An Introduction to Using EvtGen

Anders Ryd
March 2, 2004

Alea iacta est, "The die is cast", Julius Ceasar Jan. 10, 49BC as he crossed Rubicon.

An Introduction to Using EvtGen

Anders Ryd
Cornell U.
March 2, 2004

Outline:

- What problems EvtGen can and can not solve
- Selection algorithm
- Physics processes that are implemented
- How to write your own physics modules
- Using EvtGen in CLEO-c

This talk borrows much from a tutorial D. Lange and I gave for an LHC generator workshop in the summer of '03

Motivation

- Why should you care about EvtGen in CLEO-c?
  - We have been generating the 'million MC' samples using Evtgen.
  - Further simulation of hadronic decays in CLEO-c will be done with EvtGen.
  - Continuum simulated via EvtGen using the “Lund Area Law”.
  - Radiative return events are simulated via EvtGen.
  - EvtGen is integrated in the suez framework.

Sequential decays

- Many decays have interesting sequential decay chains:

$$B \to D^* \ell \nu \qquad \qquad B \to D^* \ell \nu$$
$$\hookrightarrow D\pi \qquad \qquad \hookrightarrow D\gamma$$

$$B \to D^* \; D^* \qquad \qquad B \to D^* \; D^*$$
$$\hookrightarrow D\pi \hookrightarrow D\pi \qquad \qquad \hookrightarrow D\pi \hookrightarrow D\gamma$$

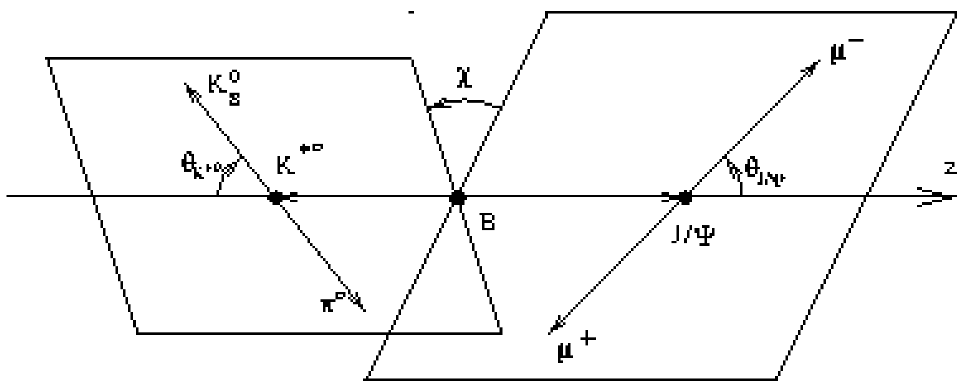
- Want to correctly simulate these decay chains while only implementing the nodes in the decay tree.

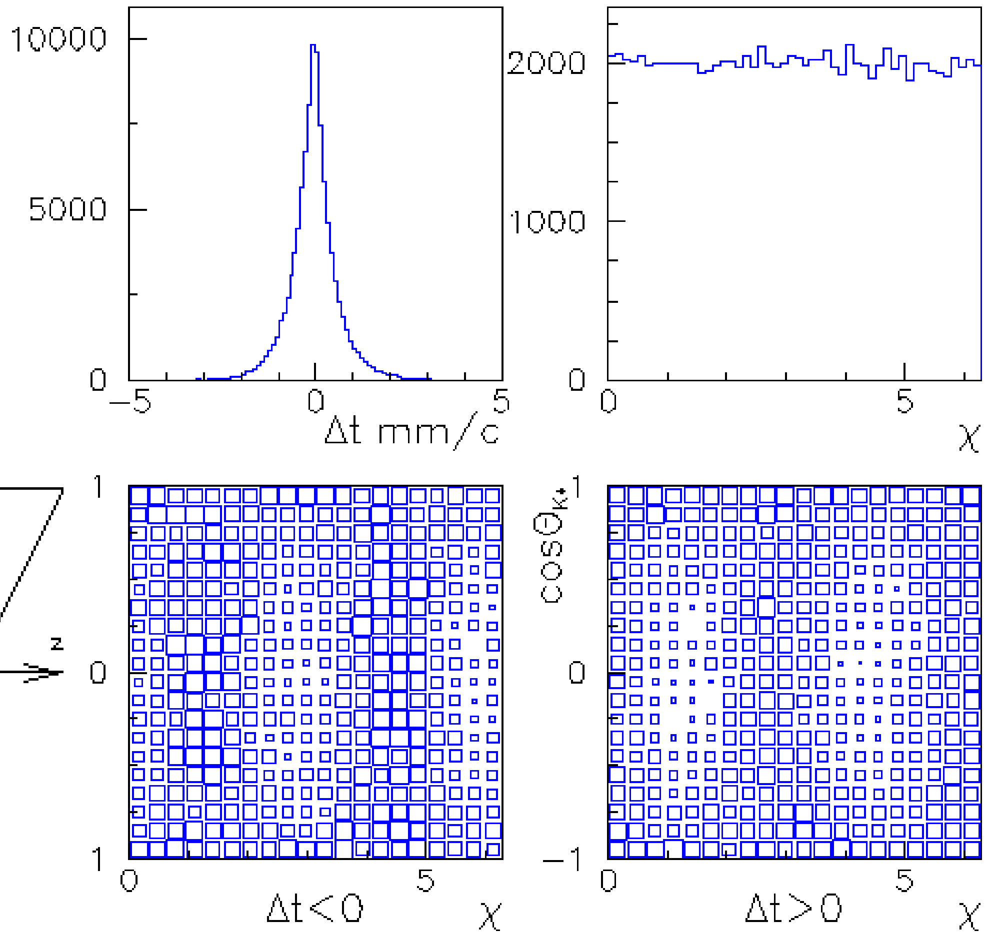
ıc nodes in the decay tree:

$$B \to D^* \ell \nu \qquad B \to D^* D^*$$
$$D^* \to D\pi \qquad D^* \to D\gamma$$

CP violating decays

- *$B\text{->}J/\psi K^{*0}$ ($K^{*0}\text{->}K^0\pi^0$)*
  - Angular correlations and time dependence





Decay amplitudes are used instead of probabilities

- EvtGen works with amplitudes to correctly handle sequential decays:

$$B \to D^* \quad \tau\nu$$
$$\hookrightarrow D\pi \;\; \hookrightarrow \pi\nu$$

$$d\Gamma = |A|^2 d\phi \qquad A = \sum_{\lambda_{D^*}\lambda_\tau} A^{B\to D^*\tau\nu}_{\lambda_{D^*}\lambda_\tau} A^{D^*\to D\pi}_{\lambda_{D^*}} A^{\tau\to\pi\nu}_{\lambda_\tau}$$

$$A^{B\to D^*\tau\nu}_{\lambda_{D^*}\lambda_\tau} \equiv \langle \lambda_{D^*}\lambda_\tau | H | B \rangle$$

sis: $\sum_{\lambda_{D^*}} |\lambda_{D^*}\rangle\langle\lambda_{D^*}| = I$

- Nodes in the decay tree are implemented as “models”. The framework of EvtGen handles the bookkeeping needed to correctly generate the full decay tree.

Selection algorithm (I)

- Generate the *B->D*lν* decay

$$P = \sum_{\lambda_{D^*}\lambda_\tau} |A^{B\to D^*\tau\nu}_{\lambda_{D^*}\lambda_\tau}|^2$$

num probability and acccpt

- Compare with maximum probability and accept or reject generated *B->D*lν* decay.
  - Maximum probability specified in code.
    - Can instead be generated on the fly, however this leads to the output of event *N* depending on the random number sequence used to determine the max probability.
- Regenerate *B->D*lν* decay until combination is accepted.

Selection algorithm (II)

- Average over τ spin and calculate the *D** spin density matrix:

$$\rho^{D^*}_{\lambda_{D^*}\lambda'_{D^*}} = \sum_{\lambda_\tau} A^{B\to D^*\tau\nu}_{\lambda_{D^*}\lambda_\tau}\left(A^{B\to D^*\tau\nu}_{\lambda'_{D^*}\lambda_\tau}\right)^*$$

: $D^* \to D\pi$ decay

- Generate the $D^*\text{->}D\pi$ decay

$D^* \to D\pi$ decay

$$P = \sum_{\lambda_{D^*}\lambda'_{D^*}} \rho^{D^*}_{\lambda_{D^*}\lambda'_{D^*}} A^{D^*\to D\pi}_{\lambda_{D^*}}\left(A^{D^*\to D\pi}_{\lambda'_{D^*}}\right)^*$$

- Compare with maximum probability and accept or reject generated $D^*\text{->}D\pi$ decay
- Regenerate $D^*\text{->}D\pi$ decay until accepted. The $B\text{->}D^* l\nu$ decay is not regenerated.

Selection algorithm (III)

- Calculate the spin density matrix for the τ

$$\rho^{\tau}_{\lambda_\tau \lambda'_\tau} = \sum_{\lambda_{D^*} \lambda'_{D^*}} \hat{\rho}^{D^*}_{\lambda_{D^*} \lambda'_{D^*}} A^{B\to D^*\tau\nu}_{\lambda_{D^*} \lambda_\tau} (A^{B\to D^*\tau\nu}_{\lambda'_{D^*} \lambda'_\tau})^*$$

- Where:

$$\hat{\rho}^{D^*}_{\lambda_{D^*} \lambda'_{D^*}} \equiv A^{D^*\to D\pi}_{\lambda_{D^*}} (A^{D^*\to D\pi}_{\lambda'_{D^*}})^*$$

c the $\tau \to \pi\nu$ decay

- Generate the $\tau \to \pi\nu$ decay

$$P = \sum_{\lambda_\tau \lambda'_\tau} \rho^{\tau}_{\lambda_\tau \lambda'_\tau} A^{\tau\to\pi\nu}_{\lambda_\tau} (A^{\tau\to\pi\nu}_{\lambda'_\tau})^*$$

- Compare with maximum probability and accept or reject generated $\tau \to \pi\nu$ decay.
- Regenerate $\tau \to \pi\nu$ decay until accepted. The *B->D*l*ν and *D*->D*π decays are not regenerated.

Advantages to using decay amplitudes

- Implementation of decay models is simplified by using amplitudes instead of probabilities.
- Keeping track of the spin density matrices allows us to generate each node of the decay chain independently.
  - More efficient
  - Avoids the need to determine uncountable # of maximum probabilities
- Generalizes to arbitrarily long decay chains
- Calculation of probabilities and spin density matrices are done by the framework. Models specify only the decay amplitudes.
- However: No interference between particles on different branches of decay tree.

States in EvtGen

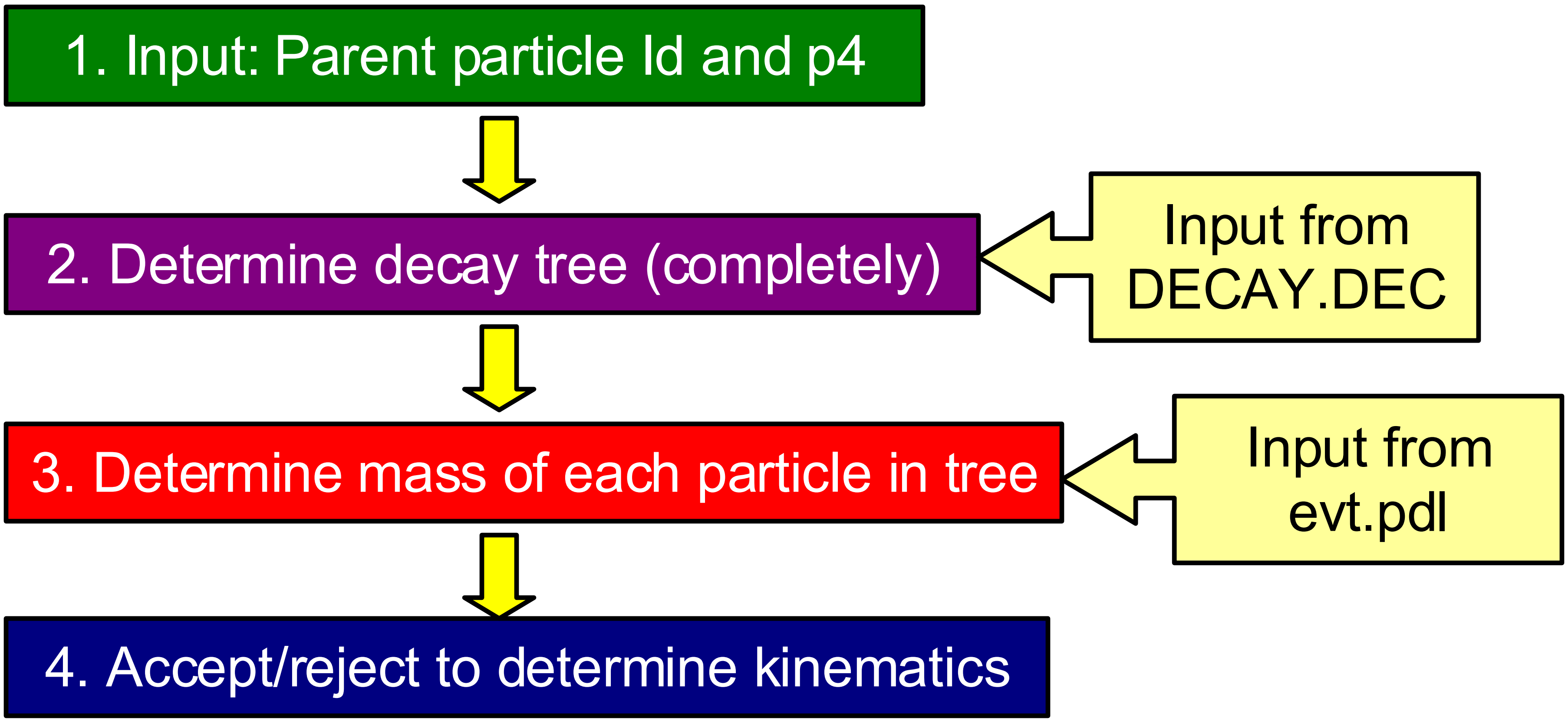
- EvtGen works with amplitudes. The amplitudes are specified as amplitudes between the initial and final state in a set of basis vector provided by EvtGen.
- EvtGen uses the following representation for the lower spin states:

Class name	Rep.	J	States	Example
EvtScalarParticle	1	0	1	π, B^0
EvtDiracParticle	u_α	1/2	2	e, τ
EvtNeutrinoParticle	u_α	1/2	1	ν_e
EvtVectorParticle	ϵ^μ	1	3	ρ, J/Ψ
EvtPhotonParticle	ϵ^μ	1	2	γ
EvtTensorParticle	$T^{\mu\nu}$	2	5	D_2^*, f_2

lifferent types of particles that supported by EvtGen. The sp
resentation has not yet been implemented.

- Also J=3/2 EvtRaritaSchwinger 4 states
- Higher spin states are represented by a generic helicity state basis

EvtGen decay algorithm



- Configuration specified by input files at run time.
  - Users override generic DECAY.DEC to generate MC as needed.

The decay table (DECAY.DEC)

- We continue to increase the ability to control EvtGen via $C3_DATA/DECAY.DEC
  – Decays and branching fractions
  – Particle masses, widths, lineshapes
  – Try to avoid hardwiring numbers that control decay models, instead specifying them as arguments.
  – Control of usage of PHOTOS packages

Additional control avoids the need to change software to produce MC for systematic studies

Defining particle decays

```
Decay D*+
    0.683 D0  pi+    VSS;
    0.306 D+  pi0    VSS;
    0.011 D+  gamma  VSP_PWAVE;
Enddecay
```

Particle to decay

Branching fraction

Decay model

Daughters

";" to end each mode definition

End of decay stanza

Defines three decay modes of the D*+

Branching fractions will be rescaled to sum to 1.0

Particle "aliases"

```
Alias MyD*+ D*+

Decay B0
    1.0 MyD*+ pi-        SVS;
Enddecay

Decay MyD*+
    1.0 D0      pi+      VSS;
Enddecay
```

- In this case, all B0s will decay to $D^{*+}\pi^{-}$, with D^{*+}->$D^{0}\pi^{+}$. However, other D^{*+} in the event will decay as defined in DECAY.DEC.

Model arguments

HQET parameters

Some models takes arguments:

```
Decay B0
1.00 D*-    e+   nu_e          PHOTOS HQET 0.92 1.18 0.72;
Enddecay
```

These arguments can be accessed in the model using the methods:

getNArg() returns the number of arguments
getArg(i) returns the ith argument

evt.pdl format

Particle properties are defined in $C3_DATA/evt.pdl:

```
Add p Lepton mu-   13   0.1056584 0      0    -3 1 658654. 13
Add p Lepton mu+ -13  0.1056584 0      0     3 1 658654. 0
Add p Meson pi+   211  0.139570  0      0     3 0  7804.5 101
Add p Meson pi-   -211  0.138570  0      0    -3 0  7804.5 0
Add p Meson rho+ 213 0.7685       0.151 0.4  3 2        0   121
Add p Meson rho- -213 0.7685       0.151 0.4 -3 2        0   0
…..
```

- 4^{th} column=particle name, 5^{th}=stdhep number, 6^{th}=mass (GeV/c2), 7^{th}=Width (GeV/c2), 8^{th}=Mass cutoff, 9^{th}=3*charge, 10th 2*spin, 11^{th}=ct (mm), 12^{th} Lund-KC number (for Pythia interface)

Available decay models

- General purpose models that decay according to specified helicity or partial wave amplitudes
  - Handle decays to two body final states with arbitrary spins. Amplitudes specified at run time.
- Specific CP violating models
- Semileptonic form-factor models
- Dalitz decays
  - Specific: D, η, π^0, ω
  - General Pseudoscalar -> 3 Pseudoscalar
- $B \to Kll$, $b \to s\gamma$
- Use PHOTOS package for final state radiation.
  - On by default for all decays.

Semileptonic decays

- HQET - Heavy Quark Effective Theory inspired form factor param.
- ISGW, ISGW2 - Quark model based prediction, Isgur, Scora et al.
- MELIKHOV - Quark model based prediction
- SLPOLE - Generic spcification of form factors based on a lattice inspired parametrization.
- VUB - For generic b->ulnu decays, uses JetSet for fragmentation.
- GOITY_ROBERTS - Decays to non resonant D(*)pi lnu.

BABAR uses, HQET, ISGW2, VUB, and GOITY_ROBERTS in its simulation.
ISGW2 should support D, D_s and B_s decays as well as B decays.

Generic amplitudes

- HELAMP, PARTWAVE - generic two-body decays specified by the helicity or partial wave amplitudes.
- SLN - Decay of scalar to lepton and neutrino.
- PHSP - N-body phase space.
- SVS, STS - Scalar decay to vector (or tensor) and scalar.
- VSS, TSS - decay of vector or tensor particle to a pair of scalars.
- VLL, SLL - Decay of vector or scalar to two leptons.
- VSP_PWAVE, vector to scalar and photon, e.g., D*->Dγ

Special matrix elements

- BTOXSGAMMA - b->X_s γ with JetSet fragmentation.
- BTOXSLL - b->$X_s$ll with JetSet fragmentation.
- D_DALITZ - 3-body *D*-decays with substructure.
- ETA_DALITZ - η–>3π with measured dalitz amplitude.
- KSTARNUNU - B->K*nunubar
- LNUGAMMA - B->lnu γ
- OMEGA_DALITZ - Dalitz structure in the ω->3π decay
- PHI_DALITZ - Dalitz structure in the ϕ->3π decay
- PTO3P - scalar to 3 scalars decay where you can specify intermediate resonances
- TAUHADNU - hadronic 1, 2, and 3 pion final states.
- TAULNUNU - leptonic tau decays.
- VSS_BMIX - Upsilon(4S) to BBbar, including mixing.
- VVPIPI - decay of vector to vector and two pions, *e.g.* psi'->psi pi pi.
- VECTORISR - ISR production of vector mesons: e+e- -> Vγ

Writing new Physics Models

- This part of the tutorial deals with writing new models
  - A model is a C++ class that implements the calculation of amplitudes for a given process.
  - This class has to be registered with the frame work in order to be used.
  - The model has a name which is used to indentify the model in the decay table.
- There are currently about 80 decay models implemented in EvtGen.

Example decay: V->SS

To illustrate how a decay model is written we will use the example of the decay of a vector particle to two scalars. The amplitude for this decay is given simply by:

$$A = \varepsilon^{\mu} v_{\mu}$$

Where ε is the polarization vector of the initial vector meson and v is the four-velocity of one of the final state particles.

We will illustrate how we write the class, EvtVSS, to implement the calculation of this amplitude for a model named 'VSS'.

EvtVSS.hh (simplified)

```
#ifndef EVTVSS_HH
#define EVTVSS_HH

#include "EvtGenBase/EvtDecayAmp.hh"

class EvtParticle;

class EvtVSS:public  EvtDecayAmp  {

public:
  EvtVSS() {}
  virtual ~EvtVSS();

  void getName(std::string& name);
  EvtDecayBase* clone();

  void decay(EvtParticle *p);
  void init();
  void initProbMax();

};
#endif
```

EvtVSS.cc

```
#include <stdlib.h>
#include "EvtGenBase/EvtParticle.hh"
#include "EvtGenBase/EvtGenKine.hh"
#include "EvtGenBase/EvtPDL.hh"
#include "EvtGenBase/EvtVector4C.hh"
#include "EvtGenBase/EvtVector4R.hh"
#include "EvtGenBase/EvtReport.hh"
#include "EvtGenModels/EvtVSS.hh"
#include <string>

EvtVSS::~EvtVSS() {}

void EvtVSS::getName(std::string& model_name){
 model_name="VSS";
}

EvtDecayBase* EvtVSS::clone(){
 return new EvtVSS;
}

void EvtVSS::initProbMax() {
  setProbMax(1.0);
}
```

```
void EvtVSS::init(){
 // check that there are 0 arguments
 checkNArg(0);

 // check that there are 2 daughters
 checkNDaug(2);

 // check the parent and daughter spins
 checkSpinParent(EvtSpinType::VECTOR);
 checkSpinDaughter(0,EvtSpinType::SCALAR);
 checkSpinDaughter(1,EvtSpinType::SCALAR);
}


void EvtVSS::decay( EvtParticle *p){

 p->initializePhaseSpace(getNDaug(),getDaugs());

 EvtVector4R pdaug = p->getDaug(0)->getP4();

 double norm=1.0/pdaug.d3mag();
 vertex(0,norm*pdaug*(p->eps(0)));
 vertex(1,norm*pdaug*(p->eps(1)));
 vertex(2,norm*pdaug*(p->eps(2)));

 return;
}
```

Registering the model

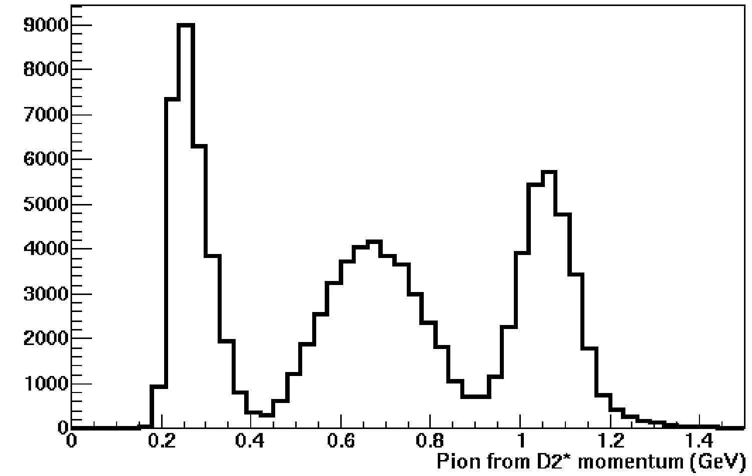
The last step to do before you can use a model is to register it with the EvtGen framework. This is done in the EvtModelReg.cc:

```
modelist.Register(new EvtVSS);
```

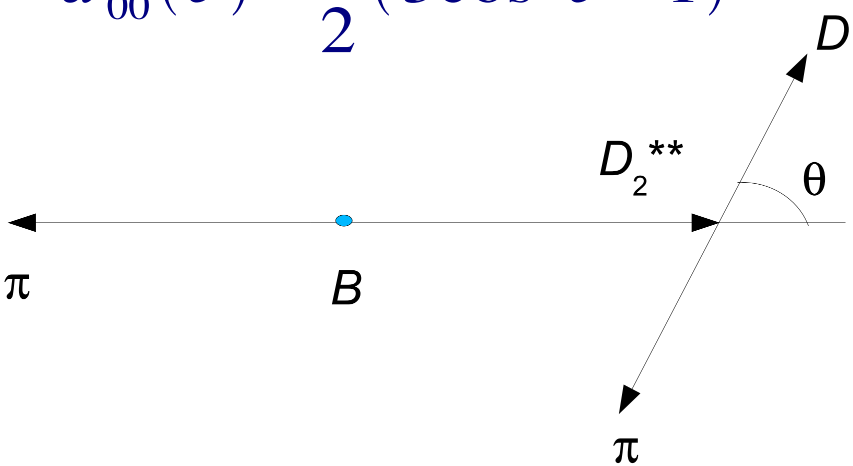
For each instance of a decay in the decay table that uses the VSS model
a new instance of the EvtVSS class is created using the clone method.

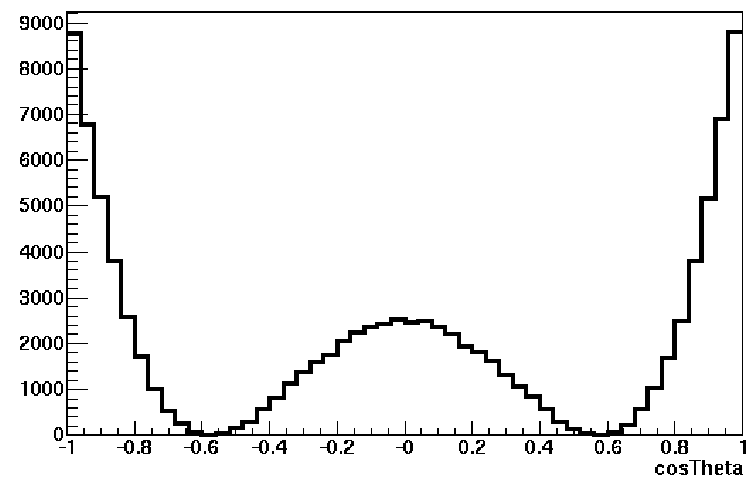
HELAMP and PARTWAVE models

- *B->D$_2$**π*   *D$_2$**->Dπ*
  - Known and nontrivial kinematical distributions.
  - For decays with multiple allowed partial waves, amplitudes are specified as model argument



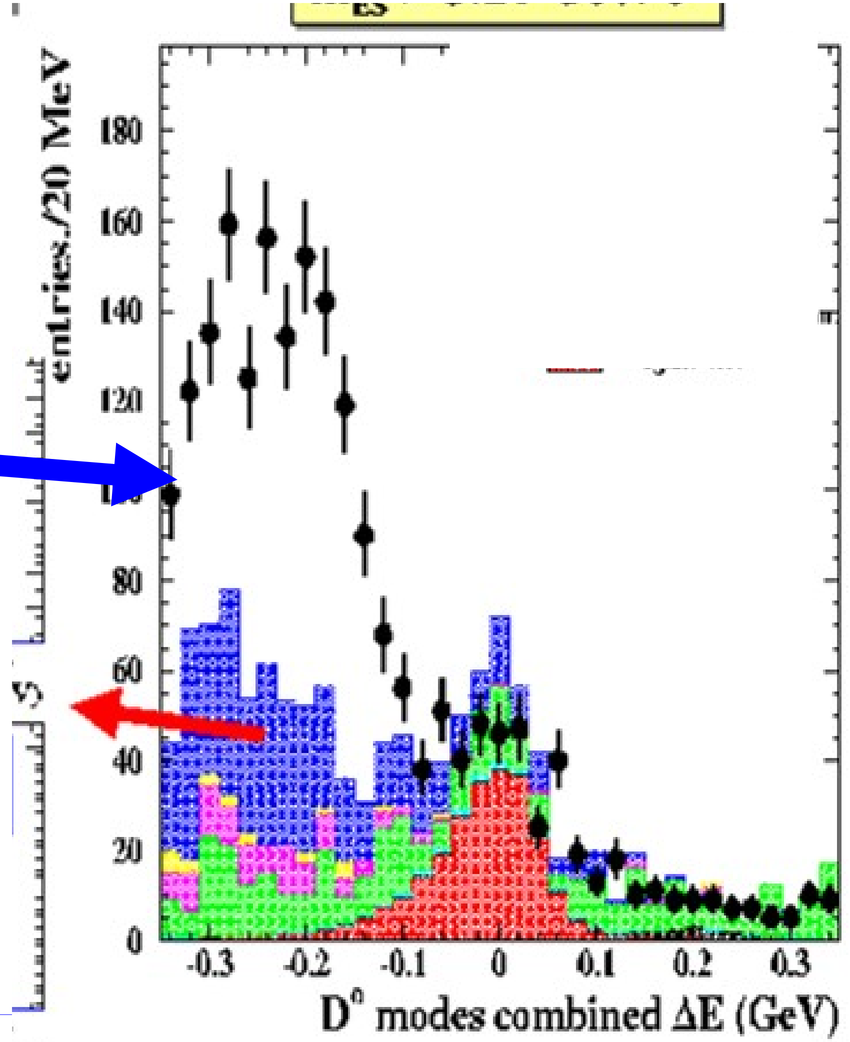
$$A = d^2_{00}(\theta) = \frac{1}{2}(3\cos^2\theta - 1)$$





Given large data sample, detailed effects must be modeled in generic *B* Monte Carlo

Mixed up two decay amplitudes in $B\text{->}D^*\rho$ for generic MC led to large data vs MC differences for some analyses.

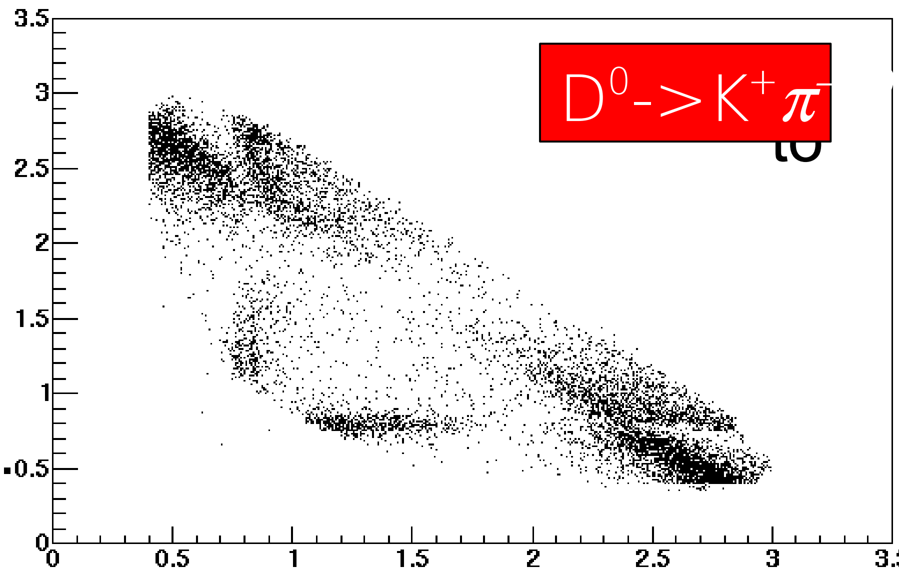


Jetset 7.4 used for inclusive decay generation

- We rely on Jetset to handle *ee->qq* fragmentation and *B* decays not specified in the decay table.
- *B* decays:
  - Approximately 40% of the *B* decay width is not explicitly listed in decay table.
  - Pythia decays are accepted if generated mode is not specified in the decay table.
  - We have performed some tuning to improve the data *vs.* MC agreement
    - BF to charmless non-resonant states too big.
    - *D** production in both *B* and *ee->cc* decays

Lineshapes and Dalitz plots

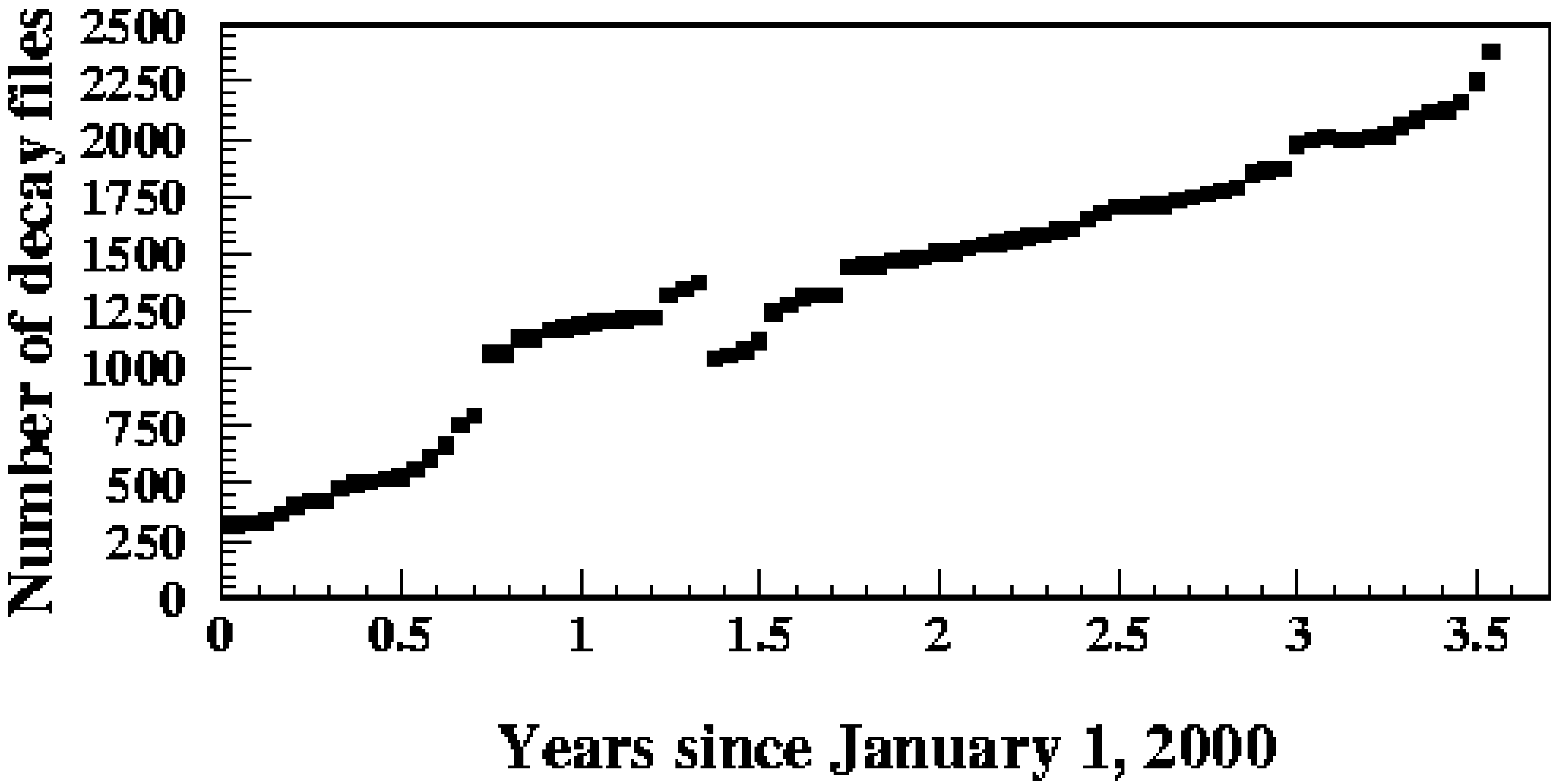
- Try to use relativistic Breit-Wigners for all particles with finite width.
  - Only for decays to two daughters
    - Otherwise non-rel BW.
  - Particles produced by Jetset have non-rel BW
- Include where possible
  - phase space factors, birth and decay form factors.
- Minimize use of mass cutoffs
  - Still needed in many cases to prevent crashes due to pathological configurations.
- Moving towards integrated lineshape and Dalitz plot code.



Monte Carlo production in BABAR

- BABAR generates Monte Carlo to match reconstruction code releases.
- Production generators "frozen" for each cycle
  - DECAY.DEC in particular.
  - Bug fixes ok.
  - Rarely, we include updates for new results. More often, improved in next production cycle.
- Given release cycle timescales, we must support multiple release cycles until analysis are completed on data from old releases..
- Last production cycle has now produced 2.1B events.

Users test and commit generator control files to CVS for centralized MC generation



- Large rate of special MC requests.

Using EvtGen in suez

In releases newer than Dec10_03, using EvtGen is easy:

```
cleog gen EvtGenProd $env(NUMEVT) out $fileout run 200556
    -user_decay $env(UDECAY)  -post {
    proc sel RunEventNumberProc
}
```

Where -user_decay <file> specifies a user file to overwrite the default decay file as given in $C3_DATA/DECAY.DEC

Currently EvtGenProd creates an initial virtual photon (vpho) that is decayed using jetset to a quark anti-quark pair:

```
Decay vpho
1.000             JSCONT 0;
Enddecay
```

Writing a user decay file

```
#
Alias myD0 D0
Alias myanti-D0 anti-D0
#
Decay vpho
0.500    myD0  anti-D0          VSS;
0.500    D0    myanti-D0        VSS;
Enddecay
#
Decay myD0
1.000    eta  pi0            PHSP;
Enddecay
#
Decay myanti-D0
1.000    eta  pi0            PHSP;
Enddecay
#
End
```

LundAreaLaw

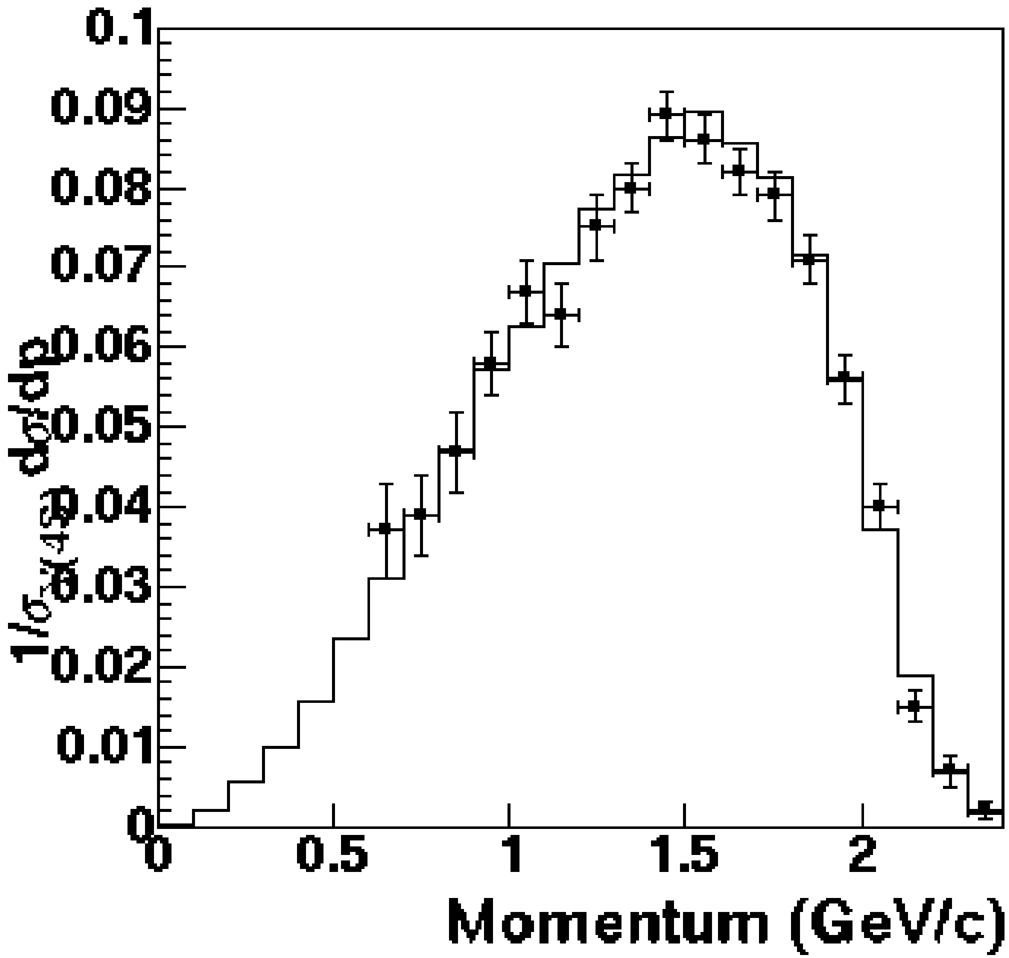
- Jim N. and I added the LundAreaLaw to EvtGen
  - The lund area law is a modified version of JetSet that should produce a more accurate fragmentation at low energy, in particular it should simulate baryon production better.
- To use the lund area law for the fragmentation in e^+e^-:

```
Decay vpho
1.000              LUNDAREALAW  0;
Enddecay
End
```

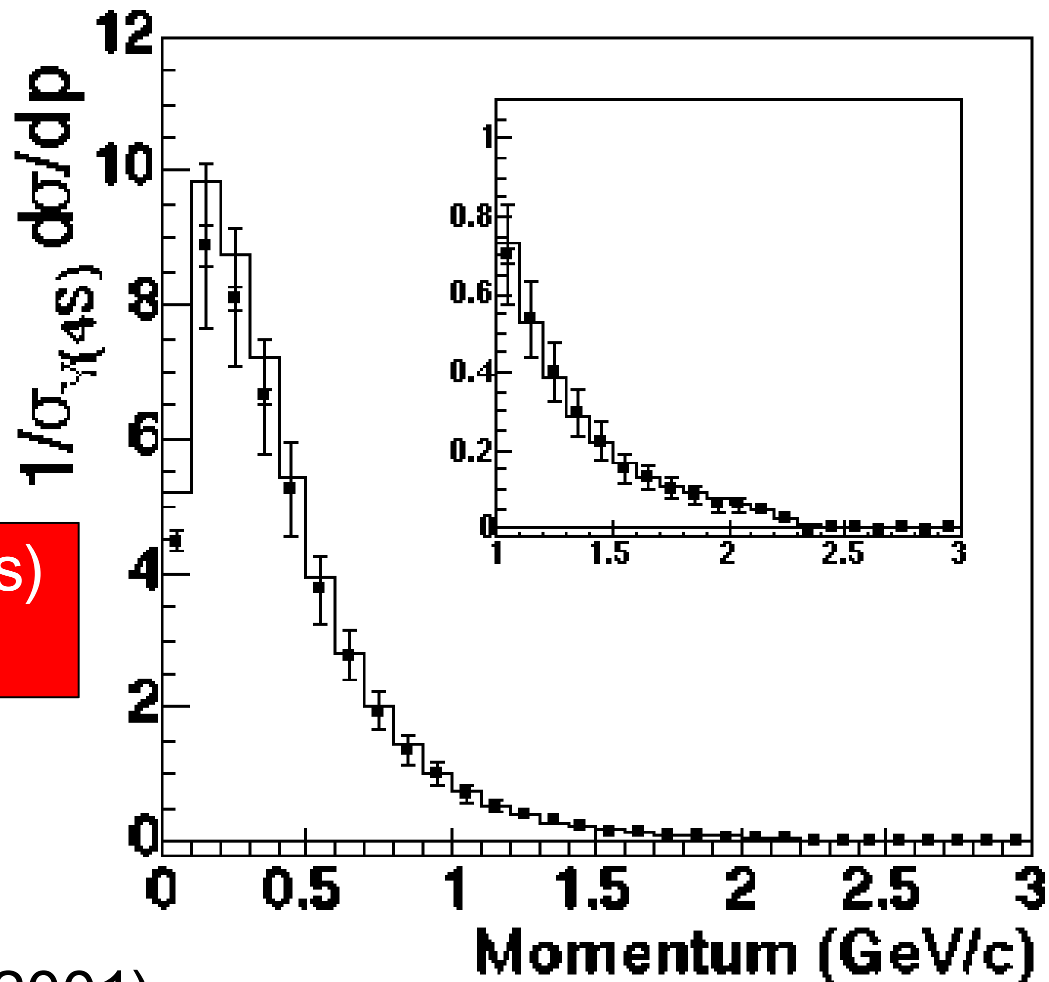
*B->Xlν l*epton energy spectrum

- Lepton energy spectrum tuned using CLEO data.
  - PRL 76 1570 (1996)

Mode	BF (%)
$D^*l\nu$	5.6
$Dl\nu$	2.1
$D_1^{**}(2420)l\nu$	0.56
$D_0^{**}l\nu$	0.2
$D_1^{**'}(2460)l\nu$	0.37
$D_2^{**}(2460)l\nu$	0.37
$D^*\pi l\nu$	0.3
$D\pi l\nu$	0.9



π^0 momentum spectrum



Points = Belle (stat+sys)
Histogram = EvtGen

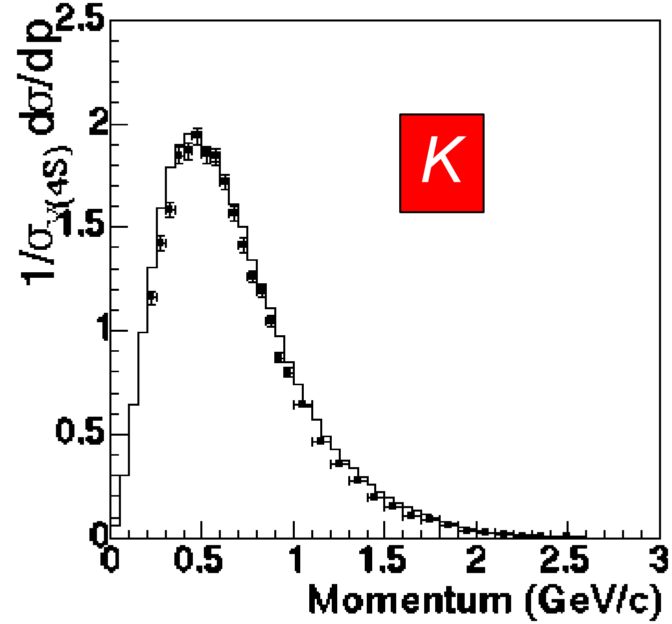
- PRD 64, 072001, (2001)

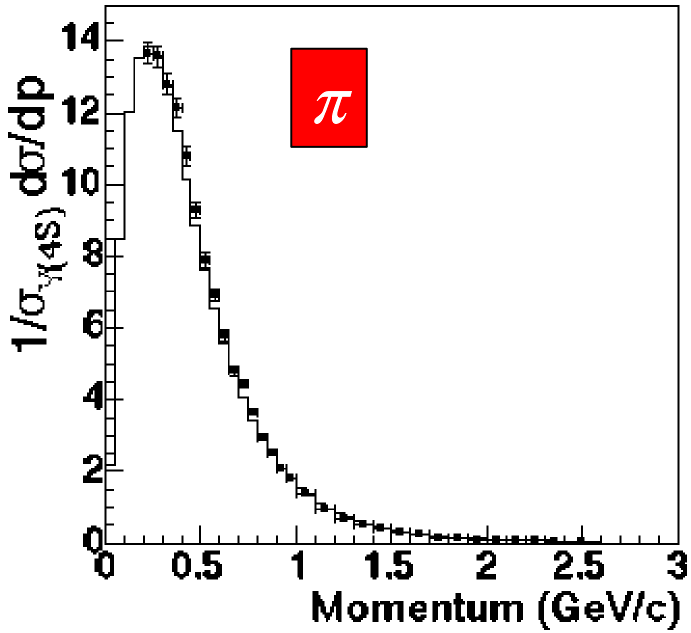
Inclusive resonance production in B decays

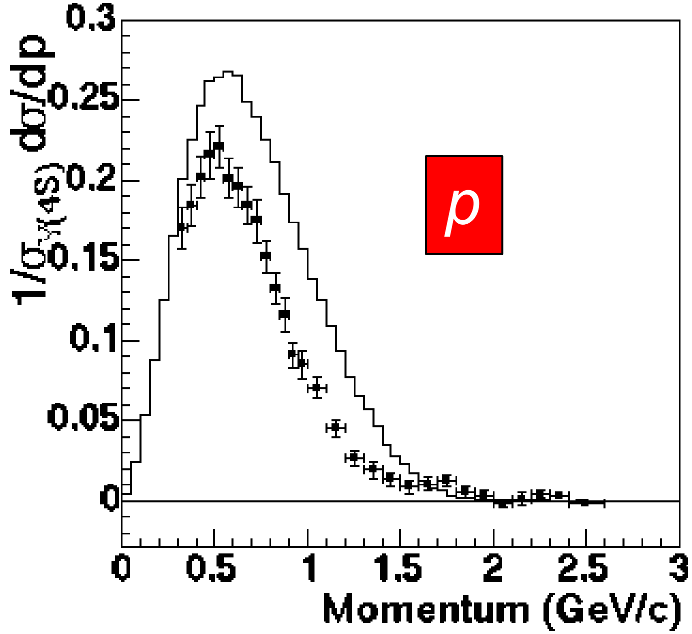
	PDG03	EvtGen
$B\text{->}Xe\nu$	10.7 +/- 0.28	10.6
$B\text{->}D^{+-}X$	24.5 +/- 2.1	32.4
$B\text{->}D^{0}X$	64.0 +/- 2.9	68.2
$B\text{->}D^{*+-}X$	22.5 +/- 1.5	26.2
$B\text{->}D^{*0}X$	26.0 +/- 2.7	25.7
$B\text{->}D^{(*)}D^{(*)}K$	7.1+2.7-1.7	7.7
$B\text{->}J/\psi X$	1.090 +/- 0.035	1.04
$B\text{->}K^{*+-}X$	18 +/- 6	17.5
$B\text{->}\eta X$	17.6 +/- 1.6	22.8
$B\text{->}\Lambda_c X$	6.4 +/- 1.1	3.7
$B\text{->}\Lambda X$	4.0 +/- 0.5	4.6
$B\text{->}\phi X$	3.5 +/- 0.7	4.7

PDG B->D(*) production Bfs not consistent with isospin (and B(B->X)=1) at several sigma level

Preliminary Υ(4S)->π/K/p spectra from BABAR







	BABAR	EvtGen
π	7.73+-0.32	7.98
K	1.54+-0.04	1.61
p	0.155+-0.004	0.224

Conclusion

- EvtGen has been interfaced to the CLEO-c framework (suez).
- EvtGen provides generic tools to solve a number of problems in simulation of particle decays
  - Still some updates/improvements are needed for the CLEO-c era.
- Used by BABAR/Belle/CDF/LHC exp. for *B*-decays
  - Headache to maintain fixes from different places...
- Modular framework
  - Makes it easy to add new physics models.
- Much tuning has been done at the $\Upsilon(4S)$, hope that we will similarly improve the simulation at the 'lower' energies.

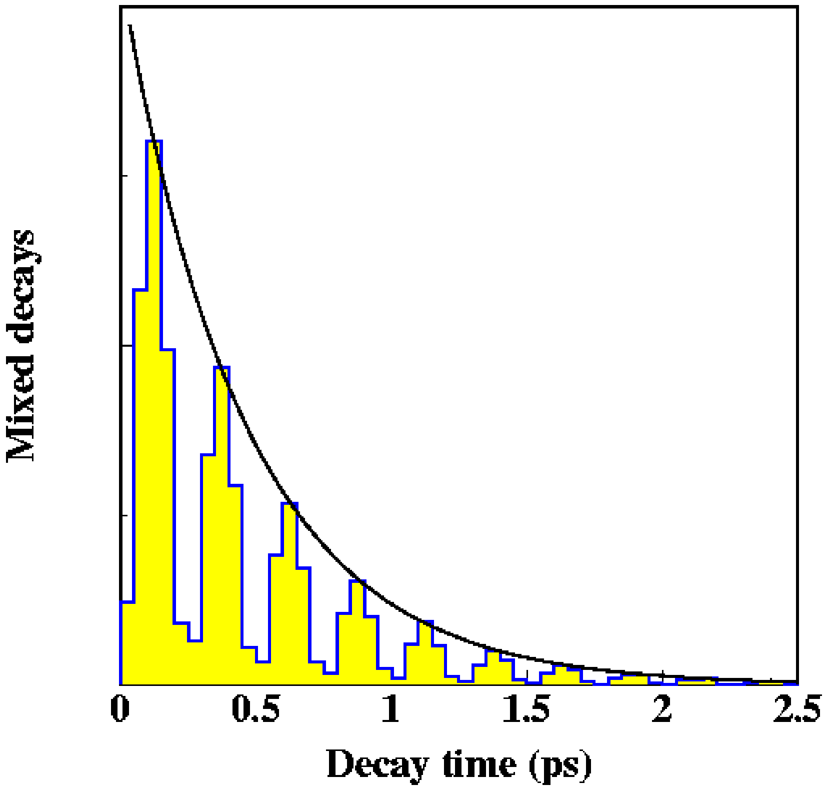
Available decay modes (III)

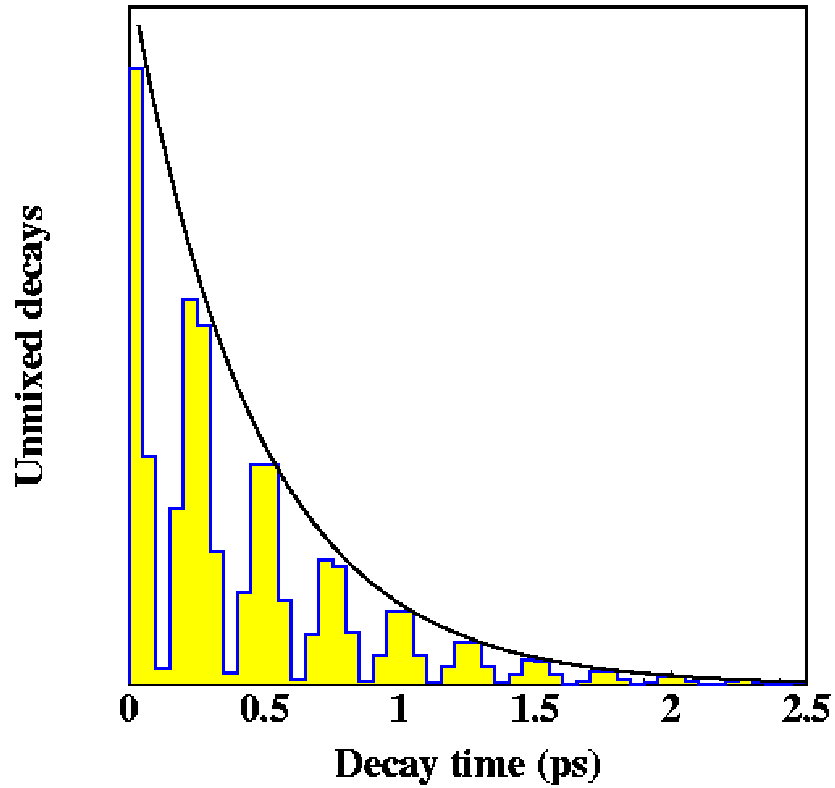
- SSD_CP model simulates CP violation for final states with a pseudoscalar + either a scalar,vector, or tensor.
  - B->$\pi\pi$, B->J/ψKs, B->D*π, etc.
  - Specify in decay table:
    - Δm
    - $\Delta\Gamma/\Gamma$
    - *q/p*
    - *A(B->f), A(Bbar->f,A(B->fbar),A(Bbar->fbar)*
    - *z*
  - Flexible but relatively new model, so we are still gaining experience with all the possible use cases.

B_s physics in EvtGen

Items different wrt $\Upsilon(4S) \rightarrow BB$ decays:

- Large # of common final states
- Incoherent mixing





Conclude about common final states

Basic EvtGen interface (EvtGen.cc)

```
EvtGen myGenerator(
                <DECAY.DEC location>,
                <evt.pdl location>
                <randomNumberEngine>.
                <FSR generator>);

myGenerator.readUDecay(<user decay file>);
EvtParticle *myParentParticle;
….. (Set up parent particle properties)….

myGenerator.generateEvent(myParentParticle,t_init);
```

Optional: PHOTOS is default.