PythiaB

interface to Pythia6 dedicated to simulation of beauty events.

Maria Smizanska (maria.smizanska@cern.ch)
University Lancaster

March 29, 2005

release 3.1.0 April 2002	PythiaBModule - algorithm and related code	new	M.Smizanska based on PythiaModule and atgenb
release 4.4.0 Oct.2002	PythiaBModule Event store at repeated hadronization bbbb, bbcc if forced decay	update	M.Muller, P.Calafiura M.Smizanska
	BSignalFiler - algorithm	new	M.Muller based on EventFilter and HistSample
release 5.0.0 2.Dec.2002	Btune.py	new	M.Smizanska Pythia6 B-tuned parameters
	PythiaB_Signal.py PythiaB_bbmu6X.py	updated ”	M.Smizanska ”
release 6.5.0 3.Aug.2003	PythiaBModule renamed to PythiaB replaced from former /Generators/GeneratorModules/PythiaBModule to /Generators/PythiaB	reorganization of Generators	G.Stavropoulos, I.Hinchcliffe
	PythiaB_Signal.py PythiaB_bbmu6X.py	updated to new Rndm Number service	M.Smizanska

Contents

1 Shortly

athena PythiaB_bbmu6X.py	simulate $b\bar{b} \rightarrow \mu(6\mathrm{GeV})X$
athena PythiaB_Signal.py	simulate one exclusive B-channel default $B_s \rightarrow J/\psi(\mu\mu)\phi$ other examples inside PythiaB_Signal.py

2 What PythiaB provides

PythiaB provides an interface to Pythia6 allowing to: **1.** speed up B events simulations, **2.** simulate only wanted decay channel, **3.** apply selection cuts organized in several levels: after parton showering (before hadronization), after hadronization: trigger-like cuts and off-line type of cuts, **4.** define b-production parameters - optimal parameters are prepared as default. User can control 1.-4. by datacards contained in jobOptions.py file.

PythiaB inherits from Pythia and GenModule, so the event is converted from Pythia common block HEPEVT to c++ HepMc tree structure and stored in Storegate as a transient object and finally written to a root persistency.

3 Where to use PythiaB

PythiaB can be used for a generation of beauty events written into a persistent root file, which can serve as an input into a detector simulation (atlsim) or into ATFAST. PythiaB can also work in the same job with ATLFAST in this case the HepMC tree is passed from PythiaB to ATLFAST in a transient form.

4 How to use PythiaB

4.1 Just changing jobOptions.py, linking only PythiaB files

To work just with PythiaB avoiding to connect large number of unwanted files to your TestRelease area you simply do following:

1. set up athena under cmt, see cmt manual [1] and Athena manual [2].

2. cmt co TestRelease
3. cd TestRelease/TestRelease-00-*/cmt
4. edit requirements and change following lines:

```
    O R I G I N A L

 use AtlasRelease AtlasRelease-*



    R E W R I T T E N

#use AtlasRelease AtlasRelease-*
use PythiaB PythiaB-00-* Generators
use AthenaCommon AthenaCommon-* Control
use AtlasROOT           AtlasROOT-*        External
use EventHdrAthenaRoot EventHdrAthenaRoot-* Event
use EventSelectorAthenaRoot EventSelectorAthenaRoot-* Database/AthenaRoot
```

5. source setup.sh
6. gmake
7. cd TestRelease/TestRelease-00-*/run
8. edit and adopt jobOptions.py to your needs
9. to run interactively: athena jobOptions.py >output.log

4.2 Changing an existing Athena code

I want to change some of the existing code, e.g. apply dedicated selection criteria in user_finsel.F, or change PythiaB.cxx. Then do following:

1. set up Athena under cmt
2. cmt co TestRelease
3. cmt co Generators/PythiaB
4. cd Generators/PythiaB/src/
5. edit and adopt user_finsel.F
6. edit and adopt PythiaB.cxx
7. cd TestRelease/TestRelease-00-*/cmt
8. change the default requirements to the requirements as written at the item 4 of the subsection 4.1
9. source setup.sh
10. cmt broadcast gmake
11. cd TestRelease/TestRelease-00-*/run
12. (edit and adopt jobOptions.py if you need)

13. to run interactively: athena jobOptions.py > output.log

The release 6.5.0 allows also running in the InstallArea, so you can as well do the last three points as follows:

11. cd InstallArea/jobOptions/PythiaB
12. (edit and adopt jobOptions.pyif you need)
13. to run interactively: athena jobOptions.py > output.log

4.3 Changing an existing Pythia6 subroutine

I have my own physics model in fortran. I want to pluck it into an existing PYTHIA6 subroutine PYxyz.F. It is similar to previous case. So you do following:

1. set up Athena under cmt
2. cmt co TestRelease
3. cmt co Generators/PythiaB
4. cd Generators/PythiaB/src/
5. cp $MYSPACE/PYxyz.F ./
6. cd Generators/PythiaB/cmt/ edit requirements and change following lines:

```
O R I G I N A L

                                   charm.F \
                                   children.F \



R E W R I T T E N
                                   charm.F \
          PYxyz.F  \
                                   children.F \
```

7. cd TestRelease/TestRelease-00-*/cmt
8. change the default requirements to the requirements as written at the point 4 of the subsection 4.1
9. source setup.sh
10. cmt broadcast gmake
11. cd TestRelease/TestRelease-00-*/run
12. (edit and adopt jobOptions.py if you need)

13. to run interactively: athena jobOptions.py > output.log

or

11. cd InstallArea/jobOptions/PythiaB
12. (edit and adopt jobOptions.py if you need)
13. to run interactively: athena jobOptions.py > output.log

5 Why dedicated code for b-physics

5.1 Speed up simulation

Pythia6 is a phenomenological model that provides three mechanisms to produce b-quark. They are classified as a flavour creation (gg → b$\bar{\text{b}}$, qq → b$\bar{\text{b}}$), flavour excitation (gb → gb) and gluon splitting ($g \rightarrow b\bar{b}$). All these three mechanisms are activated if following Pythia processes are allowed: isub=11 f + f' → f + f' (QCD) , isub=12 f + fbar → f' + fbar' , isub=13 f + fbar → g + g , isub=28 f + g → f + g , isub=53 g + g → f + fbar , isub=68 g + g → g + g , with parameter "pysubs msel=1". In this regime beauty quark is produced approximately in 1% of events (the number depends on the selected phase space). There is another mechanism which is activated by selecting "pysubs msel 5" - here b-quark is produced in each event in a hard scattering processes of type: gg → b$\bar{\text{b}}$, qq → b$\bar{\text{b}}$ using mass matrix elements. This mechanism however does not describe known b-production data from Fermilab. The difference is so large that it is not suggested to use it for simulations. The details can be found for instance in [5].

In order to speed up the simulation in "pysubs msel 1" mode PythiaB interrupts a simulation after the parton development (just before the hadronization) to check for the presence of $b\bar{b}$ quarks satisfying user's defined limits in p_T and η. If user wants the hadronization is repeated several (MHAD) times using the same parton part of the event. The resulting cross section is then divided by MHAD. It is suggested that user selects MHAD such that the average number of accepted events with the same parton part is close to 1. The latter value is printed out by PythiaB in an output log file.

5.2 Simulate only wanted channel

It is not difficult to select only wanted channel in Pythia. PythiaB does nothing special - just saves your time and work: 1. by providing an interface to datacards allowing user to close and open channels without an intervention in the code, 2. by providing special datacard files allowing to close large groups of unwanted channels. For more details see section 6.2. PythiaB provides also datacards allowing user to apply trigger-like cuts and offline-like selection cuts.

5.3 How should I calculate a cross section

If a user did not force any of B-decays, then the cross section is calculated in PythiaB and appeares in the output.log file under the name: CROSSSECTION OF YOUR B-CHANNEL IS. If you forced any channel in B-decay chain you should multiply the PythaBModule cross section by a branching ratio for this channel.

MEANING	ATLAS VALUE	Pythia6 default
b-quark production-related parameters		
Structure fuction	"pypars mstp 51 1 (CTEQ3)"	CTEQ5
Min bias	"pysubs msel 1",	1
Max parton virtuality		
factor to multiply Q^2_{hard}	"pypars parp 67 1",	1
The factorization scale Q^2_{hard} =		
${p_T}^2(P_1^2+P_2^2+m_3^2+m_4^2)/2$	"pypars mstp 32 8"	8
B hadron related parameters		
Spin s=1 probability	"pydat1 parj 13 0.65"	0.75
j=1 l=1 s=0	"pydat1 parj 14 0.12",	0
j=0 l=1 s=1	"pydat1 parj 15 0.04",	0
j=1 s=1 l=1	"pydat1 parj 16 0.12",	0
j=2 s=1 l=1	"pydat1 parj 17 0.2",	0
Peterson fragmentation ϵ_b	"pydat1 parj 55 -.006"	-.005
No B-oscillations	"pydat1 mstj 26 0",	
Multiple interactions parameters		
Model	"pypars mstp 82 4 (double gauss)",	1 (step fcn)
Regularization p_T scale	"pypars parp 82 1.4",	1.9
Double gauss parameters	"pypars parp 83 0.5",	0.5
"	"pypars parp 84 0.4",	0.2
Gluon probability	"pypars parp 85 0.9",	0.33
Two gluon probability	"pypars parp 86 0.95",	0.66
Energy scale for parp 82	"pypars parp 89 1800",	1000
Power of energy rescaling	"pypars parp 90 0.25".	0.16

Table 1: The optimized set of B-physics-related Pythia6 parameters, values for ATLAS simulations.

5.4 Optimal B-simulation parameters

PythiaB provides an interface to datacards allowing user to define beauty production parameters. The optimal values of these parameters were selected in [6] and are summarized in the Table 1. This set is provided by PythiaB as a default. The new structure functions

CTEQ5 ('mstp 51 7') fails to describe the b-production data while the older CTEQ3 ('mstp 51 1') fits fairly well [6]. The reason is that Pythia contains many parameters that influence the b-production and a complex tuning have been done by phenomenologists only for CTEQ3, [3, 4]. For CTEQ5 the job still needs to be done. Such a job exceeds possibilities of our group. Currently we recommend user to use 'mstp 51 1' in a combination with the other parameter values of the optimized set provided by PythiaB.

6 Datacards

In Athena a user controls the simulation using the datacards contained in files jobOptions.py. For PythiaB you will find in release two prepared files: PythiaB_bbmu6X.py and PythiaB_Signal.py from which you can easily derive your case. File PythiaB_bbmu6X.py simulates events $b\bar{b} \rightarrow \mu 6X$. File PythiaB_Signal.py simulates one exclusive B-channel (default is for $B_s \rightarrow J/\psi(\mu\mu)\phi$).

6.1 Job control Datacards

`Generator.Members = [`	
`"PythiaB",`	Simulate b-event in HepMc into transient store
`"DumpMC" ,`	Dump HepMc for each accepted event into log file do not use with large statistics
`"BSignalFilter"]`	Finds all B-decay chains and store in ntuple for print level=2 also dumps B chains into log file
`ApplicationMgr.EvtMax = 10`	Number of events to be accepted
`EventSelector.RunNumber = 1;`	Defines a Run numb. in the job
`EventSelector.FirstEvent  = 1;`	Defines the first evt numb. in the job
`AtRndmGenSvc.Seeds = ["PYTHIA 4789899 989240512", "PYTHIA_INIT 820021 2347532"]`	User random number seeds

6.2 Opening - closing decay channels

	Channel

`#include "CloseAntibQuark.py";`
`PythiaB.PythiaCommand+=[`
`"pydat3 mdme 1120 1 1",`
`"pydat3mdme 996 1 0",`
`"pydat3 mdme 998 1 0" ]`
	$\mathrm{B_s} \to \mathrm{J}/\psi(\mu\mu)\phi$

`#include "CloseAntibQuark.py";`
`#include"Dsphipi.py";`
`PythiaB.PythiaCommand += [`
`"pydat3 mdme 1105 1 1"]`
	$\mathrm{B_s} \to \mathrm{D_s}(\phi\pi)\pi$

`#include "CloseAntibQuark.py";`
`PythiaB.PythiaCommand+= [`
`"pydat3 mdme 1027 1 1",`
`"pydat3 mdme 996 1 0",`
`"pydat3 mdme 998 1 0" ]`
	$\mathrm{B_s} \to \mathrm{J}/\psi(\mu\mu)\mathrm{K}^0(\pi^+\pi^-)$

Be carefull: there should be just one space between numbers in 'datacards mdme'. If you put two or more - program will not complain, but ignore your 'datacard'.

6.3 How to change Pythia parameters

The optimized Pythia6 B-physics-related parameters (Table.1) are defined in ATHENA by the file Btune.py, which is included in the standard PythiaB_Signal.py and PythiaB_bbmu6X.py. To change any value you can just copy a corresponding line from Btune.py into PythiaB_Signal.py placing it after the line `#include "Btune.py"` and changing the value of parameter to a wanted one. It is suggested that the large background samples of general use are always generated using the values defined in Btune.py.

6.4 Define your selection cuts

`PythiaB.cutbq = ["7. 3.5 or 7. 3.5"]`	b-quark cuts: pT, eta, and(or) antib-quark cuts: pT, eta
`PythiaB.lvl1cut = { 1., 6., 2.5};`	lvl1 single muon cuts: switch on(1)/off(0) pT eta
`PythiaB.lvl2cut = [ 0., 13., 6., 2.5]`	lvl2 second lepton cuts: switch on(1)/off(0), PID(13mu or 11e),pT, eta
`PythiaB.offcut = [ 1., 0.5, 2.5, 0.5, 2.5, 0.5, 2.5]`	offline cuts: applied on stable particles at the end of B-decay chain switch on(1)/off(0), pT, eta of K/pi/p, pT, eta of muon, pT, eta of electron
`PythiaB.mhadr = 10. ;`	number of repeated hadronizations select MHAD such that the average number of accepted events with the same parton part is close to 1. The latter value is printed out by PythiaB in the output log file.
`BSignalFilter.SignaltoNtup = 50.;`	For how many events store B-chains into NTUPLE
`PythiaB.SignalDumptoAscii = 0.;`	For how many events store exclusive signal B-chain into ascii fort.50
`PythiaB.flavour = 5.;`	wanted heavy flavour b(5) or c(4-not yet installed)

6.5 Data cards to define output files

Data cards to define output files are in the section **Output files**.

7 Output files

7.1 NTUPLE file for B-decays chains

```
//----------------------------------------------------------------
// NTuple output file
//----------------------------------------------------------------

NTupleSvc.Output = [ "FILE1
                      DATAFILE='pythiaB.ntup'
                      OPT='NEW'
                      TYP='HBOOK'" ]
```

For each secondary particle from a B-signal and any other B-decay chain in the event a following values in column-wise Ntuple are written:

```
"particles",
"event_number",
"chain_number",
"particle_number",
"id",
"status",
"child_of",

"px",
"py",
"pz",
"pe",
"pt",
"mass",

"phi",
"rapidity",
"pseudorapidity"
```

7.2 B-signal chain, ascii file

Formated ascii file fort.50 containing lines each with one particle from a B-signal chain. Was used before BSignaFilter was written. The file is created automatically if user defines non zero values in PythiaB.SignalDumptoAscii = 10. and in PythiaB.offcut = 1., ...

Each line contains following information:

```
float(ieve) ! this accepted ev number
float(ITB)  ! # of evs hadronised to get this one
float(ntree)! # of particles in B-chain
float(itree)! this particle number in Bchain
float(I)    ! this particle number in PYJET

float(K(I,1))
float(K(I,2))
float(K(I,3))
float(K(I,4))
float(K(I,5))
```

```
P(I,1)
P(I,2)
P(I,3)
P(I,4)
P(I,5)

V(I,1)
V(I,2)
V(I,3)
V(I,4)
V(I,5)
```

7.3 Output log file

Contains Pythia messages. List of all input datacards. Pylist12 - list of all decay channels including status: opened-closed. Cross section table. Summary of principle parameters and cuts. Here is a part of log file:

```
==============================================================================
I                                  I                            I            I
I            Subprocess            I      Number of points      I    Sigma   I
I                                  I                            I            I
I----------------------------------I----------------------------I    (mb)    I
I                                  I                            I            I
I N:o Type                         I    Generated         Tried I            I
I                                  I                            I            I
==============================================================================
I                                  I                            I            I
I   0 All included subprocesses    I         3095         32166 I  1.138E+00 I
I  11 f + f' -> f + f' (QCD)       I           72           783 I  2.631E-02 I
I  12 f + fbar -> f' + fbar'       I            3             6 I  3.266E-04 I
I  13 f + fbar -> g + g            I            2             4 I  4.345E-04 I
I  28 f + g -> f + g               I          818         10581 I  2.974E-01 I
I  53 g + g -> f + fbar            I           67           262 I  2.491E-02 I
I  68 g + g -> g + g               I         2133         20530 I  7.886E-01 I
I                                  I                            I            I
==============================================================================

********* Fraction of events that fail fragmentation cuts =  0.00000 *********
```

```
I==============================================================================
I   CROSSSECTION OF YOUR B-CHANNEL IS                    I
I                       BX= PX*NB/AC/MHAD=               I    2.77963e-05 mbarn
I                                                        I
I   IN CASE YOU FORCED ANY DECAY YOU SHOULD              I
I   CORRECT CROSS SECTION BX FURTHER, MULTIPLYING        I
I   BX BY BRANCHING RATIO(S) OF YOUR FORCED              I
I   DECAY(S) AND BY A FACTOR OF 2 FOR SYMMETRY           I
I                                                        I
I   MORE DETAILS ON CROSS SECTION                        I
I   PYTHIA MSEL=1     CROSS SECTION IS    PX=            I    1.13794 mbarn
I   NUMBER OF   ACCEPTED  MSEL=1 EVENTS   AC=            I    32166
I   NUMBER OF   ACCEPTED    B-EVENTS   IS NB=            I    11
I   REPEATED HADRONIZATIONS IN EACH EVENT MHAD=          I    14
I   AVERAGE NUM OF ACCEPTED EVTS IN HADRONIZATION LOOP  I    1.1
I   IN CASE YOU FORCED ANY DECAY YOU SHOULD              I
I   CORRECT CROSS SECTION BX FURTHER, MULTIPLYING        I
I   BX BY BRANCHING RATIO(S) OF YOUR FORCED              I
I   DECAY(S) AND BY A FACTOR OF 2 FOR SYMMETRY           I
I                                                        I
I==============================================================================

I==============================================================================
I          YOUR  MAIN SIMULATION PARAMETERS AND CUTS
I==============================================================================
I   HARD SCATTERING  CUT  pysubs().ckin(3)            PT      I 15
I   STRUCTURE FCN (1=CTEQ3 7=CTEQ5) pypars().mstp(51)        I  1
I   CUTS ON b and/or anti b QUARK                            I  0 ; 102.5 ; and ; 10
I   LVL1 MUON CUTS:                              PT AND ETA   I  6 ;  2.5
I   LVL2 CUTS:         ON(1)/OFF(0); PARTICLE-ID; PT AND ETA   I  0 ;  13 ;  6 ;  2.5
I   CUTS FOR STABLE PARTICLES IN B-DECAY:        ON(1)/OFF(0) I  1
I    CHARGED HADRONS:                            PT AND ETA   I  0.5 ;  2.5
I    MUONS:                                      PT AND ETA   I  3 ;  2.5
I    ELECTRONS:                                  PT AND ETA   I  0.5 ;  2.5
I==============================================================================
```

7.4 AthenaRoot Persistency

Produced Pythia B events are stored in root persistent file PythiaB.root. This file serves as an input to atlsim (detector simulation) or atlfast.

```
//------------------------------------------------------------
```

```
// AthenaRoot Persistency
//----------------------------------------------------------------

ApplicationMgr.DLLs += [ "EventHdrAthenaRoot", "GeneratorObjectsAthenaRoot"]
ApplicationMgr.DLLs += [ "RootSvcModules", "EventSelectorAthenaRoot", "AthenaRootCnvSvc"
ApplicationMgr.ExtSvc += [ "RootSvc", "AthenaRootCnvSvc" ]
ApplicationMgr.DLLs += ["McEventSelector"]
ApplicationMgr.ExtSvc += [ "McCnvSvc", "McEventSelector/EventSelector" ]
EventPersistencySvc.CnvServices += [ "McCnvSvc" ]
ApplicationMgr.OutStream = [ "Stream1" ]
ApplicationMgr.OutStreamType = "AthenaOutputStream";
// for StoreGate
Stream1.ItemList = [ "2101#*", "133273#*"]
RootSvc.Output =["PythiaB.root"]
EventPersistencySvc.CnvServices += [ "AthenaRootCnvSvc" ]
Stream1.EvtConversionSvc = "AthenaRootCnvSvc"
```

7.5 Which output files we store in CASTOR

B-physics group has a space in CASTOR /afs/cern.ch/atlas/project/bphys/. For password into this area contact maria.smizanska@cern.ch. Following files are to be stored:

- PythiaB.root
- pythiaB.ntup
- output.log

8 List of all ATHENA files dedicated to B-physics simulation

- PythiaB/src/PythiaB.cxx
- PythiaB/src/PythiaB_entries.cxx
- PythiaB/src/PythiaB_load.cxx
- PythiaB/src/*.F
- PythiaB/PythiaB/PythiaB.h
- PythiaB/share/PythiaB_Signal.py
- PythiaB/share/PythiaB_bbmu6X.py
- PythiaB/share/Btune.py
- PythiaB/share/CloseAntibQuark.py
- PythiaB/share/Dsphipi.py

  ?

- GeneratorFilters/src/BSignalFilter.cxx
- GeneratorFilters/GeneratorFilters/BSignalFilter.h

9 Classes PythiaB inheriths from

Pythia, GenModule

References

[1] ATLAS software developers guide,

`http://atlas-sw.cern.ch/cgi-bin/cvsweb.cgi/~checkout~/offline/AtlasDoc/doc/SwDevUserGuide/userguide.html`

and cmt manual

`http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/tools/cmt/Tutorial.html`

[2] Athena User Guide,

`http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/architecture/General/index.html`

[3] T. Sjostrand et al, PYTHIA 6.206 manual, LU TP 01-21 [hep-ph/0108264],

[4] E.Norrbin, QCD phenomenology of Heavy particle dynamics, PhD theses, Lun University, Oct.2000.

[5] S.P.Baranov, M.Smizanska, Phys.Rev.D62:014012,2000 and ATL-PHYS-98-133.

[6] Pythia6 Tuning for B-physics simulations in ATLAS. NOTE under the preparation.