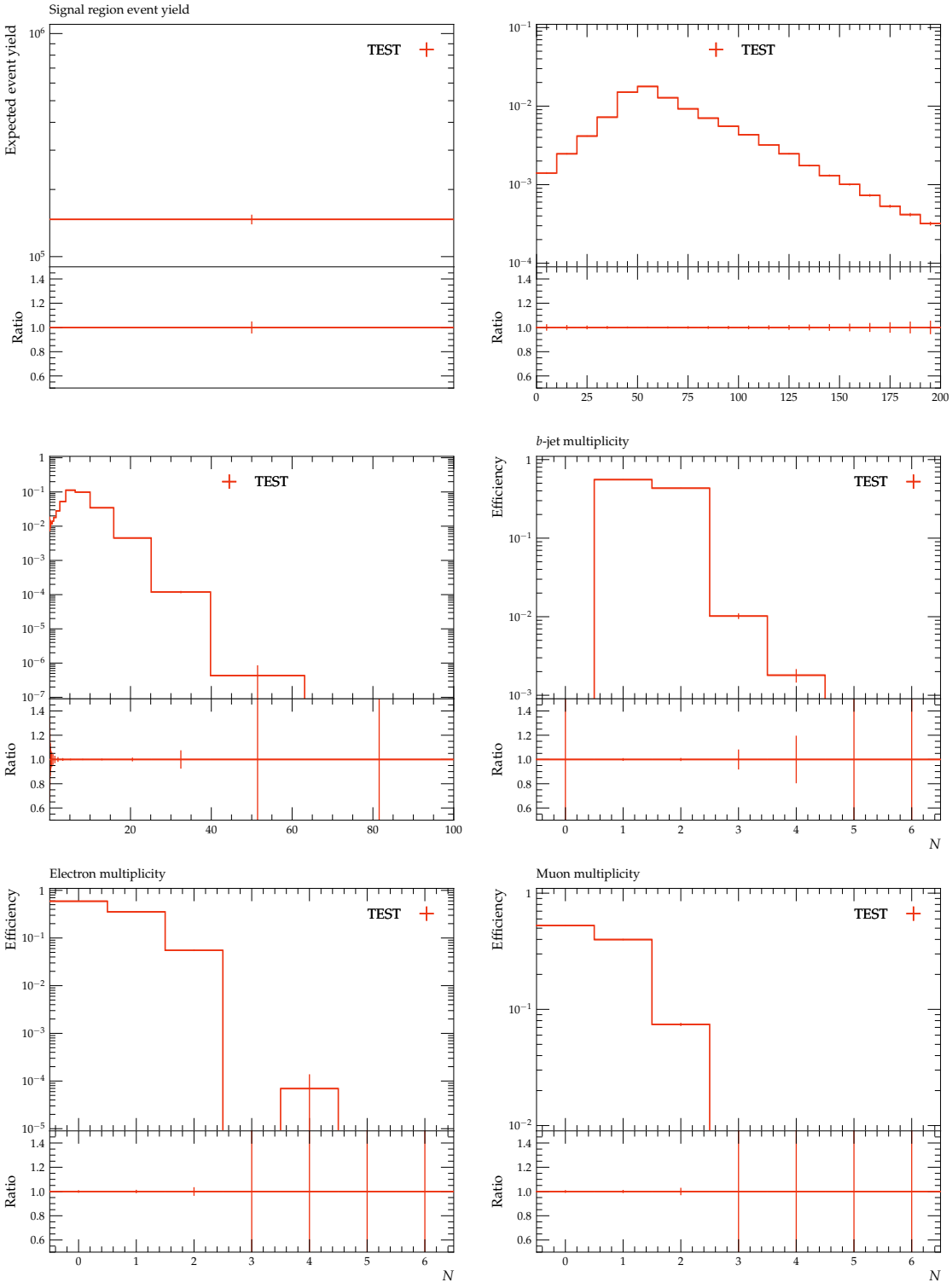
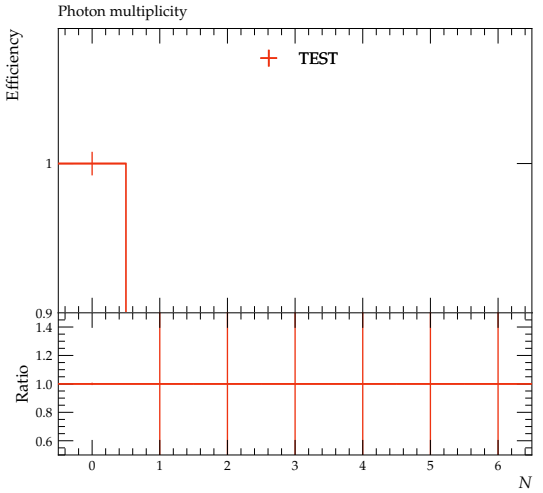
```
// -*- C++ -*-
#include "Rivet/SimpleAnalysis.hh"

namespace Rivet {


  /// @brief Demonstrate the SimpleAnalysis interface
  class EXAMPLE_SIMPLEANALYSIS : public SimpleAnalysis {
  public:

    /// Constructor
    RIVET_DEFAULT_SIMPLEANALYSIS_CTOR(EXAMPLE_SIMPLEANALYSIS);


    /// @name Analysis methods
    /// @{

    /// Book histograms
    void init() {

      // Set the detector-smearing family
      setDetSmearing(ATLAS_RUN2, MV2C20);

      // Book cutflow
      book(_cutflow, "cutflow", {"MET", "BJets", "1Lep+2BJet100"});

      // Book histograms
      book(_h["ne"], "num_elec", 7, -0.5, 6.5);
      book(_h["nm"], "num_muon", 7, -0.5, 6.5);
      book(_h["nt"], "num_tau", 7, -0.5, 6.5);
      book(_h["ny"], "num_photon", 7, -0.5, 6.5);
      book(_h["njb"], "num_bjet", 7, -0.5, 6.5);
      book(_h["ptjb1"], "pt_bjet1", logspace(20, 30, 1000));
      book(_h["ptl1"], "pt_lep1", logspace(20, 20, 1000));
      book(_h["met"], "met", logspace(20, 1, 1000));
      book(_h["met_signf"], "met_signf", logspace(20, 0.01, 100));
      // Book counter
      book(_c["npass"], "count_pass");
    }


    /// Perform the per-event analysis
    void analyze(const Event& event) {
      /// @todo Automate / build-in the cutflow
      _cutflow->fillinit();

      // Study MET
      _h["met"]->fill(met()/GeV);
```

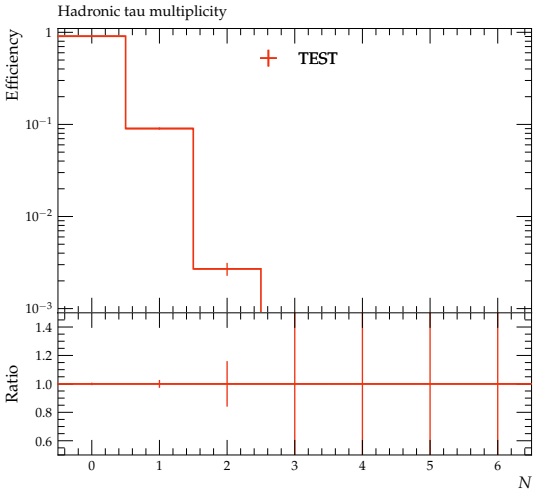
```
  _h["met_signf"]->fill(metSignf());

  // Apply a missing-momentum cut
  if (met() < 100*GeV) vetoEvent;
  _cutflow->fillnext();

  // Run the overlap removal (see the SimpleAnalysis base class for
     some info)
  doSimpleOverlapRemoval();

  // Debug writing
  MSG_DEBUG("Numbers␣of␣e,␣mu,␣tau,␣photon,and␣b-jet:␣"
            << electrons().size() << ",␣"
            << muons().size() << ",␣"
            << taus().size() << ",␣"
            << photons().size() << ",␣"
            << bjets().size());

  // Veto event if there are no b-jets (example eta cut to fully
     contain a 0.4 jet in the tracker)
  const Jets jbs = bjets(Cuts::pT > 30*GeV && Cuts::abseta < 2.1);
  if (jbs.empty()) vetoEvent;
  _cutflow->fillnext();

  // Get electrons or muons and require one to three of them
  const Particles leps =
    sortByPt(electrons(Cuts::pT > 20*GeV && Cuts::abseta < 2.5) +
             muons(Cuts::pT > 20*GeV && Cuts::abseta < 2.5));

  // Fill histograms with object counts, and leading b-jet and lepton
     pTs
  _h["ne"]->fill(electrons().size());
  _h["nm"]->fill(muons().size());
  _h["nt"]->fill(taus().size());
  _h["ny"]->fill(photons().size());
  _h["njb"]->fill(jbs.size());
  if (!jbs.empty()) _h["ptjb1"]->fill(jbs[0].pT()/GeV);
  if (!leps.empty()) _h["ptl1"]->fill(leps[0].pT()/GeV);

  // Fill counter for (made-up) signal-region passes
  if (leps.size() != 1 && bjets(Cuts::pT > 100*GeV).size() == 2) {
    _cutflow->fillnext();
    _c["npass"]->fill();
  }
}


/// Normalise histograms etc., after the run
```

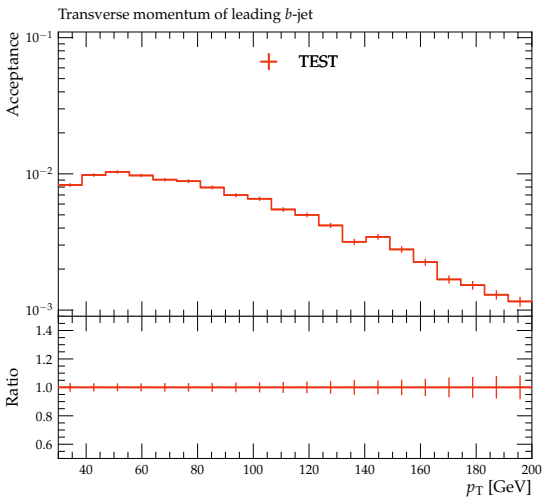
```
    void finalize() {
      normalize(_h); // normalize to unity
      scaleToIntLumi(_c, 300/femtobarn); // normalize to 300/fb
      MSG_INFO(_cutflow);
    }

    /// @}


    /// @name Histograms and cutflow
    /// @{
    map<string, Histo1DPtr> _h;
    map<string, CounterPtr> _c;
    CutflowPtr _cutflow;
    /// @}

  };


  RIVET_DECLARE_PLUGIN(EXAMPLE_SIMPLEANALYSIS);

}
```

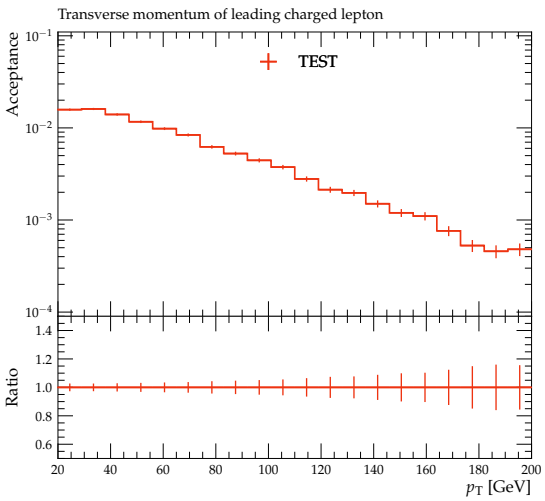
```
Rivet.Analysis.EXAMPLE_SIMPLEANALYSIS: INFO  /EXAMPLE_SIMPLEANALYSIS/cutflow cut-flow:
                     Weight            Count    A_cumu    A_incr
                    10000.0           10000.0      100%       -
Pass MET             1782.0            1782.0       18%       18%
Pass BJets           1488.0            1488.0       15%       84%
Pass 1Lep+2BJet100     48.0              48.0        0%        3%
```


Signal region event yield
Expected event yield
TEST
Ratio
TEST
Ratio
TEST
Ratio
b-jet multiplicity
Efficiency
TEST
Ratio
N
Electron multiplicity
Efficiency
TEST
Ratio
N
Muon multiplicity
Efficiency
TEST
Ratio
N


Photon multiplicity
Efficiency
TEST
Ratio
N


Hadronic tau multiplicity
Efficiency
TEST
Ratio
N


Transverse momentum of leading b-jet
Acceptance
TEST
Ratio
pT [GeV]


Transverse momentum of leading charged lepton
Acceptance
TEST
Ratio
pT [GeV]