Drell-Yan Analysis Procedure 2011

Completeness: 100%

Drell-Yan analysis Procedure

This twiki documents the most important steps of the Drell-Yan cross section measurement. It is intended to familiarize you with the technical aspects of the analysis procedure. At the end of each section there is a set of questions and exercises. Note, the software and samples evolve very frequently so some parts of the page will get obsolete soon!

Step 1: PAT/ntuples

- **Samples**
- MC samples change frequently (usually we have a new generation of MC samples every season). Even though GEN level stays the same reconstruction keeps getting improved from release to release, so it is recommnded to use the most up-to-date MC. Another things that constantly changes is the vertex information (tuned to data) and pile up.
- **MC**: 311 Monte Carlo samples (this list shows all the signals and backgrounds we consider, but it is already superseded by Summer11 42X samples)

  - Drell-Yan:
    - **/DYToMuMu_M-10To20_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/DYToMuMu_M-20_CT10_TuneZ2_7TeV-powheg-pythia/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
  - QCD
    - **/QCD_Pt-15to20_MuPt5Enriched_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/QCD_Pt-20to30_MuPt5Enriched_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/QCD_Pt-30to50_MuPt5Enriched_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/QCD_Pt-50to80_MuPt5Enriched_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/QCD_Pt-80to120_MuPt5Enriched_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/QCD_Pt-120to150_MuPt5Enriched_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/QCD_Pt-150_MuPt5Enriched_TuneZ2_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
  - DY tautau
    - **/DYToTauTau_M-20_CT10_TuneZ2_7TeV-powheg-pythia-tauola/Spring11-PU_S1_START311_V1G1-v2/GEN-SIM-RECO**
  - Dibosons
    - **/WWtoAnything_TuneZ2_7TeV-pythia6-tauola/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/WZtoAnything_TuneZ2_7TeV-pythia6-tauola/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
    - **/ZZtoAnything_TuneZ2_7TeV-pythia6-tauola/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**
  - Wmunu
    - **/WtoMuNu_TuneD6T_7TeV-pythia6/Spring11-PU_S1_START311_V1G1-v1/GEN-SIM-RECO**

Current latest MC generation is 42X Summer11

- DATA:

- We use **SingleMu and DoubleMu** Primary Datasets (PD), PromptReco (and ReReco when it becomes available)
- JSONs are constantly updated in this directory as we certify more and more data:
  - "Golden" /afs/cern.ch/cms/CAF/CMSCOMM/COMM_DQM/certification/Collisions11/7TeV/*Prompt/Cert_160404-161312_7TeV_PromptReco_Collisions11_JSON.txt
  - Calo-free /afs/cern.ch/cms/CAF/CMSCOMM/COMM_DQM/certification/Collisions11/7TeV/Prompt/Cert_160404-161312_7TeV_PromptReco_Collisions11_JSON_MuonPhys.txt

- Use DBS tool to retrieve info about the most up-to-date MC and DATA samples

- **Relevant software**: CMSSW_4_1_3: see the twiki page to get the information about CMSSW releases used for data taking as they evolve
  - Global tag
    - Data: GR_R_311_V2 for reprocessing and GR_P_V14 for prompt reco

        - MC: MC_311_V2, START311_v2
    - ElectroWeakAnalysis/Skimming, ElectroWeakAnalysis/DYskimming, UserCode
- **how to run:** PATtuple production: see the corresponding twiki, note, a lot of code in CVS is outdated (we keep updating it)

```
cmsrel CMSSW_4_1_3
cd CMSSW_4_1_3/src
cmsenv
addpkg ElectroWeakAnalysis/Skimming
cvs co -d ElectroWeakAnalysis/DYskimming -r V00-00-05
UserCode/Purdue/DYAnalysis/Skimming
scram b
cmsRun ElectroWeakAnalysis/DYskimming/PATTuple/test_cfg.py
```

Exercise 1:

Q1: Do you think the list of signal and background samples mentioned above is complete for the Drell-Yan cross section measurement in the region 15-600 GeV invariant mass? (You need to think not only about physics processes but also about kinematic region of the sample generated)

Q2: Using the DBS tool mentioned above or a command line equivalent find all the MC samples relevant for the Drell-Yan cross-section measurement for Summer11 production, example:

```
find file where file.tier=GEN-SIM-RECO and site=srm-dcache.rcac.purdue.edu
and dataset like=*DYToMuMu*Summer11*
```

Q3. (might be advanced, if you fails ask me) Port the above skimming code to 42X: you need this, because 41X and 42X releases are incompatible

Q4: Produce a PAT-tuple

More PAT-tuples can be found here: /store/user/asvyatko/DYstudy/dataAnalysis11//PATtuple/ (you can use signal and Data only for now)

Step 2: Ntuples

Since the event size even for PAT is quite big, we usually perform one more skimming step and store our data in form of ntuples. The input for ntuple-maker is PAT tuple (however, with some gymnastics you can run on GEN-SIM-RECO or AODSIM)

```
cd $CMSSW_RELEASE_BASE/src
cvs co -d Phys/DiMuonAnalyzer UserCode/Purdue/DYAnalysis/NtupleMaker
scram b
cd Phys/DimuonAnalyzer
cmsRun test/test_cfg.py
```

For mass ntuples submission you need to use batch farms. Use scripts provided in the /ntuple directory to do that:

```
./Ntuples.csh
./Submit.csh
```

Note: before submitting you need to change the input/output directory parameters and the global tag in the batchJob.pl script:

```
 # configure here
$isMC = 1; # 0 = MC, 1 = data
$trig = "HLT_Mu11/Data";
$ntpdir =  "/user/h/hdyoo/DYstudy/dataAnalysis10_39x/PATtuple/";
$outdir =  "/user/h/hdyoo/DYstudy/dataAnalysis10_39x/Ntuple/";
$workdir =
"/afs/cern.ch/user/h/hdyoo/scratch1/DYanalysis/NtupleProducer/src/Phys/DiM
uonAnalyzer/ntuples";
$gtag = "GR_R_39X_V6"; # data
#$gtag = "START39_V9"; # mc
```

to your desired directories and the latest global tag.

Exercise 2:

Q1: Try to produce ntuple

Hint: spot all the errors, they might be related to global tag, REDIGI version (for MC)

Q2: Find the invariant mass branch and inspect it

Step 3: Event Selection

Once the ntuples are ready, one can proceed to the actual physics analysis. The first step of every analysis is the event selection. Currently, we use the so-called cut-based approach to discriminate between signal and background. For more on event selection please read chapter 3 in the analysis note **CMS-AN-11-013**. Before starting to run a macro, set up the working area. Find all the necessary scripts in:

```
UserCode/Purdue/DYAnalysis/NtupleMaker/interface
UserCode/Purdue/DYAnalysis/NtupleMaker/ntuples
```

and precisely follow the recipe below preserving folder structure recommended in the recipe:

```
1. copy the Ntuples in rootfiles directory, splitting by trigger path used
at the level of skimming (as we might use a combination!): e.g.
rootfiles/HLT_Mu15/

2. modify the createChain.pl file

cd rootfiles
vim createChain.pl

change this to the full path to your rootfiles directory:
$dir = "/work/asvyatko/Work/DYanalysis/dataAnalysis10/analysis/rootfiles";
choose the trigger scenario you wish to use:
$trig = "HLT_Mu15"; // trigger name (directory name)

Run the shell script:
./Chains.csh

then all chain_* files will be created in the roorfiles directory.
```

- After you are done with creating chains (I assume you are in ./rootfiles directory) do:

```
cd ..
cvs co -d ControlPlots
UserCode/Purdue/DYAnalysis/AnalysisMacros/ControlPlots
cd ControlPlots
#note: for the sake of convenience add the alias for this directory in your
~/.profile, like:
#CONTROL_PLOTS_DIR=/whatever.is/the/absolte_path
#export CONTROL_PLOTS_DIR
```

Before running the macros, we need to fix few things which are changing frequently for our analysis:

- Mass range and binning:
  - for the early stage of 2011 analysis we keep the 2010 binning [15,20,30,40,50,60,76,86,96,106,120,150,200,600]
- Trigger selection:
  - See the presentation on event selection for 2011
  - Thus, for 2011 we consider a combination of Double muon trigger and a combination of single isolated muon triggers can be used as a cross-check. Use three following combinations:
    - HLT_Mu15, HLT_Mu24, HLT_Mu30
    - HLT_IsoMu15, HLT_IsoMu17, HLT_IsoMu24
    - DoubleMu6, DoubleMu7, Mu13_Mu8
- Offline selection: Baseline event selection has not changed compared to 2010 analysis, see
  - we will consider moving to PF muons and PF isolation: this study is in progress right now

To produce the invariant mass plot do use the analyse2.C macro, which calls the TSelector for the DY analysis (called EventSelector):

```
root -l analyse2.C
```

The macro allows to run on multiple cores.

By performing minor changes inside the EventSelector one can calculate the efficiency weighted invariant mass distribution (which is used to

estimate the corection factor as a function of invariant mass). Inside the EventSelector.C set

```
#define CORRECT_FOR_EFF true <- change to true if you want to use it for efficiency correction estimation
#define CORRECTION_TYPE "trig" <- choose the efficiency type recoid, iso or trig
```

To produce dimuon kinematic distributions run

```
root -l TransMomentum.C++
root -l Rapidity.C++
```

To produce other control plot (for all the event selection variables used in the analysis, as documented in the note), use:

```
root -l ControlPlots.C++
```

There are few macros that help us to optimize the cuts. These macros calculate the statistical significance and the uncertainty on the cross-section. Statistical significance is defined as :

S = N_sig/sqrt(N_sig+N_bkg) and normally determined from MC. As you can infer, it scales with luminosity as ~sqrt(Lumi). There are other definitions of significance used in the analyses sometimes (see for instance CMS-TDR). To run, check out just two additional macros (I assume you didn't leave ./ControlPlots directory)

```
cvs co
UserCode/Purdue/DYAnalysis/NtupleMacro/AsymmetricCutStudy/OfflineSelection
.C
cvs co
UserCode/Purdue/DYAnalysis/NtupleMacro/AsymmetricCutStudy/graphStats.C
root -l OfflineSelection.C++
root -l graphStats.C++
```

The first macro will create a txt file with an per mass bin values of signal and background. The second macro will histogram the output. These macros are adjusted to optimize the acceptance cuts, but with minimal changes it can optimize any other cut we use, and it is possible to change style to conform with the rest of plots in the note.

Note: root doesn not create output directories by itself so you should create a corresponding directory for output txt files like:

```
mkdir my_output_dir_for_macro1
```

Q1: Check data/MC agreement for each plot, look for discrepancies.

Checkpoint1 With the macros described above you should be able to *reproduce* following plots from the CMS-AN-11-013: 1,3-14, 17-29,51.

Note: for the 23,25-29 macros have different style and were produce with PU sample.

Note: plots 20-22 are reproducible by optimization macros but have different style.

Step 4: Acceptance

Another constituent of the cross-section measurement is the acceptance.

- Acceptance is determined using GEN level information, so that the generation of MC is not crucial
- Use the acceptance numbers documented in the note to save time: **CMS-AN-11-013**

Below are some additional details on the acceptance calculation which might be necessary if the procedure will change.

We use POWHEG NLO acceptance, see all the relevant formulas in this talk (the updated version), see slide18 for acceptance definitions and refer to earlier talks. Di\ue to the intrinsic discrepancies in the modeling between the POWHEG and FEWZ we correct the Z kinematics. For that, we extract the weight maps from POWHEG at NLO and from FEWZ at NNLO (to be more specific it is at NNLO below 40 GeV and NLO otherwise, as the effect of higher order corrections is negligible at the higher masses). The weight map is essentially the ratio of double inclusive cross-sections extracted from POWHEG and FEWZ per PT-Y bin (PT, Y refer to Z kinematics, which is identical here to dimuon kinematics, our final aim). Details on the re-weighting technique are also in the linked presentation.

How to run:

To setup area do:

```
cd $CMSSW_RELEASE_BASE/src
cvs co -r V00-09-11 SUSYBSMAnalysis/Zprime2muAnalysis (38x analysis)
# for 41x or higher, use V00-10-09
cmsenv
SUSYBSMAnalysis/Zprime2muAnalysis/setup.csh # This last command checks
                                            # out other packages and
                                            # executes scramv1 b to
                                            # compile everything.
cvs co UserCode/Purdue/DYAnalysis/AcceptanceAdam
#rearrange stuff a little bit
cp UserCode/Purdue/DYAnalysis/AcceptanceAdam/* .
```

First step: make the ntuples.

//description to be added - I will ask Adam. But actually we switched to Purdue ntuples so this section is obsolete....

Second step: make the cross section maps.

Inspect the countxsec.py script in the directory. What it does is following:

It takes hard interaction ntuple produced on the previous step, which contains the gen level information about the hard collision and produces the 2D histogram in the bins of dimuon Pt-Y for each mass bin, as well as 3 auxiliary histograms with mass spectrum pre-FSR (after acceptance cuts, marked with 'num'). The only part which is expected to be configured frequently is the mass, Pt, Y binning everything else is fixed. **(Note that right now the input can also be set to Purdue ntuple!!)**

To test the script locally on few hundred of events do:

```
python countxsec.py --s 1667 --n 20000000 --w 1.0 --sg
DYM1020/a2_2011/AcceptanceFromPAT_2011_mapcfg-pat_9_1_WaI/zp2mu_histos.root
```

To submit the batch script on full statistics, use the farmaout scripts and the local batch system:

```
cd /home/ba01/u112/aeverett/scratch_rcac/20112011/DYM1020 //(don't really
have to CD there but change all dirs accordingly)

farmoutAnalysisJobsPyList.sh bob $CMSSW_BASE
/home/ba01/u112/aeverett/scratch_rcac/20112011/countxsec.py --skip-srmcp
--submit-dir=$PWD/xsec04w1_new --input-files-per-job=1
--assume-input-files-exist --input-file-list=$PWD/inFileList.txt
--input-dir=$PWD/ --python-arguments="--s 3320.0 --n 9630633.0 --w 1.0 " #
--w 6.06" #9780633 in DBS but only 9630633 in a2_2011 directory
```

here, 'bob' is a random name for a batch job, inFileList.txt is a list of files to run on, it has the format:

```
a2_2011/AcceptanceFromPAT_2011_mapcfg-pat_10_1_Ky5/zp2mu_histos.root
a2_2011/AcceptanceFromPAT_2011_mapcfg-pat_11_1_l3o/zp2mu_histos.root
a2_2011/AcceptanceFromPAT_2011_mapcfg-pat_12_1_xFB/zp2mu_histos.root
a2_2011/AcceptanceFromPAT_2011_mapcfg-pat_13_1_hlY/zp2mu_histos.root
a2_2011/AcceptanceFromPAT_2011_mapcfg-pat_14_1_Ufv/zp2mu_histos.root
....
```

i.e. the absolute path to file from withi DYM1020 directory. The contents of the directory can look something like this:

drwxr-xr-x 71 asvyatko phys 10240 Nov 1 13:02 a2_2011
-rwxr-xr-x 1 asvyatko phys 6257 Nov 2 12:53 cmsRun.sh
-rwxr-xr-x 1 asvyatko phys 17621 Nov 2 12:52 farmoutAnalysisJobsPyList.sh
~~rw-rr-~~ 1 asvyatko phys 4752 Nov 2 12:48 inFileList.txt
drwxr-xr-x 2 asvyatko phys 2048 Nov 2 12:33 xsec04w1_new

The submission directory should not exist, otherwise you get an error.

Note: you obviously do not want to recalculate the weights, numbers of events for each stepwise DY sample. For Summer11 the numbers are already available here: /UserCode/Purdue/DYAnalysis/AcceptanceAdam/commands_XSEC.txt

But for Fall11 I will have to recalculate it and update the corresponding file.

Third step: make the weights

To make weights one needs to run following:

```
python makeweights.py
```

The weights referred here are the FEWZ-POWHEG cross section ratios per Pt-Y bin (see one of the Adam's presentations). What the script does is essentially following: it takes the POWHEG Pt-Y maps produced on the previous step, takes the FEWZ Pt-Y maps (which are produced by Alexey and usually latest are found here: https://twiki.cern.ch/twiki/bin/view/CMS/EWKDrellYan2011, the file starting with map2D_*.root), and just divides it bin by bin using the my_divide function (this is a clopper-pearson divide).

Complications: there is a set of parameters and input files which is configures inside the script.

First of all POWHEG input files:

```
## ADAM for NNLO
#Get the files with POWHEG maps with fine Pt binning
fileIn10fine=ROOT.TFile("DYM1020/xsec04w1_newFine/powheg_xs_full_04_2011.r
oot")
#coarse Pt binningfileIn10  = ROOT.TFile(
"DYM1020/xsec04w1_new/powheg_xs_full_04_2011.root")
#fine Pt
binningfileIn20fine=ROOT.TFile("DYM20/xsec04w1_newFine/powheg_xs_full_04_2
011.root")
#coarse Pt binningfileIn20  = ROOT.TFile(
"DYM20/xsec04w1_new/powheg_xs_full_04_2011.root")
fileIn200 = ROOT.TFile(  "DYM200/xsec04w1_new/powheg_xs_full_04_2011.root")
fileIn500 = ROOT.TFile(  "DYM500/xsec04w1_new/powheg_xs_full_04_2011.root")
fileIn1000 = ROOT.TFile("DYM1000/xsec04w1_new/powheg_xs_full_04_2011.root")
```

There are multiple input files, because we use STEPWISE DY samples, they are split in generator mass. On the other hand, we fix the Pt-Y binning on the previous step, so that if we decide to play around with binning we need to provide a different input file (you can see fileIn10fine and fileIn10 for instance). The parameter **finePtBins** controls the number of first mass bins which will use fine binning.

Secondly, FEWZ input files: this are also hardcode inside the script, because multiple versions exist (differing by Vegas integration precision and NNLO/NLO order). The parameter **nnloInBins** controls the amount of first mass bins in which we have NNLO.

The output of the script is DYMoutput/weights_stepwise_precision10-5_fine12.root - the file with histograms with weights and errors.

Fourth step: make the corrected acceptance distributions (finally!!!)

Inspect the countcorracc.py script in the directory. What it does is following:

//add

To test the script locally on few hundred of events do:

```
python countcorracc.py --r
DYMoutput/fewz_powheg_weights_stepwise_2011_fine12.root --o
fewz_powheg_corracc_2011 --n 5.78216 --s 3320.0 --e 9630633.0 --l 15 --h 20
--sg
DYM1020/a2_2011/AcceptanceFromPAT_2011_mapcfg-pat_9_1_WaI/zp2mu_histos.root
```

Note that again one has to run a separate job on each of the stepwise DY samples. And the weights, event numbers and cross section passed as an argument would have to be different for each sample.

To submit a batch job do:

```
 cd /home/ba01/u112/aeverett/scratch_rcac/20112011/DYM1020

farmoutAnalysisJobsPyList.sh bob $CMSSW_BASE
/home/ba01/u112/aeverett/scratch_rcac/20112011/countcorracc.py
--skip-srmcp --submit-dir=$PWD/corracc_stepwise_coarse7b
--input-files-per-job=1  --assume-input-files-exist
 --input-file-list=$PWD/inFileList.txt --input-dir=$PWD/
--python-arguments="
--r
/home/ba01/u112/aeverett/scratch_rcac/20112011/DYMoutput/fewz_powheg_weigh
ts_stepwise_2011_coarse7.root
--o fewz_powheg_corracc_2011 --n 5.78216 --s 3320.0 --e 9630633.0 --l 15
--h 20"
```

The tricky part about this script is:

//this is still old

One needs to configure pT and eta cuts for each muon in the script. This is done very easily in two places:

```
if input_is_MC:
    from UserCode.Examples.inclusiveMuonPlotsGENSIM_cfi import
makeInclusiveMuonPlots;
    commonInputs = cms.PSet(
        muons     = cms.InputTag('cleanPatMuonsTriggerMatch'),
        particleSrc = cms.InputTag('prunedGenSimLeptons'), #genParticles
        dilepton_src = cms.InputTag('dimuons'), #('dimuons'), #('dyz'),
        primaryVertices = cms.InputTag("offlinePrimaryVertices"),
        weight = cms.untracked.double(1.0),
        eta_acc1 = cms.untracked.double(2.4),
        eta_acc2 = cms.untracked.double(2.4),
        pt_acc1 = cms.untracked.double(15.0),
        pt_acc2 = cms.untracked.double(10.0),
        mother = cms.untracked.int32(23),#Z: 23, ZPrime: 32
        daughter = cms.untracked.int32(13),
        daughterStatus = cms.untracked.int32(1),
        )
```

And here:

```
loose_cut = 'isGlobalMuon && isTrackerMuon && ' \
            'pt > 10. && ' \
            'abs(eta) < 2.4 && ' \
            'abs(dB) < 0.2 && ' \
            '(isolationR03.sumPt + isolationR03.hadEt) / pt < 0.15 && ' \
            'globalTrack.hitPattern.numberOfValidTrackerHits > 10 && ' \
            'globalTrack.hitPattern.numberOfValidPixelHits > 0 && ' \
            'globalTrack.hitPattern.numberOfValidMuonHits > 0 && ' \
            'numberOfMatches > 1 && ' \
            'globalTrack.normalizedChi2 < 10 ' # + trigger_match
```

To configure the trigger path for trigger match:

```
trigger_match = ' && (' \
                '!triggerObjectMatchesByPath("HLT_Mu15_*").empty()' \
                ')'
```

To calculate the acceptance as a function of invariant mass:

```
 python AcceptanceFromPAT_a2_mapcfg.py
```

Next, produce histograms for invariant mass and Z/lepton kinematics you will use to calculate the acceptance:

```
 root -l macroAcceptance.C
```

To draw the plots execute:

```
root -l macroDrawHwidongAcc.C
```

Weight map producer uses the FEWZ acceptance as the input, find all the macros here: **/UserCode/ASvyatkovskiy/FEWZ_scripts,**

to work one needs to do:

```
cd $RCAC_SCRATCH/  #better to do it on scratch
cp -r /home/ba01/u115/asvyatko/TEST_FEWZ/FEWZ_2.0.1.a3/
YOUR_FEWZ_WORKING_DIR
cvs co -d TMP /UserCode/ASvyatkovskiy/FEWZ_scripts
cp TMP/* .
rm -r TMP
```

To produce the cross section in mass bins run

```
python nu_createInputFEWZ_mod_multiple.py
```

Note: before running you need to adjust the order of calculation and the input/output directories and also the PDF set you want to use, which is all done in the top of the script:

```
 #input: binning file, pdf sets, order, number of processors and path to
working dir
mass_cuts = open('input_massesPlus.txt','r')
pdfsets = ['MSTW2008NLO'] #['CTEQ66','CTEQ10','NNPDF20','MSTW2008NLO']
order = 'NLO'
nprocs = 1
path_to_wd = '/scratch/scratch95/a/asvyatko/NU_FEWZ/WorkingDir_NLO_400Plus'

masses = mass_cuts.readlines()
mass_cuts.close()
pattern = r'(\d+\.\d*|\.\d+)(d0)'
pattern2 = r'(\'\s\d?)'
pdf_pattern = r'(CTEQ66|CTEQ10|NNPDF20|MSTW2008.*O)'
```

to get the output of the jobs use corresponding get-output scripts:

```
python nu_getOutputFEWZ_multiple.py
```

Which requires analogous adjustments before use:

```
#input: binning file, pdf sets, order, number of processors and path to
working dir
mass_cuts = open('input_massesPlus.txt','r')
pdfsets = ['MSTW2008NLO'] #['CTEQ66','CTEQ10','NNPDF20','MSTW2008NLO']
order = 'NLO'
nprocs = 1
path_to_wd = '/scratch/scratch95/a/asvyatko/NU_FEWZ/WorkingDir_NLO_400Plus'
run = ''
acc = '' #acc21_, acc24_
```

After the maps are ready and saved in the root file, you can proceed with corrected acceptance calculation. Before that, review the maps to check if there is nothing strange:

```
root -l Map2D.C
```

Q: Analyze 2D histograms for maps

If you do not have time to produce the maps, you can use what we have, kindly read the description in the README file and use the root and txt files on the following webpage:

https://test-project-aeverett.web.cern.ch/test-project-aeverett/Acceptance02/

If everything looks reasonable (no very large weights, no pronounced asymmetries), continue with corrected acceptance (before doing that make sure to carefully review this talk by Adam):

```
python countxsec.py DYM1020_single_tuple.root DYM20_single_tuple.root #you
can find test root files here

#/store/user/asvyatko/DYstudy/AN11_013_archives/TestAcceptanceAdam/
                                                                    #or
use the PAT tuple produce on the step1
python makeweights.py powheg_xs_full.root fewz_xs_full.root
python countcorracc.py
--sg=DYM1020_single_tuple.root,DYM20_single_tuple.root -r
fewz_powheg_weights.root
```

Each next step uses the output of the previous step. Firstly we calculate the PT-Y dimuon kinematics map, then we calculate weight for FEWZ-POWHEG reweighting and finally we calculate the corrected acceptance. Note: the k factors are equal to 1 by default but should be readjusted in case one uses NLO FEWZ maps (rather than NNLO)

Finally, there are various plots used to estimate the FEWZ-POWHEG discrepancy for various trigger scenarios as well as NNLO-NLO discrepancy for within a given generator and a given PDF set. These plots can be reproduced using the following macros:

```
cp /UserCode/Purdue/DYAnalysis/AnalysisMacros/ControlPlots/Acceptance.C
$CONTROL_PLOTS_DIR  #Acceptance macro isn't in current dir :-/ , copy it
cd $CONTROL_PLOTS_DIR
root -l Acceptance.C
```

inside this macro, you can uncomment what you need to plot and also adjust the input PDF sets as well as orders.

Checkpoint2 With the macros and scripts described in the step4 section you should be able to *reproduce* following macros from the CMS-AN-11-013: 2,30-38,61-63 and tables: 5-10, 21-24

Step 5: Efficiencies

The details on the factorization and the application of correction factors are documented here , and can be found in this talk. With the current factorization scheme we measure four following efficiencies:

- Trigger, Reconstruction+ID, PF isoloation, tracking:
  - We use the officiela TagAndProbe package
- **How to run (on top of CMSSW 425 or later):**

```
addpkg CommonTools/ParticleFlow V00-02-07
addpkg CommonTools/RecoAlgos V00-03-13
addpkg DataFormats/PatCandidates V06-04-18
addpkg MuonAnalysis/MuonAssociators V01-13-00
addpkg MuonAnalysis/TagAndProbe HEAD
addpkg PhysicsTools/Configuration V00-10-16
addpkg PhysicsTools/PatAlgos V08-06-38
addpkg PhysicsTools/PatExamples V00-05-22
addpkg PhysicsTools/SelectorUtils V00-03-17
addpkg PhysicsTools/TagAndProbe HEAD
addpkg RecoMuon/MuonIdentification V01-19-00
cvs co -d TagAndProbe UserCode/ASvyatkovskiy/TagAndProbe
```

- The procedure goes in two steps:
  - T&P tree production -> rerun seldom (ideally once), it depends only on the definitions of the tag and probe

```
cd TagAndProbe
cmsRun tp_from_aod_Data_newofficial.py
```

- If you haven't produce TP trees you can always use the ready ones located there:

```
/store/user/asvyatko/DYstudy/TagProbeTrees/
```

- fitting: separate job for trigger and all the muonID related efficiencies -> reran frequently and usually interactively (change binning, definitions)

```
cmsRun fitMuonID_data_all_2011.py
```

- All the latest macros/configs can be found here: **UserCode/ASvyatkovskiy/TagAndProbe**
- Isolation: RandomCone - currently, code is private and not possible to use.

After familiarizing yourself with the TagAndProbe package, you need to produce the muon efficiencies as a function of pT and eta. You do not need this in the analysis, but rather to understand if everything you are doing is correct. After you are done with that, produce the 2D efficiency pT-eta map (it is alredy produced in one go when running fiMuonID.py). To do that use the simple root macros (adjust i/o, not user friendly yet!):

```
root -l idDataMC_4xy.C
root -l triggerMuonDataMC_4xy.C
```

And to produce 2D efficiency maps and correction factors do:

```
root -l perBinTable.C
```

The final step here is to produce the efficiency as function of invariant mass and the efficiency correction factor as a function of invariant mass.

```
cvs co UserCode/Purdue/DYAnalysis/AnalysisMacros/Correction
root -l efficiencyMass_newTmp.C
root -l correctionMass_newTmp.C
```

Note: you need to produce all the correction 2D maps on 2 previous steps, if you haven't succeeded you can use what we used for publication, txt files are located here:

```
/store/user/asvyatko/DYstudy/AN11_013_archives/EfficiencyCorrectionsTAndP
```

Checkpoint3 With the macros describe in the step5 section it is possible to reproduce the following plots from the CMS-AN-11-013 note: 15-16, 39-42 and tables 11-12

Note: plot 40 was produced with LKTC method, code for which is currently not public and not possible to be retrieved from the authors. Currently (2011 data) the result is consistent with that obtained with Tag-And-Probe.

Step 6: Background estimation

QCD data driven background estimation

There are various methods employed to estimate the QCD background in a data-driven way (QCD is currently the only background estimated not from MC). The most important are the template fit method and the weight map method: carefully read chapter6 of the CMS-AN-11-013 for more details on the methods.

Reweighting method. First of all, read the quick description (in addition to what is written in the note) and also see this presentation .

```
cd $CONTROL_PLOTS_DIR
cd ..
cvs co -d ControlPlots
UserCode/Purdue/DYAnalysis/AnalysisMacros/QCDEstimation
```

There are few steps in this method. First of all, create a pT-eta weight look-up table indicating probability of a muon to be isolated as a function of muon pT-eta:

```
cd ControlPlots
root -l WeightMapFiller.C
```

The next step is to view the map, and to test it on the sample of dimuons and single muons:

```
root -l testWeightMapDouble.C
root -l testWeightMapSingle.C
```

Other methods used for the QCD background estimation in the note are the SS/OS pair method and template fit method (carefully read the note on the description!). For the SS-OS method, which uses the discriminative power of the isolation variable, considering classes of events having 2, 1or 0 isolated muon. You can get the plot by running:

```
cd $CMSSW_RELEASE_BASE/src
cvs co UserCode/ASvyatkovskiy/QCD_Background
cd UserCode/ASvyatkovskiy/QCD_Background
root -l bgSS_OS.C
```

As for the template fit method:

```
root -l
.L auto_fits.C; fitAll()
#It needs input parameters - the important ones are
#bool etaBelow1_2 = false;
#bool fitFirstMu = true;
#bool singleMu = false;
```

Note: the original input files can be found at:

```
/store/user/asvyatko/DYstudy//AN11_013_archives/StoyanBackground_SSOS_TemplateFit/
```

Checkpoint: this macros will allow one to reproduce the plots 45-48 from the note as well as tables 13-15 from the note

ABCD method

We estimate QCD background using ABCD method in order to improve our systematic uncertainty on the background estimation. ABCD method is very simple.

1) choose 2 variables: assume two variables are independent

2) assume the fraction should be same if there is no correlation: N_A / N_B = N_C / N_D

3) In our study, use two variables: sign of muon pair, muon isolation

4) QCD fraction in each region has a dependence. We produce the correction factor for each region: B, C, D

5) Produce N_B, N_C, N_D from data sample, and estimate N_A from them at the end (applying the correction factors)

In UserCode/Purdue/DYAnalysis/AnalysisMacros/ABCDmethod

QCDFrac.C: to produce correction factors for each region

ABCD2vari.C: to produce the ABCD results. The correction factors from the QCDFrac.C are plugged in this macro as an input.

ttbar data driven background estimation

We employ the so-called e-mu data driven background estimation method. See the following comprehensive talk for more details on the method. Currently the procedure to apply this method consists of 2 steps:

1) produce the root files with histograms

2) run the macros on the root files produced

For both steps one needs to check out the following tags:

```
V00-05-00    MuonAnalysis/Examples
V01-13-00    MuonAnalysis/MuonAssociators
V01-01-11    RecoVertex/VertexTools
V00-05-00    SHarper/HEEPAnalyzer
V00-11-00    SUSYBSMAnalysis/Zprime2muAnalysis
V00-03-00    UserCode/Examples
```

The highleted tags are important for step2).

Following is the description of how to produce the root files.

The mother script file is Zprime2muAnalysis/test/DataMCSpectraComparison/histos.py

Instructions related to this script file are at
https://twiki.cern.ch/twiki/bin/view/CMS/Zprime2muAnalysisDataMCSpectraComparison

The short instruction is this:

```
python histos.py submit testing no_data
```

or when you are ready

```
python histos.py submit
```

Wait for root files to be done. Currently it is configured to have histograms with selection marked 'VBTF' as what we have in DY2011.

Below I describe the step2 in detail. Check out addtional macros, and copy them to your working directory:

```
cvs co UserCode/Purdue/DYAnalysis/AnalysisMacros/TTbarEstimation
cp UserCode/Purdue/DYAnalysis/AnalysisMacros/TTbarEstimation/*
cd SUSYBSMAnalysis/Zprime2muAnalysis/test/DataMCSpectraComparison
```

Make sure the paths to datafiles inside the macros are pointing to the location of the root files you have produced. To produce the control plots for emu and mumu mass spectra use

```
python emu_basichists.py
```

To produce the correction factors run:

```
python emu_corrfactors.py
```

And finally, the MC expectation vs. data driven method prediction plots are produced with:

```
python emu_prediction_plots.py
```

A good agreement between data and MC for both the mumu and emu spectra is necessary for a method to work reliably.

Step 7: Unfolding

Unfolding is applied to correct for migration of entries between bins caused by mass resolution effects (FSR correction is taken into account as a separate step). For use in the Drell-Yan analysis, the choice for unfolding is matrix inversion. Provides a common interface between channels for symmetry and ease in combination and systematic studies.

To do any unfolding with MC, this requires 3 things:

- Producing the response matrix
- Making the histogram of measured events
- Making the true histogram (clearly not used/available when unfolding data)

First, one can do some exercise, for that use script that demonstrates how the unfolding/fore-folding object works.

```
cvs co UserCode/kypreos/drellYan2010/unfolding
cd UserCode/kypreos/drellYan2010/unfolding/
source setup.sh
```

```
root -l test/testExpo.C++
```

To get back the pulls:

```
root -l test/testPulls.C++
```

The macros in the note are produced with the following:

```
 cvs co /UserCode/Purdue/DYAnalysis/Unfolding
```

1. To rpoduce the response matrix:

```
root -l unfoldingObs.C
```

2. To produce the unfolded yield plot do

```
root -l yield.C
```

Checkpoint7 with this macros one should be able to reproduce the plot 49-50 from the note and Tables 17-18 (note, the table 18 uses the background yield result from the background section)

Step 8: FSR correction

The effect of FSR is manifested by photon emission off the final state muon. It leads to change of the dimuon invariant mass and as a result a dimuon has invariant mass distinct from the propagator (or Z/gamma*) mass.

For our analysis we estimate the effect of FSR and the corresponding correction by estimating the bin-by-bin correction in invariant mass bins. Which is done by comparing the pre-FSR and the post-FSR spectra. The pre-FSR spectrum can be obtained by requiring mother of muon to be Z/gamma*, post FSR spectrum is when the mother is whatever.. The corresponding plots in the note are: 52-55 they all can be calculated with the information avaialble in the ntuple using

```
cd $CONTROL_PLOTS_DIR
root -l InvMassFSR.C++
```

To get the FSR histograms one needs to turno on **calculateFSR** flag on.

Checkpoint: this macro will allow one to get plots 52-55 from the note

Step 9: Systematic uncertainty estimation

There are various sources of systematics affecting our analysis: the PDF, theoretical modeling uncertainty, efficiency estimation uncertainty, background estimation, unfolding etc.

For the **background estimation**, with the data driven method we estimate the systematic uncertainty as the difference between the result obtained with the method and that

expected from MC per mass bin. Corresponding numbers are obtained with the emu_prediction_plots.py

macro (see the recipe in the step 6 section).

PDF uncertainty estimation. The recipe for the method currently used (step by step).
Reweight the PDF using the current existing MC samples as implemented in CMSSW. First, check out the necessary packages:

```
scramv1 p CMSSW CMSSW_4_2_3
cvsco -r CMSSW_4_2_3 ElectroWeakAnalysis/Utilities
```

then replace the LHAPDF library as described here to the current up-to-date one:
/afs/cern.ch/cms/slc5_amd64_gcc434/external/lhapdf/5.8.5-cms2/share/lhapdf/PDFsets
or you can directly change in:
CMSSW_4_2_3/config/toolbox/slc5_amd64_gcc434/tools/available/lhapdffull.xml
with above path:

```
touch $CMSSW_BASE/src/ElectroWeakAnalysis/Utilities/BuildFile.xml
cmsenv
scramv1 b
cd ElectroWeakAnalysis/Utilities/test
```

then change the input file in PdfSystematicsAnalyzer.py and run:

```
cmsRun PdfSystematicsAnalyzer.py
```

With the up-to-date LHAPDF, one can use CT10, MSTW2008*, CTEQ66, NNPDF2.0, and other PDF sets.

Efficiency estimation uncertainty. The current method for efficiency estimation in the DY analysis is following: we estimate the MC truth efficiency and then we apply the efficiency correction map (Pt-eta) extracted using the data-driven tag and probe method applied to data and MC to weight the MC events. The systematic uncertainty associated with the Tag-and-Probe efficiency estimation is due to line-shape modelling, the difference between fit and counting and due to the binning. The two first are calculated inside the macros described in Step5. The binning systematic uncertainty is estimated using the following macro:

UserCode/Purdue/DYAnalysis/AnalysisMacros/Correction/correctionMass_systematics.C

it takes as input the root files having the histogram with efficiency correction as a function of invariant mass with two binnings (to estimate the binning uncertainty), the other sources of uncertainty are also accessed.

Step 10: Plotting the results

The main result of the measurement is the cross-section ratio or r (and R) shape. We distinguish R and r shapes (see the note chapter9 for details on the definition and also see Figures 64). The figure 64 shows the shape R for theory and measurement (for two independent trigger scenarios). It relies on the theoretical cross-section measurement (1-2GeV bin), the final numbers for acceptance correction and also the final numbers for cross-section measurement. To give a clearer feeling of what this plot depends on I name the tables that are used to produce the number in the plot 64:

> Table 21-24: Theoretical predictions
>
> Tables 25-26: Measurement
>
> Table 5-10: Acceptance-efficiency corrections

To run the code one simply needs:

```
cd $CMSSW_RELEASE_BASE/src
cvs co UserCode/Purdue/DYAnalysis/AnalysisMacros/GautierMacro
cp UserCode/Purdue/DYAnalysis/AnalysisMacros/GautirMacro/*
$CONTROL_PLOTS_DIR
cd $CONTROL_PLOTS_DIR
root -l theory_plot.C++
```

Use Gautier style macros to get the same plots with different style:

```
root -l DY.C
root -l plot.C
```

To get all the up to date values for the shape r/R use:

```
cvs co UserCode/Purdue/DYAnalysis/AnalysisMacros/ShapeR./shapeDY.make
./shapeDY
```

Among the requirements to style of the results presented is to put the measurement point to the weighted position (i.e. the location of the point inside the bin makes the integral over sub-bins equal from both sides). The following macro can be used to calculate these positions do in root:

```
.L compare_r.cc;
compare_r();
```

Useful links

A lot of intersting information can be retrieved from the Zprime JTERM SHORT and LONG exercises (which are constructed along the same lines as this tutorial).