewPWplots.sh
pwPlot.py
bgCorrecter.C
(needs sig and bg files)
OrthogonalityLoop.sh
BuildOrtho.cpp
ortho.root
Orthogonality.C
countEvents.cpp
kinVsRun.cpp
tables.root
TabulateBinMeans.cpp
diagnosticsDiphoton.cpp
diagdiph/
fit.$outrootDir.root
cat.$outrootDir.root
set/*.root
diphotonFit.cpp
OR
hadd
(auto-called by
condor/slurm wrapper)

Asymmetry Injection

injectAsymModel.C

injection.root

outroot/*.root

singleBinTreeMaker.sh … -j injection.root

catTreeMC.root

loop_asymBruFit.rb

asymBruFit.C

bruspin….inj*/

loop_checkInjectionFit.rb

checkInjectionFit.C

bruspin….inj*/*injectionTest.root
bruspin….pullresults/files.list

pullDists.rb

π0 Analysis Sequence

outroot/*.root

loop_singleBinTreeMaker.rb

hadd_makeBibendingCatTrees.rb
(use truncation for MC, for now…
since we also have sWeights)

catTree*.root

loop_TrimCatTree.rb

(if using reweighting)

catTree*.trimmed.root

loop_sPlotBru.rb

catTreeWeights/*/Tweights.root

splots/*/Tweights.root

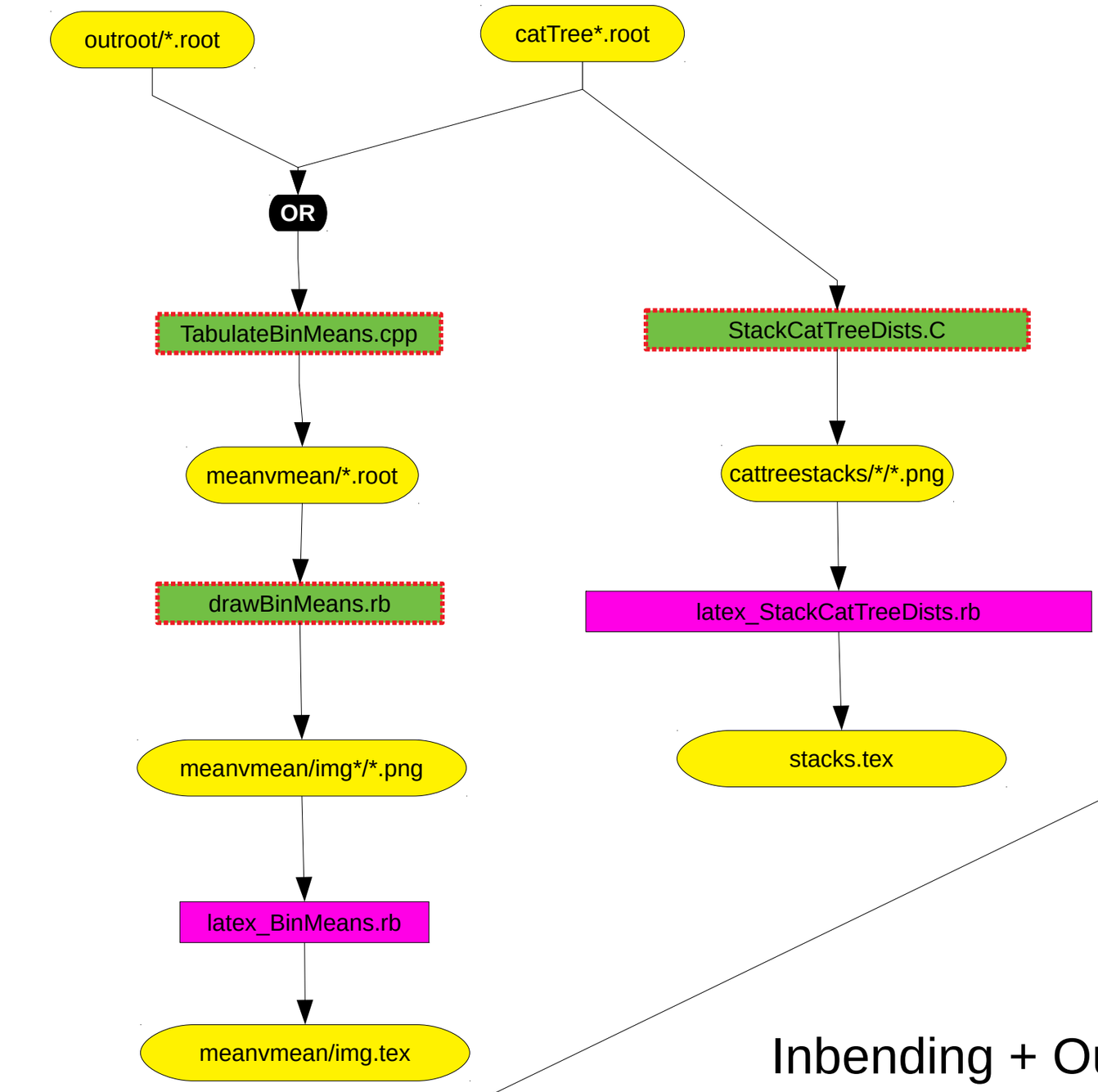
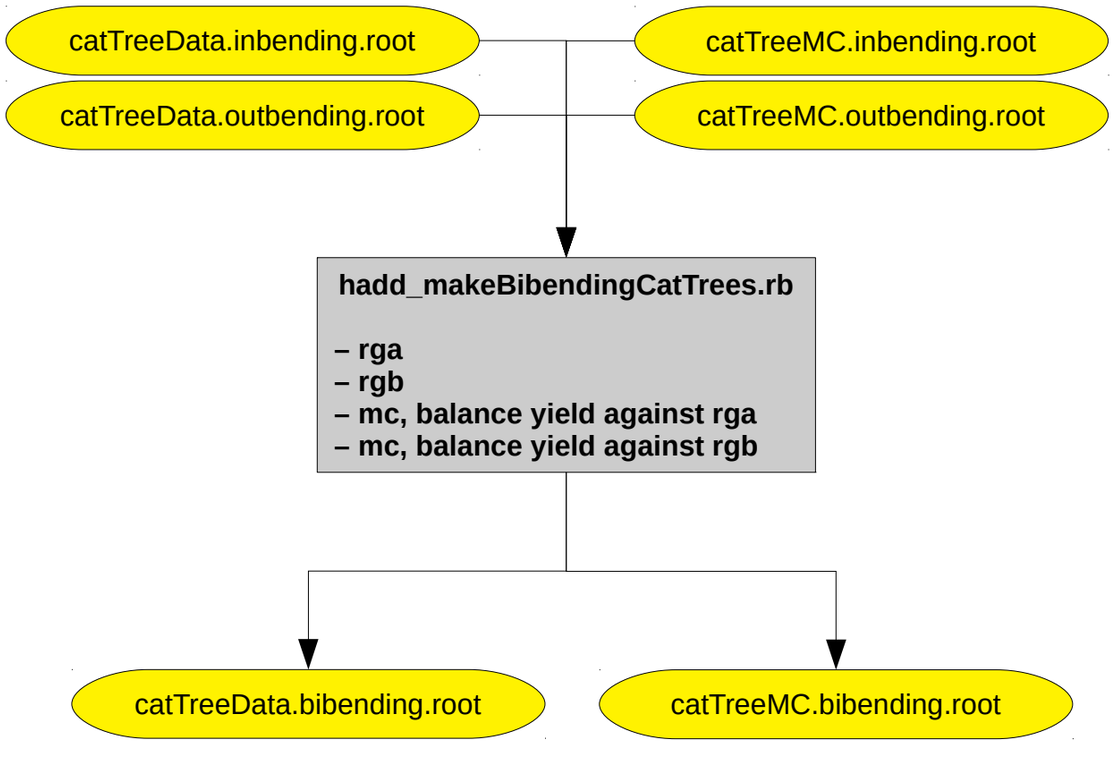
loop_DrawSweightedDists.rb

TODO: multiply these
weights by sWeights ??

finalFit.rb

finalPlot.rb

Coverage Plots



Inbending + Outbending = Bibending