ACTS Track fitting/finding Tutorial

This tutorial sets up the truth fitting/finding with KF/CKF and looks at the performance plots.
Note: This has to be done based on truth hits and particles produced by fatras simulation.

Setting up ACTS

Please follow the guide here (https://indico.cern.ch/event/917970/contributions/3861993/attachments/2044486/3424773/go) for the installation.
When compiling, the option for building Fatras, Examples and Pythia8 should be turned on:

```
cmake -DACTS_BUILD_FATRAS=ON -DACTS_BUILD_EXAMPLES=ON -DACTS_BUILD_EXAMPLES_PYTHIA8=O
```

Setting up Detector and Magnetic field

This should be the same for both simulation and reconstruction

Detector

Currently, reconstruction (including truth fitting/finding) only works for `GenericDetector`.

Magnetic field

- **Constant B field**: set up via `--bf-values`, e.g. `--bf-values 0 0 2`
- **Using B field map**: set up via `--bf-map`
  For example, to use ATLAS B field map, we need to git clone the `acts-data` repository first:

```
git clone https://gitlab.cern.ch/acts/acts-data.git
export actsdata=$PWD
```

Then we could set up the B field with

```
--bf-map ${actsdata}/MagneticField/ATLAS/ATLASBField_xyz.txt
```

NB: The following job running will be done in the installation directory `build/bin` where all the executables are located. But you can run those commands below anywhere by using the full path of the executables.

Running Fatras simulation

This will generate truth particles and truth hits. Examples to generate samples to be used for reconstruction later:

- **Single muon** generated with Particle Gun:

```
./ActsSimFatrasGeneric --bf-map ${actsdata}/MagneticField/ATLAS/ATLASBField_xyz.txt --evg-input-type gun --p
```

- **ttbar at mu = 200** generated with Pythia8:

```
./ActsSimFatrasGeneric --bf-map ${actsdata}/MagneticField/ATLAS/ATLASBField_xyz.txt --evg-hard-process Top
```

Running Truth fitting with KalmanFitter

- Run truth fitting with **single muon** sample generated above:

```
./ActsRecTruthTracks --bf-map ${actsdata}/MagneticField/ATLAS/ATLASBField_xyz.txt --input-dir=sim_generic_e
```

- Run truth fitting with **ttbar** sample generated above:

```
./ActsRecTruthTracks --bf-map ${actsdata}/MagneticField/ATLAS/ATLASBField_xyz.txt --input-dir=sim_generic_A
```

Running Track finding with CKF

Note that we might want to change the source link selection criteria for the CKF. Currenlty, the CKF example has the interface to tune two criteria:

- **Global maximum chi2 of Kalman filtering**. This could be set up via `--ckf-slselection-chi2max`
- **Global maximum number of source links on a surface**. This could be set up via `--ckf-slselection-nmax`

The default selection criteria is `--ckf-slselection-chi2max 15 --ckf-slselection-nmax 10`

- Run track finding with **ttbar** sample generated above using default CKF criteria:

```
./ActsRecCKFTracks --bf-map ${actsdata}/MagneticField/ATLAS/ATLASBField_xyz.txt --input-dir=sim_generic_AT
```

- Run track finding with **ttbar** sample generated above using `--ckf-slselection-chi2max 100 --ckf-slselection-nmax 1` (i.e. only select the best source link with a very loose chi2 constraint):

```
./ActsRecCKFTracks --bf-map ${actsdata}/MagneticField/ATLAS/ATLASBField_xyz.txt --ckf-slselection-chi2max 1(
```

Looking at performance plots

- KF performance (resolution, efficiency)

```
root -l rec_kf_generic_ATLASB_ttbar_e1_pu200_eta2.5/performance_track_fitter.root
```

To look at the perigee parameters resolution:

```
root -l '../../Examples/Scripts/perigeeParamResolution.C("rec_kf_generic_ATLASB_ttbar_e1_pu200_eta2.5/perfor
```

- CKF performance (efficiency, fake rate, duplication rate)

```
root -l rec_ckf_generic_ATLASB_ttbar_e1_pu200_eta2.5/performance_ckf.root
```