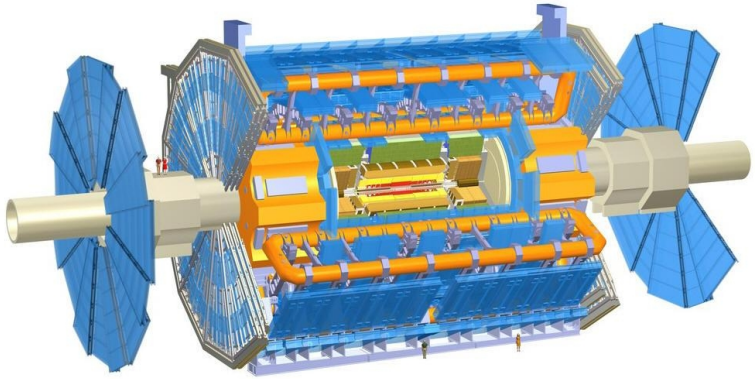
Einführung in Event-Displays für ATLAS Analysen

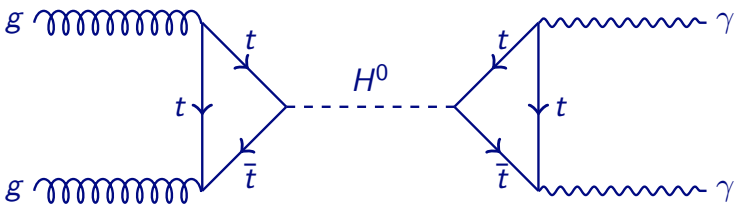
Jochen Jens Heinrich

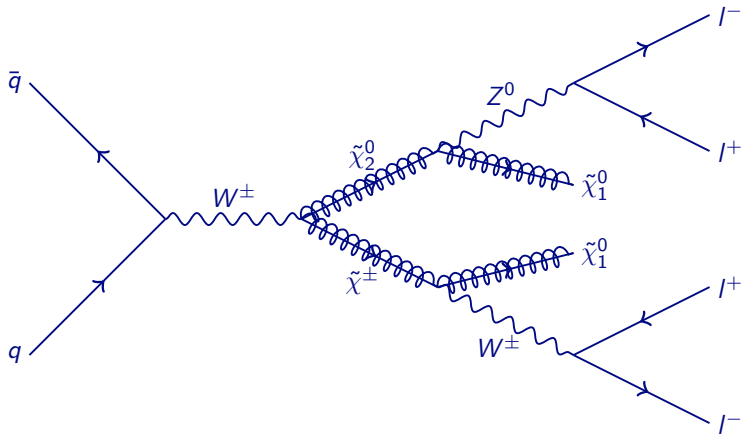
Datenauswertung in der Teilchenphysik

14. April 2015

Eine typische Analyse





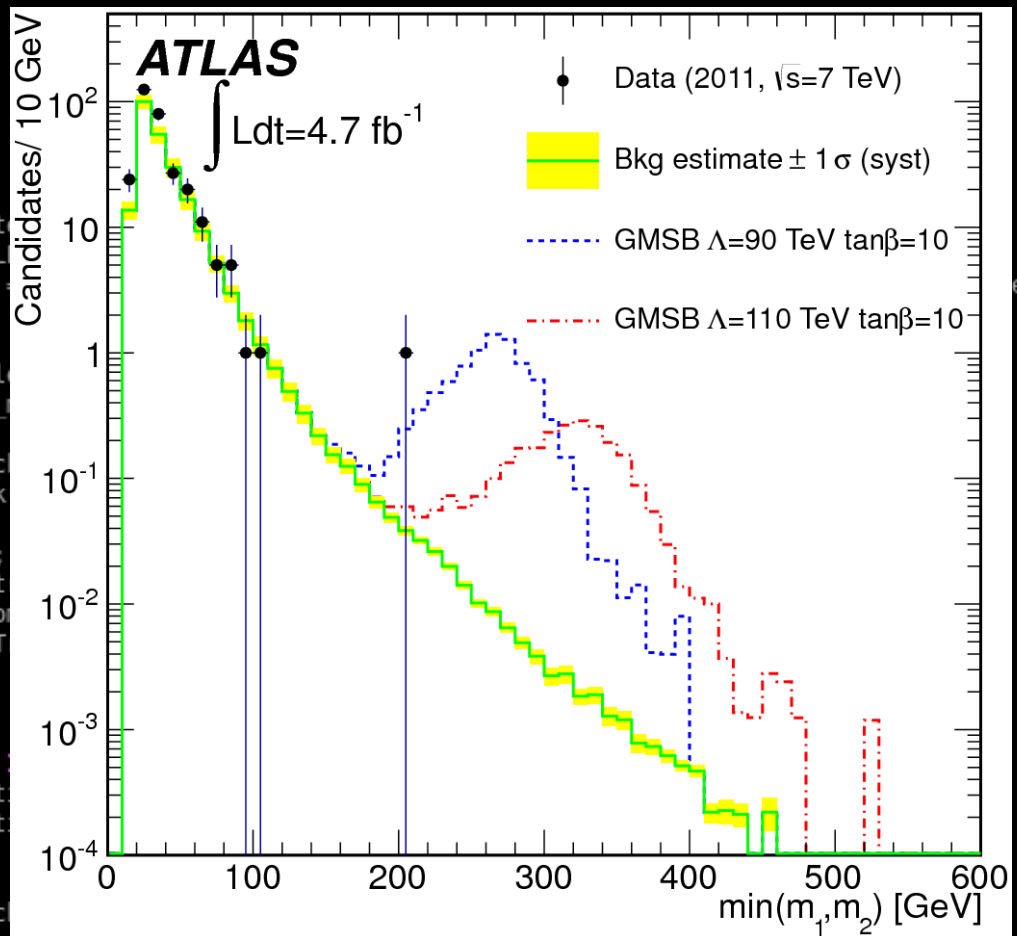


Eine typische Analyse

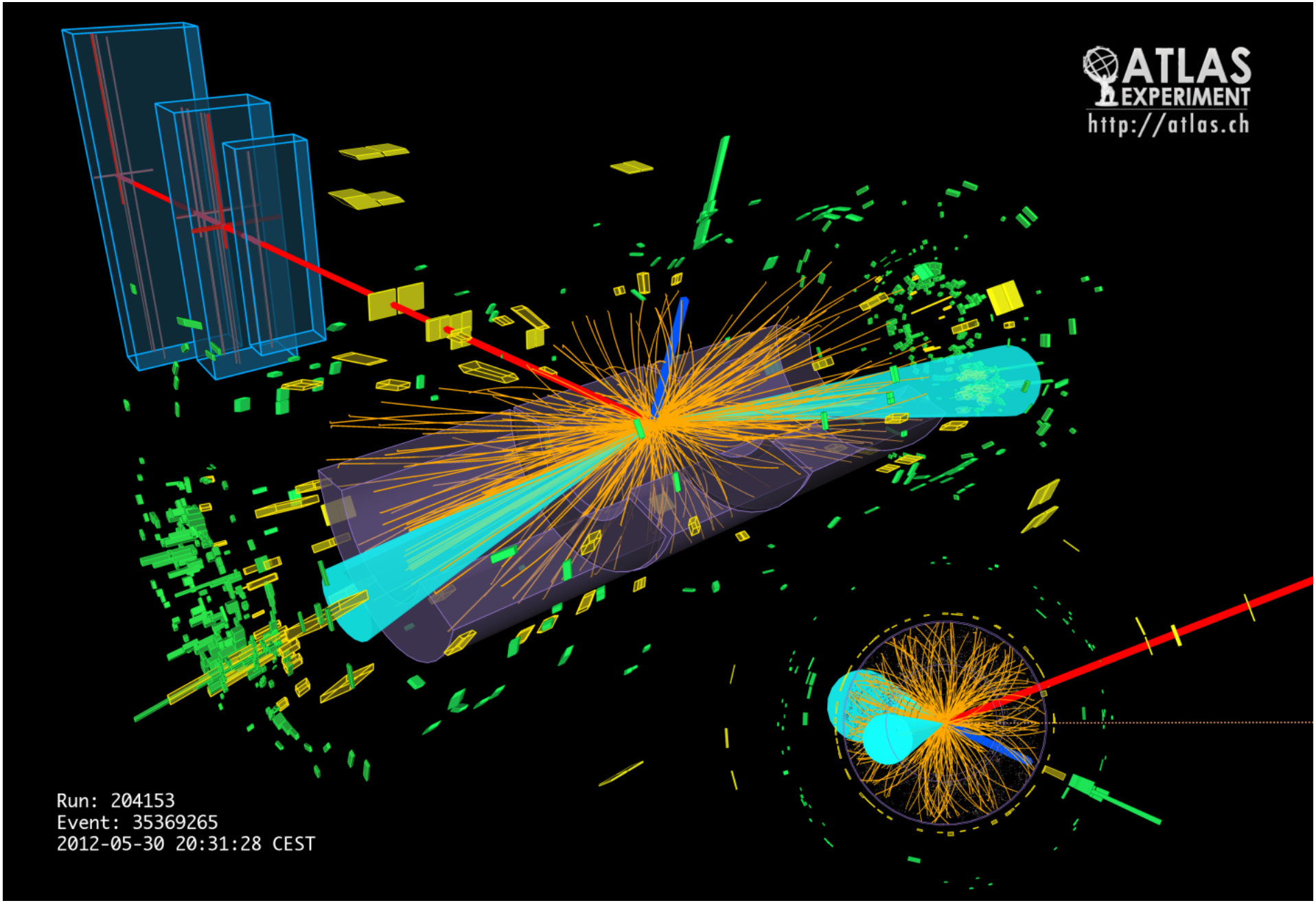
```
//              Creating and filling the LLPs container                 //
////////////////////////////////////////////////////////////////////////////

  if ( DEBUG ) Info("execute()","Creating and filling the LLPs container.");
  xAOD::LLPContainer* llps = new xAOD::LLPContainer();
  xAOD::LLPAuxContainer* llpsAux = new xAOD::LLPAuxContainer();
  llps->setStore( llpsAux );
  std::vector<int> selTrackIndex;
  // Looping over all muons and identifying LLP candidates
  if ( DEBUG ) Info("execute()","Searching for LLP candidates in the muon container.");
  xAOD::MuonContainer::const_iterator muon_itr = muons->begin();
  xAOD::MuonContainer::const_iterator muon_end = muons->end();
  for( int i_muon = 0; muon_itr != muon_end; ++muon_itr, ++i_muon ) {
        // Dummy selection: Only muons with |eta| < 1.5
        if ( fabs((*muon_itr)->eta()) > 1.5 ) continue;
        xAOD::LLP* llp = new xAOD::LLP();
        xAOD::LLP::Test myTest = xAOD::LLP::Test::unknown;
        llp->makePrivateStore( );
        int author = (*muon_itr)->author();
        llp->setTest( myTest );
        // set candidateType
        xAOD::LLP::CandidateType candidate_type = xAOD::LLP::CandidateType::undetermined;
        if ( (*muon_itr)->muonType() == 0 ) candidate_type = xAOD::LLP::CandidateType::CombinedMuonCandidate;
        if ( (*muon_itr)->muonType() == 1 || (*muon_itr)->muonType() == 3 ) candidate_type = xAOD::LLP::CandidateType::MuonStandAloneCandidate;
        llp->setCandidateType( candidate_type );
        // setting links
        static ElementLink< xAOD::MuonContainer > *link_muon = new ElementLink< xAOD::MuonContainer >;
        *link_muon = ElementLink<xAOD::MuonContainer>( *muonsSlim, i_muon );
        llp->setMuonLink( *link_muon );
        static ElementLink< xAOD::TrackParticleContainer > *inDetTrackParticleLink = new ElementLink< xAOD::TrackParticleContainer >;
        *inDetTrackParticleLink = (*muon_itr)->inDetTrackParticleLink();
        llp->setInDetTrackParticleLink( *inDetTrackParticleLink );
        selTrackIndex.push_back( (*inDetTrackParticleLink).index() );
        static ElementLink< xAOD::TrackParticleContainer > *muonSpectrometerTrackParticleLink = new ElementLink< xAOD::TrackParticleContainer >
        *muonSpectrometerTrackParticleLink = (*muon_itr)->muonSpectrometerTrackParticleLink();
        llp->setMuonSpectrometerTrackParticleLink( *muonSpectrometerTrackParticleLink );
        llps->push_back( llp );
  }
  // Now we loop over all ID tracks
  if ( DEBUG ) Info("execute()","Searching for LLP candidates in the InDetTrackParticles container.");
  xAOD::TrackParticleContainer::const_iterator track_itr = track_particles->begin();
  xAOD::TrackParticleContainer::const_iterator track_end = track_particles->end();
  double track_eta;
  bool previousCandidate;
  for(int i_track = 0 ; track_itr != track_end; ++track_itr, ++i_track ) {
        // Check if the considered track is already a candidate
```

Eine typische Analyse

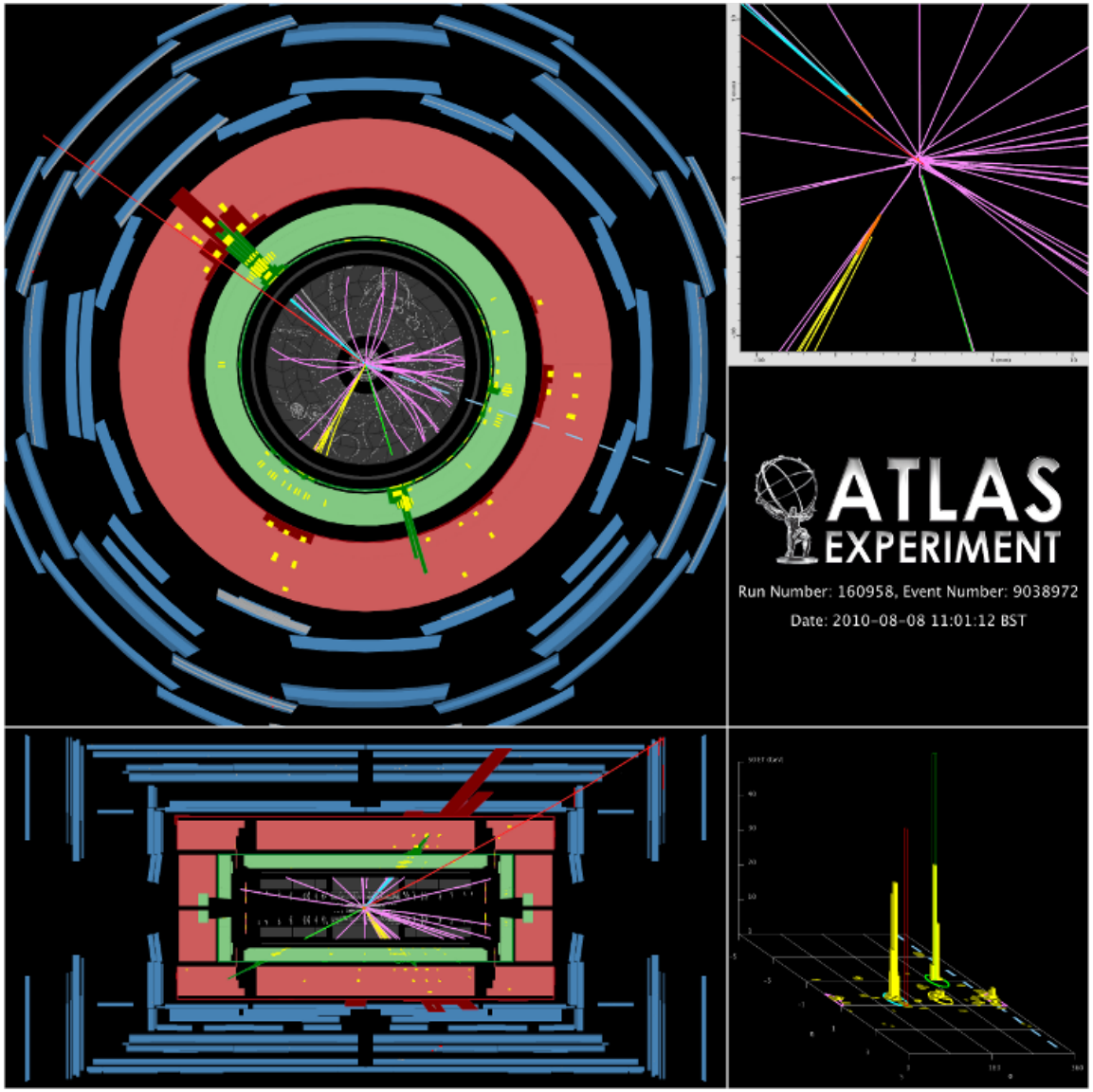
```
//              Creating and filling the LLPs container                 //
//////////////////////////////////////////////////////////////////////////

  if ( DEBUG ) Info("execute()","Creating and filling the LLPs container.");
  xAOD::LLPContainer* llps = new xAOD::LLPContainer();
  xAOD::LLPAuxContainer* llpsAux = new xAOD::LLPAuxContainer();
  llps->setStore( llpsAux );
  std::vector<int> selTrackIndex;
  // Looping over all muons and identifying LLP candidates
  if ( DEBUG ) Info("execute()","Searching for LLP candidates in the muon container.");
  xAOD::MuonContainer::const_iterator muon_itr = muons->begin();
  xAOD::MuonContainer::const_iterator muon_end = muons->end();
  for( int i_muon = 0; muon_itr != muon_end; ++muon_itr, ++i_muon ) {
        // Dummy selection: Only muons with |eta| < 1.5
        if ( fabs((*muon_itr)->eta()) > 1.5 ) continue;
        xAOD::LLP* llp = new xAOD::LLP();
        xAOD::LLP::Test myTest = xAOD::LLP::Test::unknown;
        llp->makePrivateStore( );
        int author = (*muon_itr)->author();
        llp->setTest( myTest );
        // set candidateType
        xAOD::LLP::CandidateType candidate_type = xAOD::LLP::Candidat
        if ( (*muon_itr)->muonType() == 0 ) candidate_type = xAOD::LL
        if ( (*muon_itr)->muonType() == 1 || (*muon_itr)->muonType()
        llp->setCandidateType( candidate_type );
        // setting links
        static ElementLink< xAOD::MuonContainer > *link_muon = new El
        *link_muon = ElementLink<xAOD::MuonContainer>( *muonsSlim, i_
        llp->setMuonLink( *link_muon );
        static ElementLink< xAOD::TrackParticleContainer > *inDetTrac
        *inDetTrackParticleLink = (*muon_itr)->inDetTrackParticleLink
        llp->setInDetTrackParticleLink( *inDetTrackParticleLink );
        selTrackIndex.push_back( (*inDetTrackParticleLink).index() );
        static ElementLink< xAOD::TrackParticleContainer > *muonSpect
        *muonSpectrometerTrackParticleLink = (*muon_itr)->muonSpectro
        llp->setMuonSpectrometerTrackParticleLink( *muonSpectrometerT
        llps->push_back( llp );
  }
  // Now we loop over all ID tracks
  if ( DEBUG ) Info("execute()","Searching for LLP candidates in the
  xAOD::TrackParticleContainer::const_iterator track_itr = track_part
  xAOD::TrackParticleContainer::const_iterator track_end = track_part
  double track_eta;
  bool previousCandidate;
  for(int i_track = 0 ; track_itr != track_end; ++track_itr, ++i_trac
        // Check if the considered track is already a candidate
```



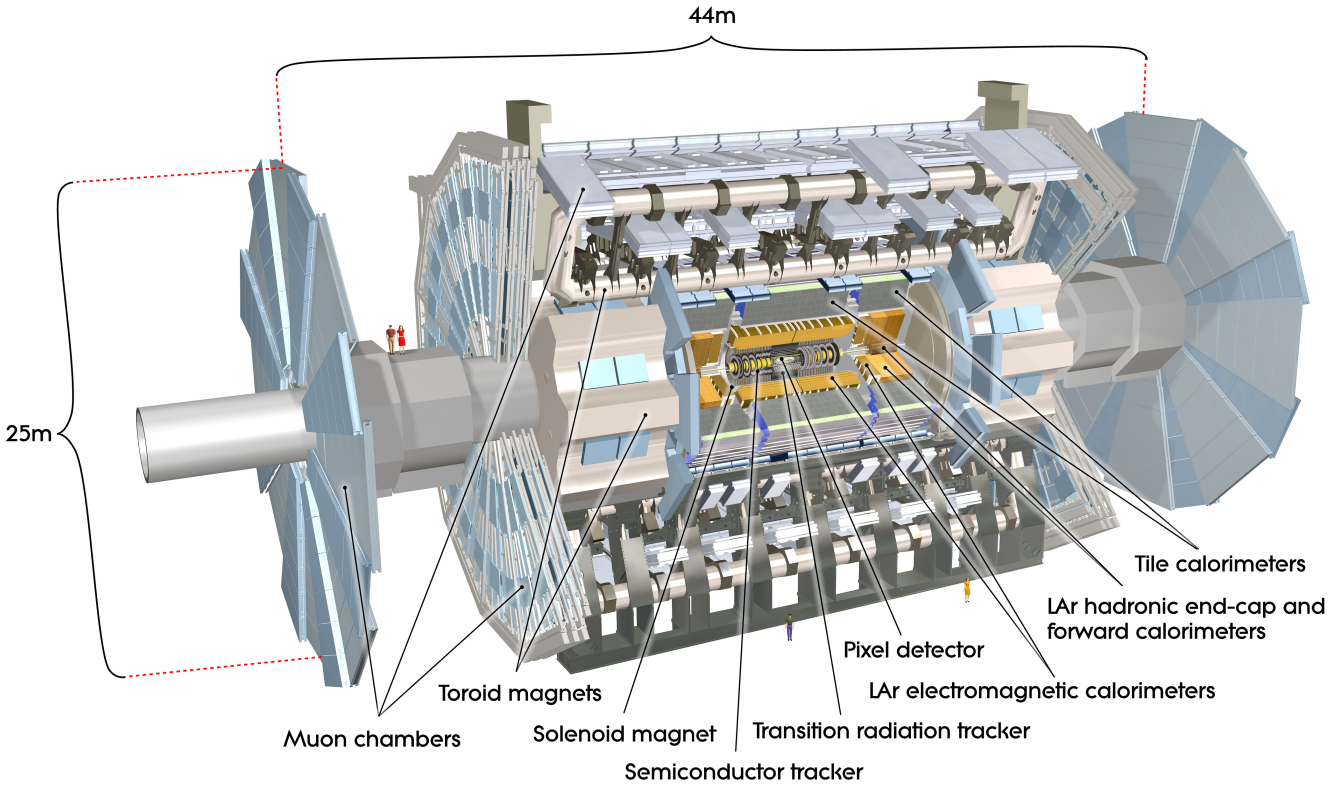
Event displays

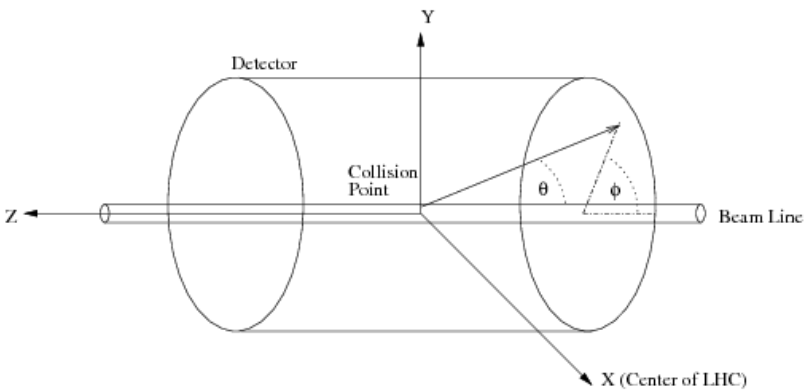


Event displays



A Toroidal LHC ApparatuS

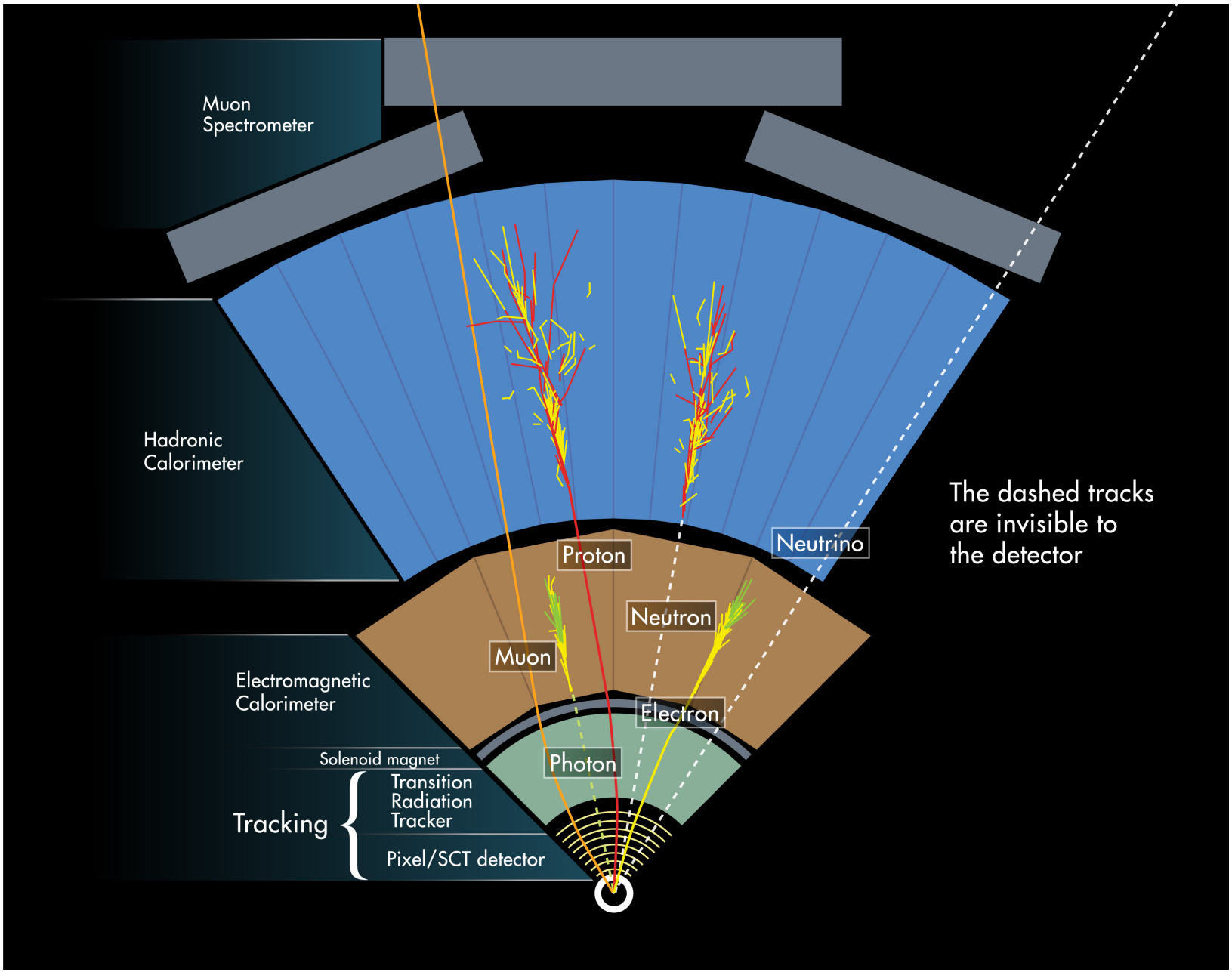




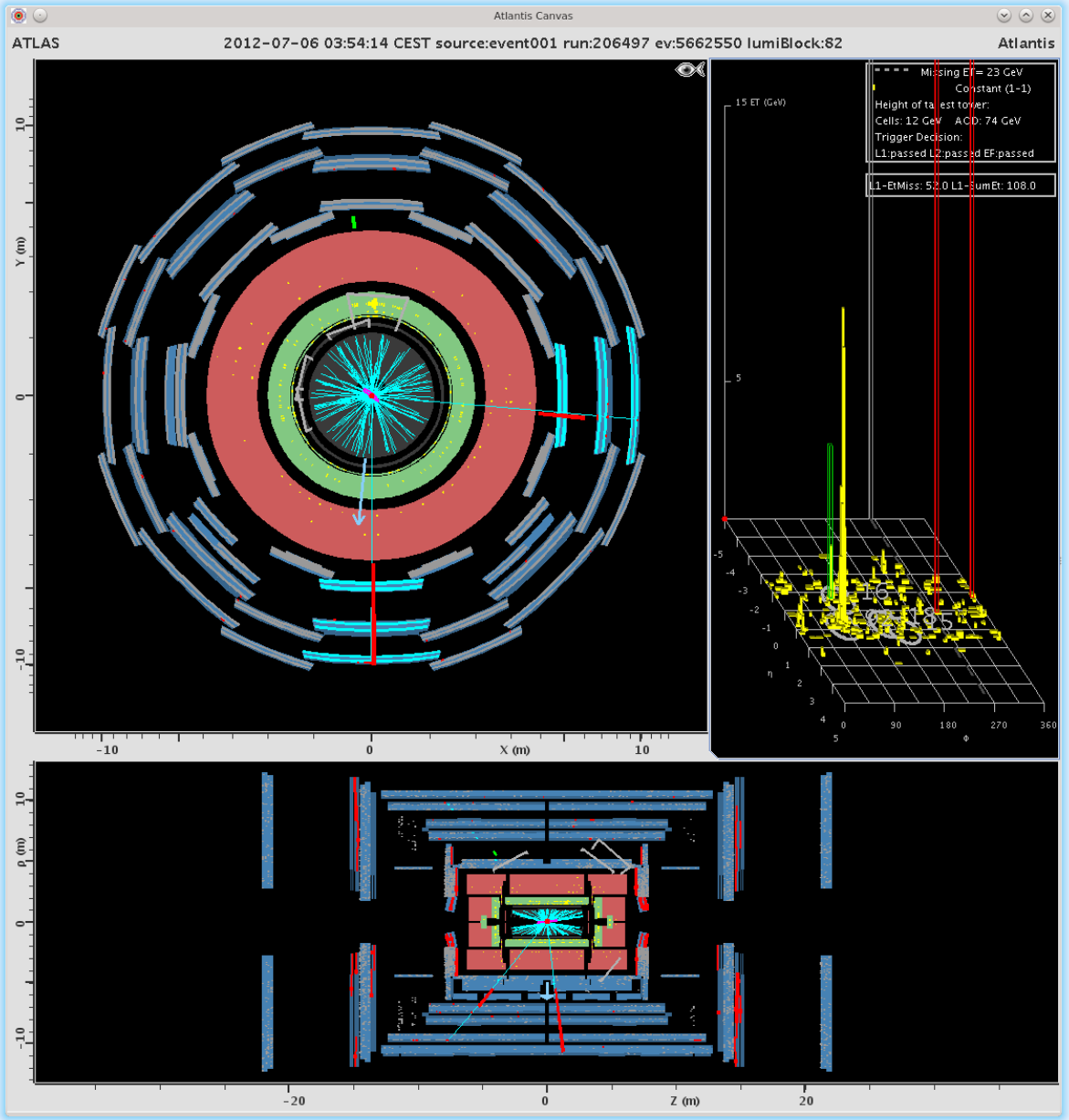
$$y = \frac{1}{2} \cdot \ln\left(\frac{E + p_L}{E - p_L}\right) \quad \text{Rapidität}$$

$$\eta = -\ln\left(\tan\left(\frac{\theta}{2}\right)\right) \quad \text{Pseudorapidität}$$

Teilchenidentifikation



HYPATIA - Event Displays für ATLAS



HYPATIA - Event Displays für ATLAS

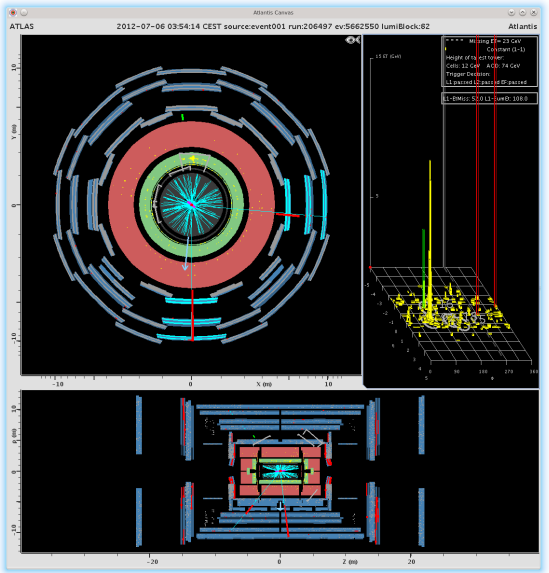
HYPATIA

Hypatia starten

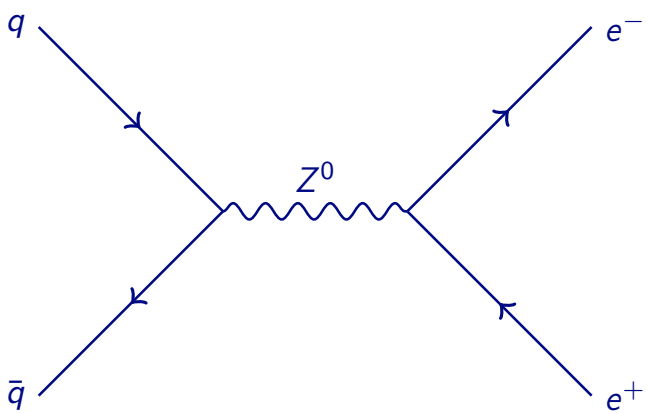
- `cp -r -v /project/etp6/heinrich/Hypatia/ .`
- `cd Hypatia`
- `./HYPATIA_for_Linux.sh`

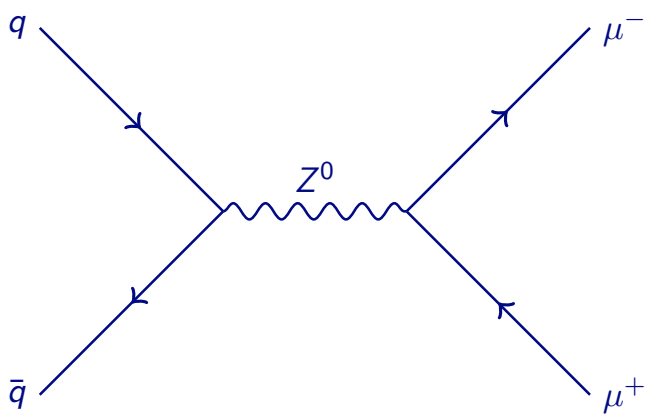
Einlesen der Ereignisse

- `File -> Read Event Locally`
- `datenpakete/groupT.zip` auswählen



Wir benutzen HYPATIA um das Z-Boson zu finden





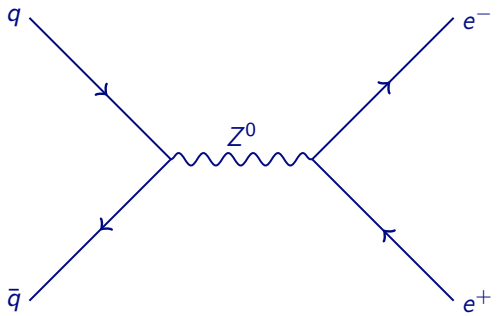
Z^0-Versuch

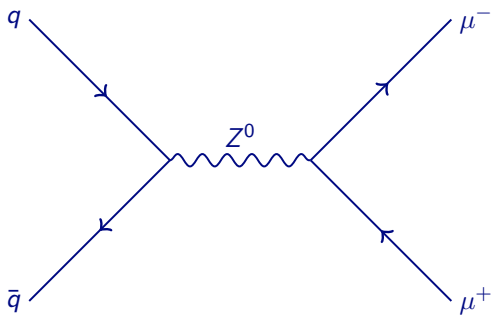
- Öffnen des Datenpakets aus `datenpakete/group?.zip`
- Für jedes Ereignis muss überprüft werden:

Ereignis und Objekt Auswahl

- ETMiss < 50 GeV
- Genau zwei gleichartige Leptonen mit $p_T > 20$ GeV
- Leptonen haben unterschiedliche Ladungen
- Leptonen sind isoliert von Jets
- Klassifiziere als $Z \to e^- e^+$ oder $Z \to \mu^- \mu^+$ Ereignis

- Ereignisse die mindestens ein Kriterium nicht erfüllen sind Untergrund Ereignisse
  $\to$ NEXT EVENT

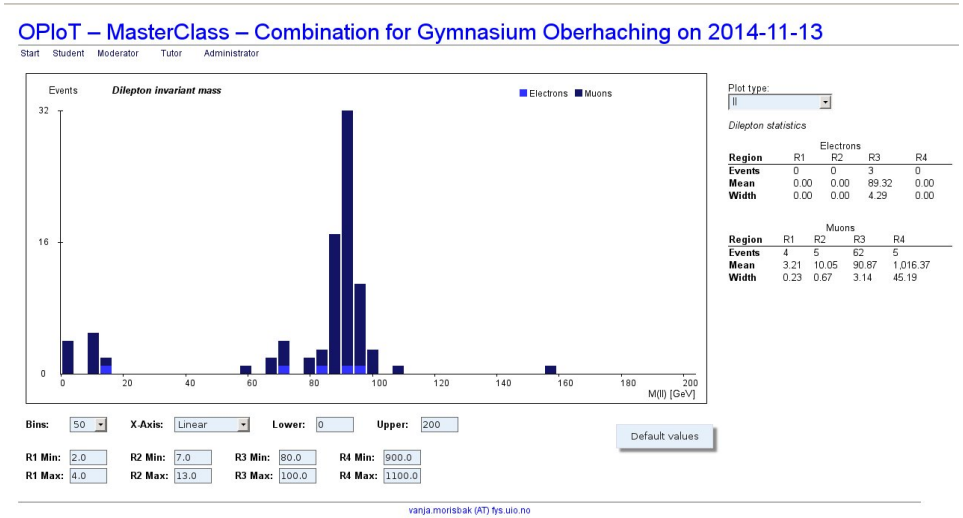




Auswertung

1) Exportieren der Massen

- Im Menü auf „File“ klicken
- auf „Export Invariant Masses“ klicken
- Dateinamen auswählen und speichern

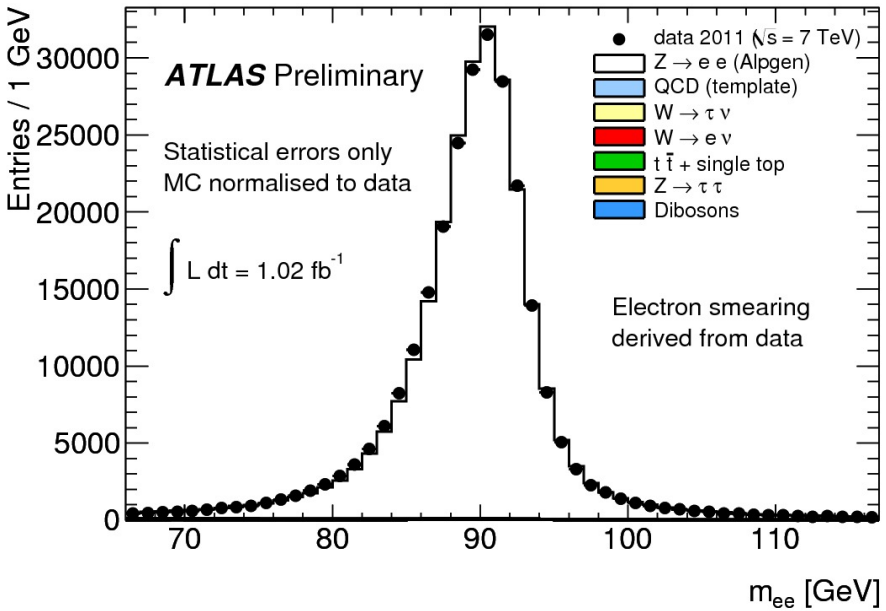


2) Datei hochladen

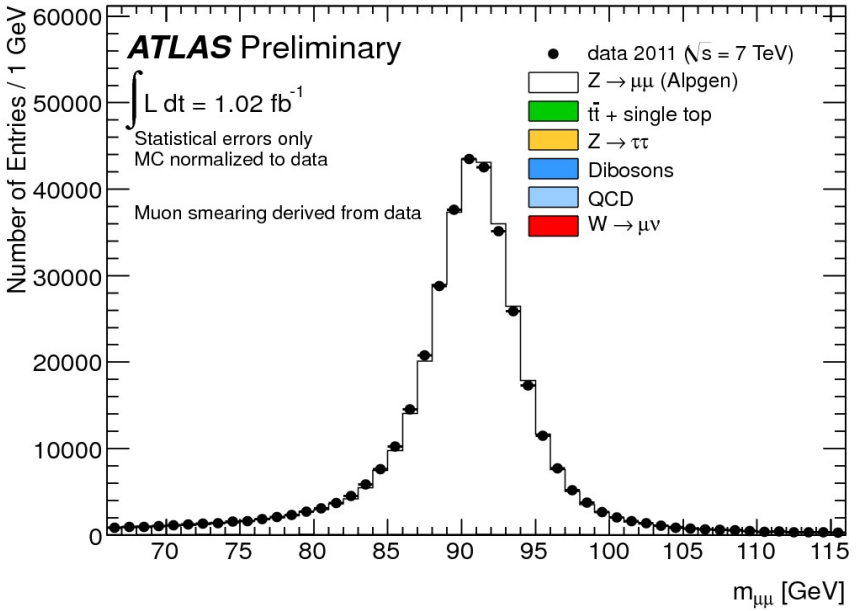
- auf **http://cernmasterclass.uio.no/** gehen
- „**OPlot**“ auswählen
- auf „**Student**“ klicken
- Username: „**ippog**“ / Password: „**mc13**“
- “LMU Workshop Datenauswertung“ und Gruppe 1 auswählen
- mit „**Browse**“ und „**Submit**“ eure Datei hochladen

ATLAS Ergebnis

Und was misst ATLAS?



$Z \to e^+e^-$



$Z \to \mu^+\mu^-$

Der aktuell gemessene Wert ist

$$m_Z = 91.1876 \pm 0.0021 \, GeV/c^2$$