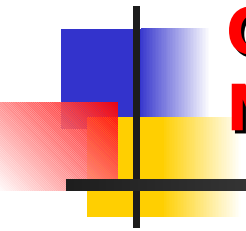


RunMC v3.2

C++ object-oriented framework for Monte Carlo models

S.Chekanov (ANL)

January 18, 2005

HERA-LHC workshop (CERN)

Introduction

- An analysis framework to run most popular Monte Carlo model

  (at this moment – written in FORTRAN)
- Good for validations, tuning, comparisons, calculations of correction factors (hadron-to-parton corrections etc.)
- Based on CLHEP
- Includes C++ physics libraries (jet algorithms, event-shape calculations) with unified input
- Fully integrated with the ROOT analysis environment

News

1) PHOJET (1.05) is included
   (in addition to PYTHIA,HERWIG,ARIADNE, CASCADE,AROMA,LEPTO)

2) Put most recent versions of HERWIG, PYTHIA, CASCADE (Jan 2005)

3) HZTOOL v3.0 is incorporated (as an external module)

4) JRunMC (beta version) for testing - RunMC GUI written in Java

5) Several example RMC modules for loading to RunMC

6) New web page (DESY + mirrored at ANL)

7) paper hep-ph/0411080(arXiv) - sent to Commp. Phys. Comm.

PHOJETv1.05 in RunMC

- Not supported by the author (Ralf Engel)
- Older version 1.05 was implemented:
  - PHOJET V1.1 has several bugs and cannot run at this moment
  - All steering cards – as they come with the original PHOJETv1.05 program
- Has unusual design – the event loop is unavailable for the user
  - To access it, requires significant changes in the code
- PHOJET was included with minimum modifications of the original code:
  - RunMC writes the event record first (HEPEVT record)
  - Then it analyses the events as usual
- The user will notice a little difference compared to the standard models based on event loops (<50k generated events)

New RMC modules

- Description
- Snapshot
- Structure
- Documentation
- Installation
- RMC modules
- Contributions

default.rmc	No any MC settings and physics calculations. Only dummy functions
dis_kinematics.rmc	DIS kinematic variables for HERA (Q2,x, etc), SC
charm_dis.rmc	Studies of D* cross sections in DIS (HERA), SC
dis_strange.rmc	Strangeness production (cross sections for K0s, Lambdas etc), SC
jets_HERA.rmc	Jets at HERA using longitudinally-invariant KT algorithm (Breit frame), SC
jets_LHC.rmc	Kt jets at LHC with charm, SC
jets+charm_LHC.rmc	Kt jets at LHC with charm, SC
event_shapes.rmc	Event shape studies, SC
par-had-jetLHC.rmc	Jet parton-to-hadron corrections, SC
sbumps.rmc	Search and identification of resonances, SC
hztool.rmc	HZTOOL calculations (v2.0), SC
hztoolv3.rmc	HZTOOL calculations (v3.0), SC
view3d.rmc	Look at 3D pictures of ttbar production at NLC (Geant imitation), SC
view3DjetsHERA.rmc	Look at 3D pictures of Kt jets at HERA (Geant imitation), SC
view3DjetsLHC.rmc	Look at 3D pictures of Kt jets at LHC (Geant imitation), SC

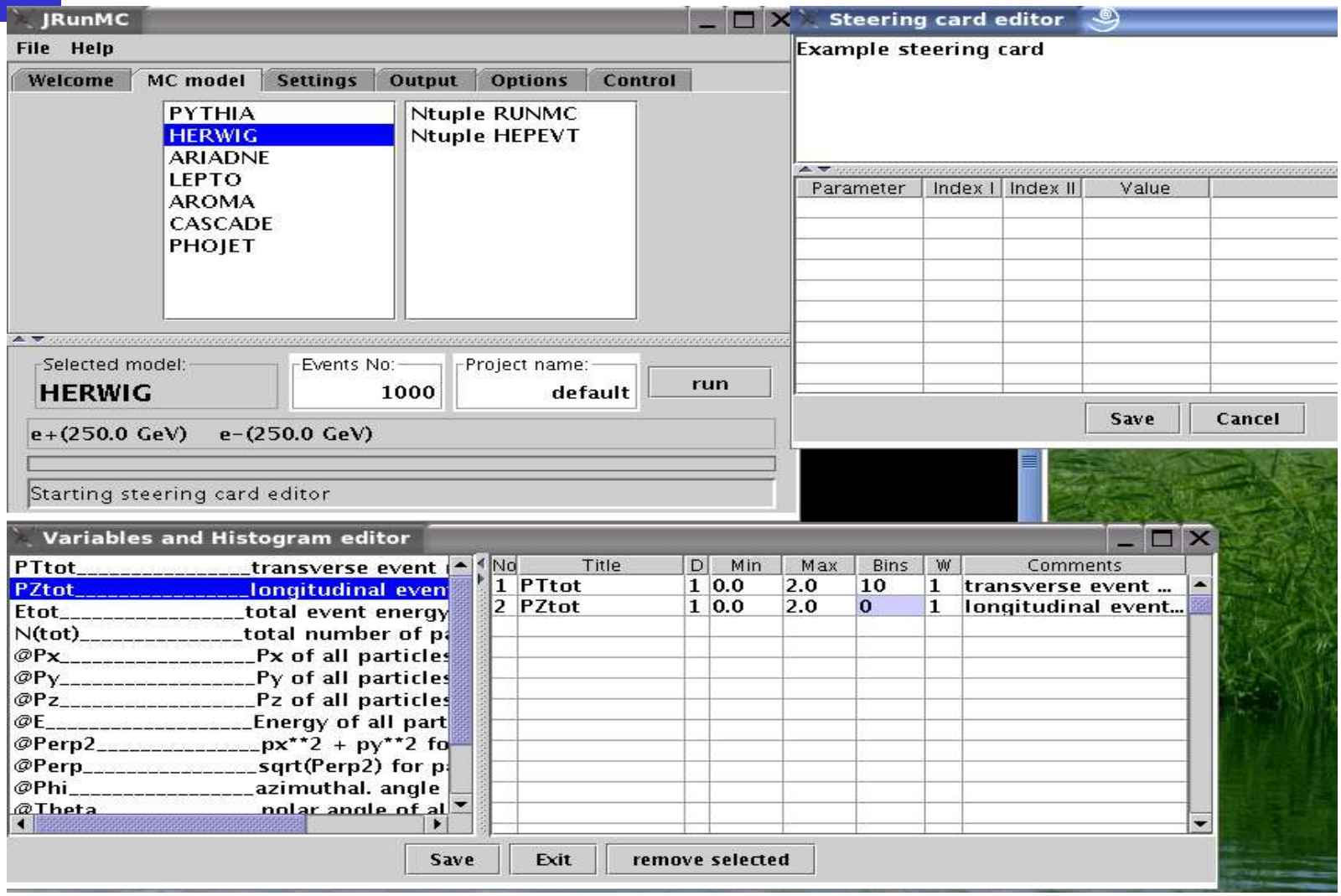
new

each 20-50 lines of the code (except for hztools)!

JRunMC GUI (Java based)

- To start default RunMC C++ GUI – use "runmc" command
- Main problems:
  - Difficult to maintain (not many people know WideStudio C++ library)
  - Large size - 7.8M after compilation
  - Takes long time to compile (~5 min, CPU 2 Ghz)
  - advantages – fast (compared to Java)
- Test version of GUI written in Java (v1.4) is available
  - JAR and source files come with the RunMC package (~120k only!)
  - Can incorporate future GUI for C++ MC models (ThePEG etc.)
  - Easy to debug, extend, use under Windows PC etc.
  - Test it by typing "jrunmc" (or java -jar $RUNMC/bin/JRunMC.jar)
- After testing, JRunMC will be the default GUI, and I'll stop maintaining C++ GUI (eventually I'll remove it from the package)

JRunMC GUI (Java based)



Physics analysis packages

- **Build-in physics libraries:**
  - HepLorentzVector (part of the RunMC event class)
  - Breit-frame calculations (for ep)
  - Event-shape calculations (C++ library by M.Iwasaki – ROOT based)
    - it also contains JADE & DURHAM algorithms
  - KTjet C++ library (J.Butterworth, J.Couchman, B.Cox, B.Waugh)
- **Physics libraries can also come together with loadable RMC modules**
  - can be written in C++ and/or Fortran
  - will be compiled automatically when a RMC module is loaded
  - Example: HzTOOLv3.0.rmc (see below)

RunMC : how to use C++ libraries

- **Two possibilities:**
  - **Method I:** Set up appropriate variables (load jet-HERA.rmc example)
    - Build-in physics libraries are used
    - No need a lot of programming
    - But not very flexible
  - **Method II:** Traditional method (load par-had-jetLHC.rmc)
    - Build-in or/and external physics libraries can be used
    - ***Any*** calculations can be implemented

Method I

Simplified RunMC-specific method (jet-HERA.rmc example):

1) Set "Jets" instead of "Particles" using RunMC GUI / JrunMC

2) Set ET(min) ET(max) etc. using RunMC GUI

3) Modify the user function user_afill.cpp

variable name (access from GUI)

```
// fill jet varibales (1-particle densities)
  if (nc1 != -1 && nc2 ==-1 && strcmp(type,"jet") == 0) {
  if    (strcmp(nnn,"Px(jets)") == 0)  { *getval=hep.PJ[nc1].p.px();}
  else if (strcmp(nnn,"Py(jets)") == 0)  { *getval=hep.PJ[nc1].p.py();}
  else if (strcmp(nnn,"Pz(jets)") == 0)  { *getval=hep.PJ[nc1].p.pz();}
  else if (strcmp(nnn,"E(jets)")  == 0)  { *getval=hep.PJ[nc1].p.e();}
  else if (strcmp(nnn,"ET(jets)")  == 0) { *getval=hep.PJ[nc1].p.perp(); }
  else if (strcmp(nnn,"Phi(jets)")  == 0)  { *getval=hep.PJ[nc1].p.phi(); }
  else if (strcmp(nnn,"Theta(jets)")  == 0) { *getval=hep.PJ[nc1].p.theta(); }
  else if (strcmp(nnn,"Eta(jets)")  == 0)   { *getval=hep.PJ[nc1].p.pseudoRapidity(); }
  else {
    return 1;
  };
  };
```

Lorentz-vector functions

4) The user does not have access to the C++ libraries – everything is done inside the RunMC program

5) Good for beginners

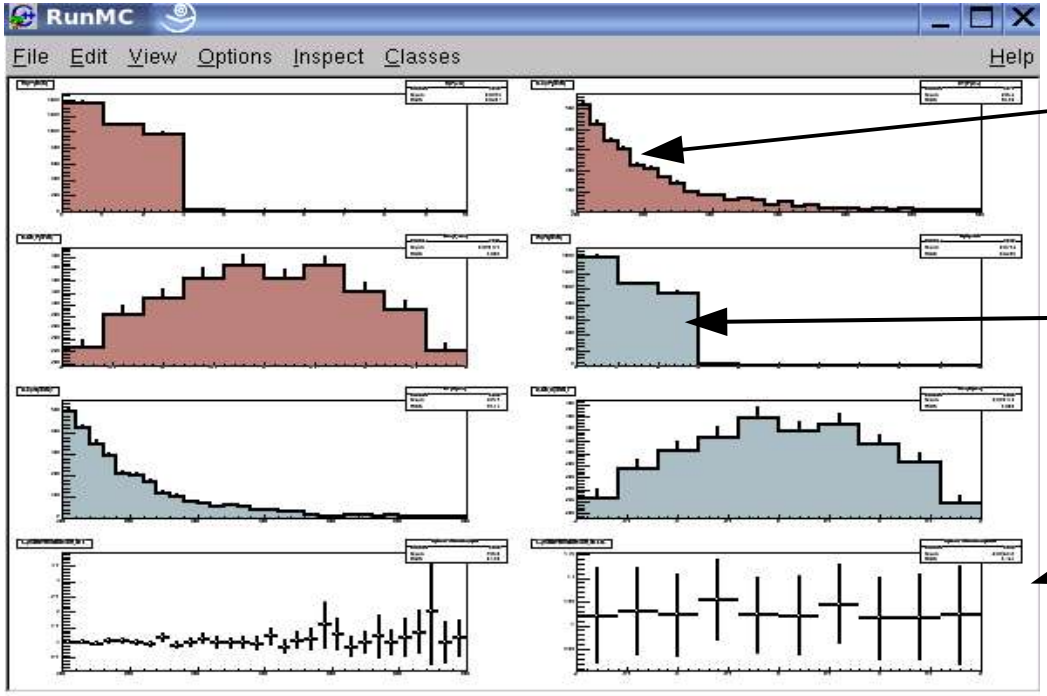
Method II

Traditional approach (par-had-jetLHC.rmc):

1) Full access to C++ libraries – good for complicated studies

2) RunMC GUI cannot control all details of calculations,
the user should modify initialization, analysis, and termination functions

Output of par-had-jetLHC.rmc module:



jets reconstructed from partons

jets reconstructed from partons

ratio
("hadron-to-parton" correction)

Method II

Example code (in the user function user-run.cpp):

```
// fill partons
    hep.fill_Partons();
// do KT jets
    KTjet(inival.ETmin, inival.ETAmin, inival.ETAmax);
    hp[1]->Fill( hep.PJ.size() );
    if (hep.PJ.size()>0) {
    for (unsigned int n=0; n<hep.PJ.size(); n++ ) {
        hp[2]->Fill( hep.PJ[n].p.perp() );
        hp[3]->Fill( hep.PJ[n].p.pseudoRapidity() );
        }; };

// fill final state
    hep.fill_ALLstable();
// do KT jets
      KTjet(inival.ETmin, inival.ETAmin, inival.ETAmax);
      hp[4]->Fill( hep.PJ.size() );
      for (unsigned int n=0; n<hep.PJ.size(); n++ ) {
        hp[5]->Fill( hep.PJ[n].p.perp() );
        hp[6]->Fill( hep.PJ[n].p.pseudoRapidity() );
       }
```

HzTOOL as an RMC module

Load the file hztoolv3.rmc to RUNMC and study the user directory "proj":

1) New directory "hztoolv3"
will be created with libhztoolv3.a

2) New runmchztool.f (FORTRAN!) subroutine is added which provides interface to RunMC.

3) Use conventional HzTOOL codding

4) In future RunMC could have different approach, i.e. based on C++ HzTOOL
In addition, HzTOOL Java GUI panel can be added to JRunMC where the user can select particular papers

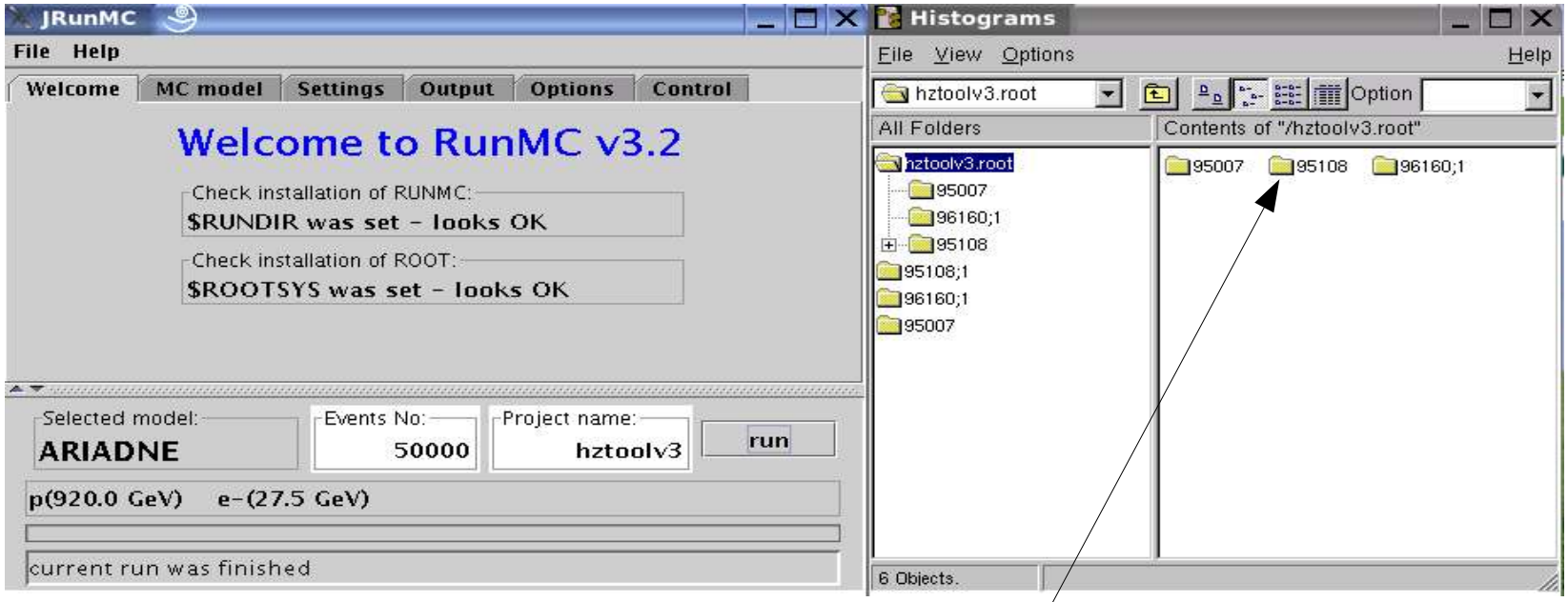
code inside "runmchztool.f":

```
      Call hropen(45,'HISTO',fname, 'n',1024,istat)
      Call HZ95108(1)
      Call HZ95007(1)
      Call HZ96160(1)
    else if (im .eq. 2) then
C run over events
      Call Hzfilhep ! fill HEPEVT
      Call HZ95108(2)
      Call HZ95007(2)
      Call HZ96160(2)
    else if (im .eq. 3) then
C termination
      Call HZ95108(3)
      Call HZ95007(3)
      Call HZ96160(3)
C do not change the lines below
      call hcdir('//PAWC',' ')
      call hcdir('//HISTO',' ')
      call hrout(0,icycle,'T')
      call hrend('HISTO' )
```

HzTOOL as an RMC module

- After loading hztoolv3.0.rmc file, start the run as usual
- RunMC will fill histograms using Fortran (HBOOK + HEPEVT-based)
- RunMC converts HBOOK histograms to Root and displays them:

Output of hztoolv3.0.rmc module:

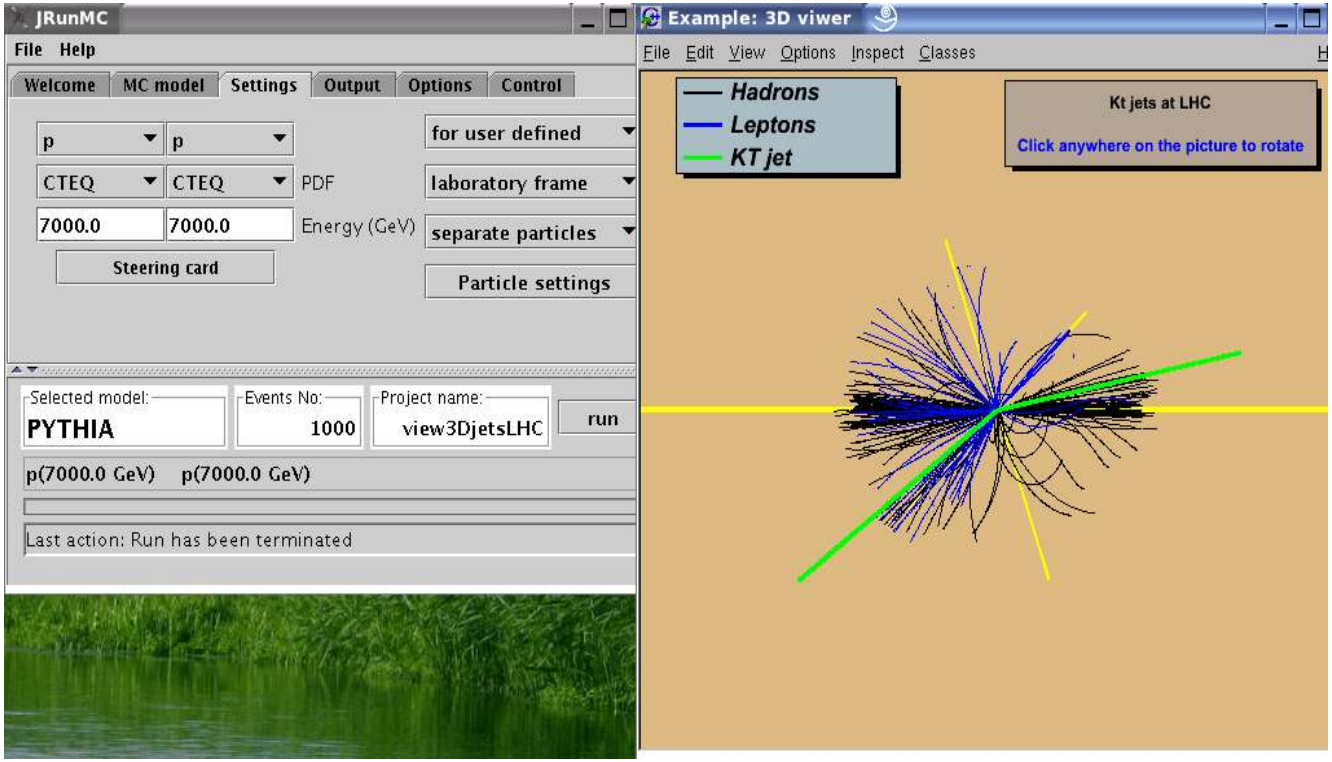


Click here to open ROOT histograms

RMC as an event gun

Example: view3DjetsLHC.rmc - converts RunMC into "3D LHC event gun"
Click "run" to view single events. Kt jets are indicated in green

try it for fun



good example of how to create user-defined canvas without using build-in RunMC canvas

Summary

Several changes done for RunMC (still version 3.2)

http://www.desy.de/~chekanov/runmc
http://www.hep.anl.gov/chakanau/runmc/

Can be done in the future:

- add RAPGAP model
- debug JRunMC
- add ThePEG (and write Java-based panels for JrunMC)
- can contribute to HzTOOL C++
- probably more RMC examples