

TruthDAOD

Responsible:


Not yet
Certified as
ATLAS
Documentation

Main.JamesCatmore

- ↓ [Introduction](#)
- ↓ [Available formats and how to make them](#)
- ↓ [TRUTH0](#)
- ↓ [TRUTH1](#)
- ↓ [TRUTH2](#)
- ↓ [TRUTH3](#)
 - ↓ [MetaData](#)
 - ↓ [Collections for specific states](#)
 - ↓ [Isolation and Dressing details](#)
 - ↓ [Jet/b-tagging contents](#)
 - ↓ [Truth MET](#)
 - ↓ [Mini-collections with vertices and decay products](#)
- ↓ [Additional Truth Containers and helpers for Derivations](#)
- ↓ [Collections in DAOD_PHYS and DAOD_PHYSLITE](#)
- ↓ [Testing and examples](#)
- ↓ [Print truth in ASCII form](#)
 - ↓ [Set-up](#)
 - ↓ [Dumping HepMC to ASCII](#)
 - ↓ [Dumping xAOD to ASCII](#)
- ↓ [Further description of different truth jets](#)

Introduction

Event generation jobs in ATLAS produce a format called EVNT, which contains the truth record in HepMC format, wrapped in a form readable by Athena. These formats cannot easily be read by stand-alone ROOT or [AnalysisBase](#) releases. Furthermore HepMC objects, having been developed externally, are not compatible with the ATLAS analysis EDM (xAOD). And, of course, there are many analysis objects (jets, dressed leptons) that are constructed from particles in the truth record, but are not themselves immediately available from the HepMC record. For these reasons, software has been prepared to convert EVNT into a variety of xAOD formats which can be read with ROOT, and which use the ATLAS analysis EDM.

These pages give details on the range of formats available and their contents. They do not give documentation on the xAOD truth itself - information on that can be found in the most recent [tutorial](#) .

Each group is responsible for defining their own truth derivation tags (this is not done centrally), so we can't provide a list of p-tags and functionality here for you. Instead, at the bottom of the page there are some release notes that might prove useful.

The truth derivation formats evolved during 2019, but should have stabilized. If you wish to see a previous version of the page for historical purposes, please use the history link at the bottom of this twiki. This page will document the current status, and will not attempt any lengthy review for brevity.

Available formats and how to make them

There are several formats defined. In release 21.2, these formats are:

- TRUTH0: faithful copy of the truth record, with nothing added or removed
- TRUTH1: TRUTH3 with the particles from TRUTH0 and truth charged particle jets
- TRUTH2: TRUTH3 with a special setup which thins the hard process
- TRUTH3: the main format for truth analysis. If you aren't sure what you need, this is the one for you. Its full contents are below.

To make these formats locally from an EVNT file you should first set up a working release, e.g.: `asetup 21.2.86.0,AthDerivation` and then run as

```
Reco_tf.py --inputEVNTFile evnt.pool.root --outputDAODFile test.pool.root --reductionConf TRUTH3
```

substituting the string at the end for the desired format(s). This will spin over the input file and produce the xAOD file, which in this case will be called `DAOD_TRUTH3.test.pool.root`. This can now be opened and inspected with the ROOT TBrowser, and analysed with Athena, [AnalysisBase](#), [AthAnalysis](#), etc in the usual way.

Note that you can also include the special TRUTH1 and TRUTH3 containers in your reconstructed-level derivations. To do this you need

```
from DerivationFrameworkMCTruth.MCTruthCommon import addStandardTruthContents
addStandardTruthContents()
```

and then make sure you add them to the slimming helper, for example:

```
from DerivationFrameworkMCTruth.MCTruthCommon import addTruth3ContentToSlimmerTool
addTruth3ContentToSlimmerTool(TRUTH3SlimmingHelper)
TRUTH3SlimmingHelper.AppendContentToStream(TRUTH3Stream)
```

Note that we have helpers for adding the standard truth contents to your output, but you are free to select a subset or to add additional collections as necessary.

These formats all contain several decorations on the `xAOD::EventInfo`: - The MET and HT that are used for the standard ttbar sliced samples, `GenFiltHT` and `GenFiltMET`; - HFOR information for Alpgen samples; - Truth classifications used by the Higgs and SUSY groups for relevant samples

If there are other generally useful event-level filter variables, they can be added.

TRUTH0

[Definition](#)

This is an exact copy of the input EVNT, in xAOD format. In consequence it is very large: ~25-50KB/event.

TRUTH1

[Definition](#)

This includes the reduced containers in TRUTH3 (see below) and the particles in TRUTH0. TRUTH1 also includes links to truth jet constituents and truth charged particle jets.

TRUTH2

[Definition](#)

TRUTH2 includes the TRUTH3 collections and the "compact hard truth record" as defined [here](#). The idea of the

compact hard truth is to keep the part of the generator record that people care most about, maintaining as much of a full navigability as possible, while still dropping many of the soft particles that users do not normally need.

TRUTH3

[Definition](#)

[Old summary slides](#)

This is the main truth analysis format. In addition to the main truth record it contains a series of extra containers for specific states, as well as various Truth Jet and Truth MET collections. The main truth record is also aggressively thinned to reduce the size. The idea of TRUTH3 is that all truth information that you might need by parsing the truth record is saved for you. This reduces the size of the derivation and also helps to standardize some of the truth information.

MetaData

All truth derivations come with [xAOD::TruthMetaData](#) objects, which include the MC process ID, a list of the names of all the weights in the HepMC record, and various metadata to do with the generator configuration. Note that truth weights should *not* be accessed directly, but only via the [recommended PMG tool](#).

Collections for specific states

These collections contain only particles of a certain type: TruthElectrons, TruthMuons, TruthTaus, TruthNeutrinos, TruthPhotons (no photons from pi0 or light mesons unless pT>20 GeV), TruthBoson, TruthBottom, TruthTop, TruthBSM, and TruthForwardProtons all with their corresponding aux containers.

These light collections are decorated with navigation information to access their mothers and daughters. You can directly access children and parents via [the standard parent and child accessors](#).

Electrons, muons, taus, and photons are decorated with truth isolation information, and all but photons are decorated with dressed truth kinematics.

Each particle in these collections has been decorated with the classification from the [MCTruthClassifier](#). The decorations can be accessed in analysis code as follows:

```
unsigned int recoPhoton_type   = ( *truthPh_itr )->auxdata< unsigned int >( "classifierParticleType" );
unsigned int recoPhoton_origin = ( *truthPh_itr )->auxdata< unsigned int >( "classifierParticleOrigin" );
```

The numbers stored in the decorations correspond to the classification enums defined [here](#).

Note that leptonically decaying bosons are included in most Sherpa truth derivations even when they were not provided by the generator. These bosons are derived from the leptons in the truth record and come with a number of caveats, but they may be useful to some users.

A special collection of Born leptons is also included, as originally defined by the Higgs to ZZ group. The implementation of their code is [here](#).

Isolation and Dressing details

The default settings for lepton isolation and dressing (adding soft FSR photons to a bare lepton's 4-momentum)

used in the TRUTH3 derivations are found [here](#). Isolation variables are constructed to be as close to the offline definitions as possible. For electrons and muons, the default dressing uses $R=0.1$, and for taus the default dressing uses $R=0.2$.

Options are available for using a jet algorithm for dressing.

Jet/b-tagging contents

There are two jet collections stored in TRUTH3:

- Small R jets: anti- k_t radius 0.4 jets with $p_T > 20$ GeV (`AntiKt4TruthDressedWZJets`). Muons and electrons from W/Z/H/tau are not included, nor are photons from Higgs decays or that are used in dressing (see below; i.e. an improvement from the `WZTruth` jets - see note below). For reference, `AntiKt4TruthJets` exclude all neutrinos and muons in the truth record, and include all other stable particles. If you use small-R jets in your analysis, these are good truth jets to use. Aside from the jet 4-vector, some information for classifying the type of the jet is stored:
 - `vector<int> AntiKt4TruthDressedWZJetsAuxDyn_HadronConeExclTruthLabelID`; ([code](#), [doc](#)): The decoration [recommended](#) by the HF tagging group for identifying a jet as b- or c-tagged at particle level.
 - `vector<int> AntiKt4TruthDressedWZJetsAuxDyn_PartonTruthLabelID`; ([code](#), [doc](#)): The decoration [recommended](#) by the jet group for identifying a jet as quark- or gluon-initiated and for [related uncertainties](#).
 - `vector<int> AntiKt4TruthDressedWZJetsAuxDyn_TrueFlavor`; A merging of the above two tags. In the case that `HadronConeExclTruthLabelID` indicates that the jet is not b- or c-tagged, but the `PartonTruthLabelID` suggests that it should be, the label is multiplied by 100 (e.g. 105 indicates `HadronConeExclTruthLabelID` of 0 and `PartonTruthLabelID` of 5).
- Large R jets: anti- k_t radius 1.0 jets with $p_T > 100$ GeV after trimming with $f_{\text{cut}}=0.05$. The standard $R=0.2$ subjets with k_t subjets are used. Aside from the 4-vector, we also keep the first three n-subjettiness values (with the winner-takes-all axis) and truth D2.

Truth MET

Truth missing transverse momentum is documented [here](#). Note that while some analyses use `NonInt` (the vector sum of non-interacting particles), in many cases it may be more accurate to sum `NonInt` and `IntOut`, and muons outside of acceptance may need special treatment.

Mini-collections with vertices and decay products

Special collections are available for certain particles that include the particles their decay products, and any intermediate vertices. By default these are:

- BSM and decay particles. This saves the BSM particles, their decay products, and the vertices between them so that the record is connected and navigable and the positions of the decays are available. Note that in the case of strong decays or prompt decays this collection can become quite large; it works best for things like displaced leptonic decays, where the tree is minimal.
- Boson and decay particles. This saves the W, Z, and Higgs bosons and their immediate daughters. The

intent is to allow re-weighting for boson polarization, for example.

- A [hard scatter collection](#). This comes in two flavors. The first is simply grabbing the vertex identified as the "hard scatter" in the truth record and keeping all its incoming and outgoing particles, as well as the next generation of outgoing particles (to handle intermediate resonances). Because of the way Pythia8 works this doesn't always get everything, some additional incoming particles are added based on their status codes. There is also a special setup for a pure-Pythia8 matrix element that largely reconstructs what that hard scatter ought to be based on particle status codes.

Additional Truth Containers and helpers for Derivations

A number of other tools and options are available for those looking for additional information in their derivations, including:

- [addTausAndDownstreamParticles](#), for a small collection of taus and their decay products
- [addEgammaAndDownstreamParticles](#), for a small collection of e/gamma particles and their decay products
- [addHFAndDownstreamParticles](#), for a small collection of heavy-flavor hadrons and their decay products
- [addPVCollection](#), to add a primary vertex collection to your derivation (one truth vertex per event).

For many of the above options, it is possible to set the number of generations after the particle that you would like to save, if you don't want to keep everything downstream. Additional helper functions can be designed upon request, preferably via [this JIRA ticket](#).

One important helper is `addMiniTruthCollectionLinks()`, which will reset all truth particle links for reconstructed electrons, muons, and photons to point to the mini truth containers. Using that function will ensure that you have working truth particle links on your reconstructed objects.

Collections in DAOD_PHYS and DAOD_PHYSLITE

The new derivations DAOD_PHYS and DAOD_PHYSLITE contain the standard TRUTH3 contents. They also include the truth primary vertex collection (see above). DAOD_PHYS includes the truth HF hadron and decay collection.

Testing and examples

You can find an example of code running over a truth derivation and producing histograms [here](#). [SimpleAnalysis](#) is also an analysis package designed to run over TRUTH3 formatted derivations.

There is an outdated but dedicated tutorial on truth DxAODs [here](#).

Print truth in ASCII form

Sometimes, in order to understand some feature coming out of a generator, it is useful to print a few events directly to text. As it happens, the HepMC definition originally came from an ASCII structure, so there is a very well defined form for printed truth. This is used in the applications below.

Set-up

The code for printing both HepMC and xAOD are implemented as simple Athena algorithms. This is a necessity for HepMC, and whilst it would be possible to use ROOT for the xAOD printing, as a matter of convenience we use the same package for both. The package is the same one that contains the main HepMC → xAOD converter,

which is run in all xAOD-building jobs. To run the text printing applications you need to set up any xAOD-compatible ATLAS software release, e.g.

```
setupATLAS
asetup 20.1.8,gcc48,here
```

Dumping HepMC to ASCII

First you need to acquire the relevant job options:

```
get_files HepMCTruthReader_jobOptions.py
```

Open the file and modify the input EVNT file name to point at the one you are using in the exercises. Also modify the number of events from -1, which means "process everything", to 5 events. This will avoid you getting a monster text file.

To run the printing you just do:

```
athena HepMCTruthReader_jobOptions.py > HepMCTruth.txt
```

Open the log file, and, using the information on the slides which preceded this tutorial, see if you can make sense of the output. Ask for help if you can't!

Dumping xAOD to ASCII

The relevant job options are:

```
get_files xAODTruthReader_jobOptions.py
```


Open the job options and modify the name of the input file to match the input xAOD file, and also reduce the number of events to 5 or so to avoid making a massive log file. Now, run the job:

```
athena xAODTruthReader_jobOptions.py > xAODTruth.txt
```

The format of the ASCII dump is the same as for HepMC. Note, however, that within a given decay the particles may emerge in a different order so a direct diff isn't usually possible. **Also, note that this algorithm requires graph-complete xAOD truth, so if thinning has been performed and decay/production vertices have been removed, the algorithm will crash.** In practical terms this means you can read TRUTH0 and full xAOD, but the other TRUTH derivations and the truth in the main physics derivations will not be readable by this algorithm, since they do not usually preserve full graph connectivity.

Further description of different truth jets

There are several different types of truth jets in use in ATLAS, and what exactly goes into each can be rather confusing. Here a brief but precise description is given, with links to code.

- **TruthJets**: These are the 'traditional' truth jets. They exclude muons and weakly interacting stable particles (neutrinos, neutralinos), and include all other particles. These were built with the calorimeter response in mind: the calorimeter does not gather much energy from muons, and so these should represent the true reference for calorimeter jet calibration. They remain the true reference used in the jet calibration group.
- **TruthWZJets**: These exclude prompt leptons (from W, Z, Higgs, and taus) and prompt photons from Higgs decays. They also exclude FSR photons in a cone of 0.1 around the prompt leptons. Their [getter is here](#), .

which uses [this truth collection](#). That truth collection is made with [this tool](#), which has a default radius of 0.1 for photon dressing and uses its own [definition of prompt](#). It looks for MCTruthClassifier output indicating that the photon is an FSRPhoton.

- **TruthDressedWZJets**: These are conceptually identical to the above, with a slightly different dressing configuration. They are [defined here](#) using the "Decoration Name" option instead of the "!FSR Cone" option of the `CopyTruthJetParticles` tool. The dressing decoration is set [here](#) and is applied in the [dressing tool](#). The default dressing tool configuration [is here](#). Note that in this case, the dressing tool uses a "not from hadrons" definition that is defined [here](#). This somewhat more complicated definition of a dressing photon came from the PMG group and is the current default. The more complex configuration allows greater flexibility: if we change at some point to a jet algorithm for dressing, the tools can adapt relatively quickly.

Major updates:

-- [JamesCatmore](#) - 2015-04-02 -- [BenNachman](#) - 2015-09-22 -- [ZacharyMarshall](#) - 2019-10-31

Responsible: [JamesCatmore](#)

Last reviewed by: **Never reviewed**

Topic revision: r45 - 2020-01-28 - [ChrisGutschow](#)

Copyright &© 2008-2020 by the contributing authors. All material on this collaboration platform is the property of the contributing authors.
Ideas, requests, problems regarding TWiki? [Send feedback](#)

