How to run ALPGEN in CMS from Event generation to GEN collections

Maurizio Pierini

in collaboration with
Maria Spiropulu, Thiago Tomei(*)

CERN PH
Saõ Paulo
(*) in charge for the ALPGEN & ALPGENInterface in CMSSW

Alpgen@CMS in a nutshell

Alpgen generates events at pp and $\bar{p}p$ collision through

- Matrix Element calculation
- Events generation according to phase space
- Matching of partons to shower (through Pythia interface)

Generation is performed in three steps

- Generation of weighted events
- Unweighting of the events and decay of unstable particles (W, Z, H, top)

Standalone Alpgen Code

- Parton shower in pythia

CMSSW through GeneratorInterface/AlpgenInterface

What ALPGEN package looks like

```
>ls ALPGENv213/

2Qlib          4Qwork         ft90V.tar.gz   Njetlib        pylib          topwork
wcjetwork      wphjetwork     zjetlib        2Qphlib        alplib         herlib
Njetwork       QQhlib         validation     wjetlib        wphqqlib       zjetwork
2Qphwork       compare        hjetlib        phjetlib       QQhwork        vbjetlib
wphqqwork      zqqlib         2Qwork         compile.mk     hjetwork       phjetwork
vbjetwork      wjetwork       wqqlib         zqqwork        4Qlib          DOCS
Makefile       prc.list       toplib         wcjetlib       wphjetlib      wqqwork
```

The libraries (don't touch them)

The working directories (where you compile and run the code)

The ALPGEN code (a black box for the user)

The interface to the parton showers

The documentation

Alpgen in CMS CVS

The generation

Pre-compiled executables (v2.12) used for CSA07
With the customized parton selection (pT bins for W and Z, ...)
Good to start with (no issues with compilation) but better use the official latest code for new productions
`$CMS_PATH/sw/slc4_ia32_gcc345/external/alpgen/213/bin`

CSA07 grids

`GeneratorInterface/AlpgenInterface/data/`

CSA07 configuration files (for cmsGen)

`Configuration/Generator/data/`

The parton-shower interface (PYTHIA ONLY)

`GeneratorInterface/AlpgenInterface`

The Alpgen info products (to save in the ROOT files also the weighted and unweighted events)

`SimDataFormats/HepMCProduct/interface/AlpgenInfoProduct.h`
`SimDataFormats/HepMCProduct/interface/AlpWgtFileInfoProduct.h`
(see also `https://twiki.cern.ch/twiki/bin/view/CMS/AlpgenInfoProduct`)

How Alpgen works

WEIGHTED event production

INPUTS

A configuration file to generate weighted events

OPTIONAL: a grid representing the population of the phase space can be provided

- n warmup runs of N events each. The output grid of the cycle i is used as input for cycle i+1
- the main run (of M events, in general >> N)

UNWEIGHTING run

INPUTS

A configuration file to unweight the events

specifying the decay modes for heavy particles (W, Z, top, H, ...)

- take the input of the previous step and produce unweigthed events

MATCHING

INPUTS

A CMSSW cfg file for the ALPGEN interface

- shower through PYTHIA, match partons to jets and produce the GEN

Step 1: The Input File for weighted events (I)

```
1           ! imode
z2j         ! label for files
0 ! start with: 0=new grid, 1=previous warmup grid, 2=previous generation grid
10000 2    ! Nevents/iteration,  N(warm-up iterations)
100000       ! Nevents generated after warm-up
```

imode sets the running mode:

imode 0 run ALPGEN in weighing mode, no events given as output
imode 1 run ALPGEN in weighting mode, events sample produced as output
imode 2 run ALPGEN in unweighting mode (after imode was done)
imode 3 print parameters options and defaults, then stop
imode 4 write the par.list file (complete list of process parameters)
imode 5 write the prc.list file (complete list of parameetrs and inputs)

The label specifies the name of the output files (e.g. z2j.wgt, z2j.unw, z2j_unw.par,...)

The third line specifies where to get the input grids from
(the grid has to be named as specified by the label, the input grid is saved as .old)

The other two lines specify the required number of events for warm-up+run
if previous warmup or generation is used, set to 0 the number of warm-up cycles

Step 1: The Input File for weighted events (II)

Beam Configuration

```
ih2 1        ! 1 for proton-proton, -1 for proton-antiproton
ebeam 5000   ! energy of the beams in CM frame
```

Process Configuration

```
ndns    5    ! Choose PDF, se par.list for details)
iqopt   3    ! Q = qfac*sqrt(m^2+pt^2) (other choices in par.list)(*)
qfac    1    ! scaling factor for fact. and renorm. scale Q
ickkw   1    ! 1/0 to enable/disable matching
ktfac   1    ! scale factor for ckkw alphas scale
njets   2    ! number of LIGHT PARTONS in the ME calculation
```

Kinematic Configuration

```
ptjmin   15    ! minimum pT for light partons
plmin     0    ! minimum pT for leptons
etajmax   5    ! maximum eta for light jets
etalmax   5    ! maximum eta for leptons
drjmin    0.7  ! minimum ΔR between partons
izdecmod  4    ! force Z->ll  (other choices in par.list) (*)
```

Other configuration cuts in par.list (*) process dependent

Step 2: The weighted events run

Before running, you need to set the warm-up cycle such that statistical fluctuations are reduced

Suggestions

It is better to do a test run to produce the grids (saving CPU time by doing the warm-up just once)
It is better to use the grids from the full test run (more events than the warm-up)
It is better to plan the production (N to ask to have some luminosity)
It is better to split productions in a set of O(1000) output files
(each file is a single cmsRun process)

To populate the grids

At least 3 warm-up cycles for njets = 0
Add a cycle for each additional parton you require
ask at least 1E6 events per cycle (increase it when goign to high jet multiplicities)
Ask at least O(5E6) events
Consider that you need more events when you go on the tails of the processes
(e.g. CSA07 high pT bnd for 5 partons we asked 8E8 events)

To run: ./zjetsgen < your_input_file

Step 3: the unweighted events

To unweight the events you need to

- start from the same input file used for weighted events
- run from the same directory (the output files from the weighted-events generation are needed)
- set imode to 2
- specify the final state (through izdecmod, iwdecmod, itdecmod, etc etc)

To run: ./zjetsgen < your_input_file

Additional remarks

- ALPGEN is faster than other generators, but in any case the generation might take time
- The time for generating weighted events can go from few minutes for 1 parton to several hours for 5 partons (up to O(day) for the tails of the distributions, as pTZ for CSA07)
- Unweighting is much faster that weighting (10K events are done in few seconds)

Step 4: the parton shower

The cfg file is given as a default in the AlpgenInterface

What you need to do:

1) set the minimum jet pT for matching (usually = pT threshold used for partons)
2) set minimum distance between partons and jets (usually = dR used for partons)
3) specify the matching mode: exclusive vs inclusive

Exclusive mode:

- The event is rejected if after matching all the partons to a jet additional jets (harder than those matched to the partons) are found as leftover
- To be use for ALL the parton multiplicities different than the largest

Inclusive mode:

- The event is kept even if additional jets (harder than those matched to the partons) are present after parton-jet matching
- To be used only for the largest jet multiplicity

IMPORTANT

- Include the filter in your path (see the interface) to remove empty events produced by the interface
- Add your sequence (simulation+reco) to get the output in the format you want

Using the official ALPGEN code

```
>mkdir ALPGENv213
>cd ALPGENv213
>wget http://mlm.home.cern.ch/mlm/alpgen/V2.1/v213.tgz
>tar -xf v213.tar
>cd wjetwork/
>make gen
>wjetgen < input
```

Or any other process you are interested to

You can customize the generation adding coding specific cuts in the function usrcut in the *usr.f file (specific of each process)

EXAMPLE: the pT binning for Z+jets in CSA07

```
c  USR will add possible extra cuts at this point.
c      if(cut-not-passed) goto 10
       ptw=sqrt(pw(1)**2+pw(2)**2)
       if(ptw.le.100) goto 10
       if(ptw.gt.300) goto 10

       return
 10       lnot= 1
       end
```

FAQ

How many events to ask?

If you want a corresponding luminosity L, you need to run a test (which also gives you a grid, such that you save time in generating the full sample)

You need:

- the cross section after unweighting σ_{unw}
- the matching efficiency ε_{match}
- the number of events asked N_{hasked} in the test run
- the number of events N_{match} after matching

You need then to ask $\varepsilon_{match} m * \sigma_{unw} * L * N_{asked} / N_{match}$

I need a sample of N jets. Is it enough to generate N partons?

In general, you need to generate all the multiplicities up to N

Sometimes (hard jets for SUSY/Exotica) this might not be the case

Do I need different generations for the same process in different final states (e.g. Z(ll)+jets vs Z(nn)+jets)?

You can use the same weighted events for different final states (except for specific cases, such as ZZ)

Can I run ALPGEN+HERWIG?

Yes in general. Not yet in CMSSW

To know more about ALPGEN

The ALPGEN web site (code, manual, and references)
http://mlm.home.cern.ch/mlm/alpgen/

A reference for MLM matching
http://arXiv.org/abs/hep-ph/0611129

THE CMS ALPGEN twiki
https://twiki.cern.ch/twiki/bin/view/Main/CmsAlpgen

A simple ALPGEN tutorial in CMS
https://twiki.cern.ch/twiki/bin/view/Main/AlpgenTutorial

The ALPGEN CSA07 notes (with an explanation on how to
set an ALPGEN production from the CMS user point of view)
http://cms.cern.ch/iCMS/jsp/openfile.jsp?type=IN&year=2007&files=IN2007_031.pdf
http://cms.cern.ch/iCMS/jsp/openfile.jsp?type=IN&year=2007&files=IN2007_038.pdf

IMPORTANT:

before doing anything

1) read (at least) the part of the manual related to the process you are interested to
2) get from ALPGEN the par.list file (running in imode 4) and go through ALL the parameters and ALL the default values
3) keep track of generation efficiency and cross sections through the three generation steps