CDF/DOC/BOTTOM/CDFR/5618
Release Version 3.15.0int1
April 20, 2001

User Guide for EvtGen @ CDF

William Bell
Argonne/Fermilab/Glasgow

Joe Boudreau
Luis R. Flores
Pittsburgh University

J. Pablo Fernández
Purdue University

Frank Würthwein
Massachusetts Institute of Technology

Abstract

This document describes how to use EvtGen, a B decay Monte Carlo package at CDF. This is a living document with the present version referring to software release 3.15.0int1 . The proper home of this document is **$CDFSOFT2_DIR/EvtGen/doc/CdfUserGuide.ps** .

We are committed to keep this document current with all frozen releases of the CDF Run2 software.

1 Getting Started

In general, the EvtGen package is spiked with README files in all of its subdirectories. For the beginner, the following directories and files should be particularly useful:

- **EvtGen/testTcl** . This directory contains a number of examples. Check out the README for more details.
- **EvtGen/testExe** is the source directory for EvtGen.exe, the tbin exe that comes with EvtGen. It also contains getSpectra.cc which is an exe used to validate EvtGen.
- **EvtGen/doc/CdfUserGuide.ps** . This is this document.
- **EvtGen/doc/guide.ps** , the master manual from BaBar. This manual describes all the classes, as well as physics models implemented in EvtGen. It get's updated whenever EvtGen @ CDF is updated from the BaBar cvs.

- **EvtGen/testRootMacro** contains root macros and kumacs used for the physics validation of the generator. This is discussed in more detail in section 3.
- **EvtGen/test** contains a large number of decay files. All of these come straight from BaBar, and experience tells us that not all of them work properly. If you find one that doesn't work, send email to **fkw@mit.edu**.
- Input files for EvtGen:
  - **EvtGen/pdt.table** is pointed to by the environment variable **EVTGEN_PDT_TABLE**. It contains all the particle definitions EvtGen knows about. If you want to introduce a new particle, e.g. an exotic bsquark, then you would have to add it here.
  - **EvtGen/DECAY.DEC** is pointed to by the environment variable **EVTGEN_DECAY_FILE**. It contains all the default particle decay definitions EvtGen knows about. However, to override the default for just one particle you should write your own decay file rather than modifying DECAY.DEC. Examples of how this works are provided by **EvtGen/testTcl/ForAnikeev.tcl**, **EvtGen/testTcl/ForXin.tcl**, as well as the decay files specified in these tcl files.

If you find the documentation provided to you with this package insufficient, incorrect, or plain confusing, please complain via email to **fkw@mit.edu** , and we'll try to do better in the next release.

2 Some Details

2.1 EvtGen Interface to AC++

The EvtGen interface, connecting AC++ to EvtGen classes, was written as two separate classes: `EvtGenModule` and `EvtGenInterface`. The class diagram for these pieces of code is illustrated in Figure 1

The EvtGen interface can operate in two modes. It can either be used in "standalone" mode, or to decay B mesons generated and added to `HEPEVT` common block by some other generator, e.g. Pythia. (The common block is used directly to avoid producing excess HEPG banks. This would cause problems in events where there are many simulated interactions.) In either of the two possible operating modes the module needs to be linked with Pythia or another module that initializes the HEPEVT common block.

An example use of the "standalone" mode is simulation of $\Upsilon(4s)$ decays like at BaBar.

2.1.1 Standalone Example: Producing $\Upsilon(4s)$ decays

Talktos used in the $\Upsilon(4s)$ example, together with their default values can be found in table 1.

The only complication is that some of the particle codes used in the Particle Data Table (pdt.table) file are not the same as those given in the PDG Monte Carlo listing. In particular Υ(4s) is different. It is advisable to check the particle id code given in **EVTGEN_PDT_TABLE** (pdt.table).

An example tcl file for Υ(4s) decay is given in **EvtGen/testTcl/U4S.tcl**.

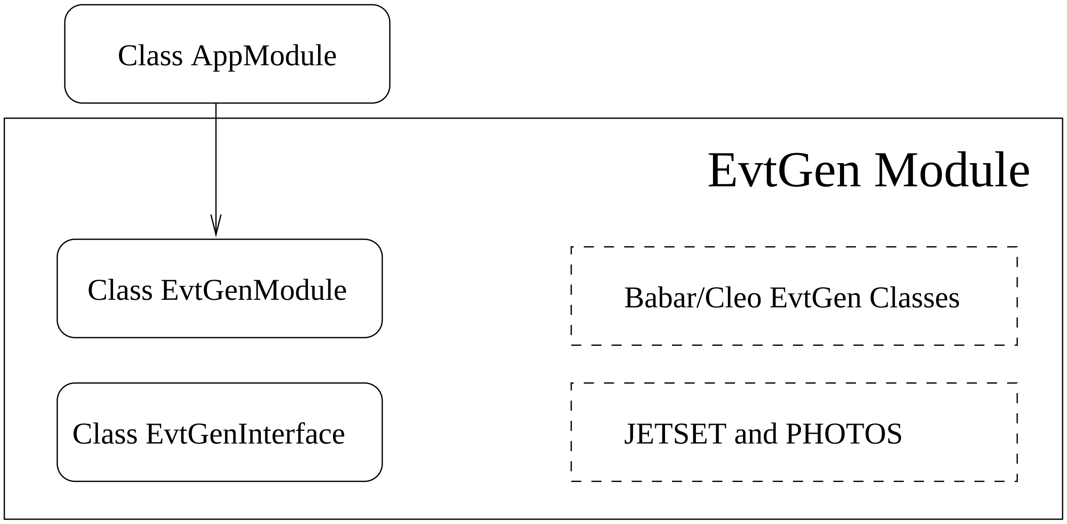

Figure 1: Classes contained in the EvtGen Module

Name	Default Value	$\Upsilon(4s)$ Example
UseRootParicle	false	true
RootPariclePx	0.0	0.0
RootPariclePy	0.0	0.0
RootPariclePz	0.0	0.0
RootParicleHepID	0	70553

Table 1: Talktos used before generation of $\Upsilon(4s)$ decays

2.1.2 Decaying Bottom Meson and Charmed particles

At present EvtGen module only handles decays of bottom mesons, and their daughters. In the short term future prompt charmed particles and bottom baryons will be added.

Name	Default Value	B Meson Example
UseRootParicle	false	false
UseUserDecayFile	false	true
UserDecayFile	dummy	SVVCPLH.DEC
AdjustCPAsymm	false	true

Table 2: Talktos used before decay of HEPEVT particles

PYTHIA, ISAJET and any other generator module that fills HEPEVT with particles can be used as a starting point. The generator module must be configured such that the HEPG output is switched off. An example of how to this with Pythia is in **EvtGen/testTcl/PythiaGenericB.tcl**.

EvtGenInterface maintains a list of particle types EvtGen is supposed to decay. This is at present hard coded in *EvtGenInterface::ListDecayParticles*. Those particle types are then searched for in HEPEVT. If such a particle is found, all its decay daughters are deleted. These particles are then decayed one at a time by EvtGen. The resultant children are copied from the EvtParticle tree into the CdfHepEvt object private data members, (which are really HEPEVT common block entries). The loop continues until all of the selected particles have been decayed and added to the CdfHepEvt event record. Finally the CdfHepEvt method `makeHEPG_Bank(anEvent)` is called to create a HEPG bank.

In principle the duplicate decay of B mesons first in Pythia, and then again via EvtGen could be avoided by declaring the relevant meson types stable in Pythia. If an event is fed to the EvtGen module where some of the particles to be decayed are stable the module will simply skip the daughter deletion step.

2.1.3 Random Numbers and CP Violation

`EvtGenModule` was written using the HepJamesRandom engine from CLHEP. This generator is also used by the copies of PHOTOS and JETSET that are bundled together with EvtGen. In evtgen-event_.cc two functions of interest are declared: `begran_` and `rlu_`. The former is used by PHOTOS and the latter by JETSET. Within PHOTOS PHORAN calls were replaced for BEGRAN. In the JETSET source code the RLU function was commented out.

2.1.4 Error Conditions

If the EvtGenModule is run and there is already a HEPG bank in the event a
ERRLOG(ELsevere2,"[EVTGEN_BAD_HEPG]")
will be issued and a HEPG bank will not be written out. To generate minimum bias and EvtGen decays in the same event the EvtGen HEPG must therefore be created first.

If the environmental variables **EVTGEN_DECAY_FILE** and **EVTGEN_PDT_TABLE** are not set at the time of execution the AC++ job will terminate with
ERRLOG(ELfatal,"[EVTGEN_NO_DCYTBL]").

2.2 Maintenance of EvtGen @ CDF

We plan on bringing EvtGen from BaBar roughly once every three months. A standarized procedure has been developed to do that and merge the BaBar code into the CDF cvs. This is described at:
http://mit.fnal.gov/~fkw/EvtGen/HowTo.html
As last step of this procedure, we test the new version comming from BaBar with respect to nature.

We will explain in detail, the way this is done so that anybody can reproduce the results if they wish.

- First one should create a new release, add the EvtGen package to it and do a gmake.

- As a second step comes the file generation. We generate $\Upsilon(4s)$ events using EvtGen.
To make our lives easier, the script example located in

$CDFSOFT2_DIR/EvtGen/testTcl/runEvtGen

does all the work for us. You need to specify some parameters if you want to use this script. The parameters that you need as input will be shown in your screen by just running the script with no arguments:

Usage: runEvtGen FileName SampleID NumberOfSeeds EventsPerFile

- *FileName is used for both .tcl and .dec files*
- *SampleID is used as the initial seed (use 2000, 3000, ...)*
- *NumberOfSeeds = number of files [if omitted, 5 is assumed]*
- *EventsPerFile. For AC++ -nev line [if omitted, 500 is assumed]*

Generation will only occur if EventsPerFile is either 200 (for tests) or a Multiple of 1000.

Another important file is the tcl file. In the same directory

$CDFSOFT2_DIR/EvtGen/testTcl

you will find a tcl (*U4S.tcl*) that you can take as an example. Note that you will have to make a copy of those files to your local area because this scripts will generate some files that you will need to use later.

● Once you have generated your files, the next step is to pass them through a program that selects those particles comming from B's that we are interested in.

A script that runs over such a program can be found in

$CDFSOFT2_DIR/EvtGen/testTcl/getAllSpectra.csh

Note that you will have to make a copy of those files to your local area because this scripts will generate some files that you will need to use later and also will use as input files the ones you have generated in the previous step. You should also make a link of the executable getSpectra (that is called by the script) to this area. The executable getSpectra is created when you do a gmake EvtGen.tbin .

At this point you will have a set of rz files that you should merge into one by just using the paw command:

PAW > hmerge myoutput.rz myinput01.rz myinput02.rz ...

● Once you are done with all that and you have strengh enough to keep going, you will have to run the kumac

$CDFSOFT2_DIR/EvtGen/testRootMacro/evtgen.kumac.

In the second line of this kumac you will have to specify were your myoutput.rz file is located.

● The final output will be a set of postcript files like the ones we will present later in this note.

3 Physics Benchmarking

Arguably the most important part of a Monte Carlo generator is to get the physics right. The best way of guaranteeing that the physics in the generator is "sufficiently" correct is to heavily rely on "experts", i.e. the people who do a given analysis ought to know best what nature has to say about their physics.

"Validation" of the generator in the Run2 framework is thus heavily decentralized. Anybody with a favorite physics topic is encouraged to participate by contributing:

- a member function to EvtGenTestModule that checks out their favorite physics.[1]
- a tcl file to run EvtGen.exe , the tbin exe that's part of the EvtGen package.
- A root file and some root scripts that analyze it to check that everything is working correctly.
- A tex file etc. to be added in this document to demonstrate that things were working at some point.

The following subsections describe the successful validations we have so far gone through. If you have something to add, please email **fkw@mit.edu** so that we can include it here, and add your name to the author list of this document.

3.1 CP violation @ Hadron Colliders

EvtGen correctly generates the time dependence of decay for a wide variety of physics models. However, it can not change the flavor of the B meson produced by Pythia without screwing up the whole fragmentation process. Time integrated decay rates are thus not dealt with correctly whenever CP violating processes are involved.

For the time being, we kludged this for $B \to J/\psi K_s^0$ by throwing an additional random number in the EvtGenInterface. This is presently not working for CP violating decays other than $B \to J/\psi K_s^0$. If you have trouble with this, or want to generate CP violation in some other decay, please send email to **fkw@mit.edu** until we have fixed this in a way to make it more general, and more user friendly.

To verify that it works for $B \to J/\psi K_s^0$ an additional program has been written: [2]
$CDFSOFT2_DIR/EvtGen/testExe/getSpectra.cc

Figure 2 shows the result of running this program on a sample of 77K events. The tcl (mm.tcl), the executable (EvtGen.exe), decay files (mm.dec) and the kumac (cpplot.kumac) used for this purpose can be found in the directory **EvtGen/testTcl**.

3.2 Comparing EvtGen to Nature and QQ

Over the years CLEO and ARGUS have measured the inclusive B decay spectra for a variety of mesons. These spectra provide important information about the bulk properties of B decay. A decent B decay generator ought to have its decay file tuned such that it reproduces these spectra.

Figures 3 to 7 compare these spectra as given by Nature, EvtGen, and QQ (Run1 version).

- The CLEO data was taken from [1] to [10]. The samples' size was typically of $\approx 2 \times 10^6$ Υ(4s).

[1] At present this module doesn't exist yet. Instead standalone programs are still used.
[2] This program is intended to be a module within EvtGen in the future.

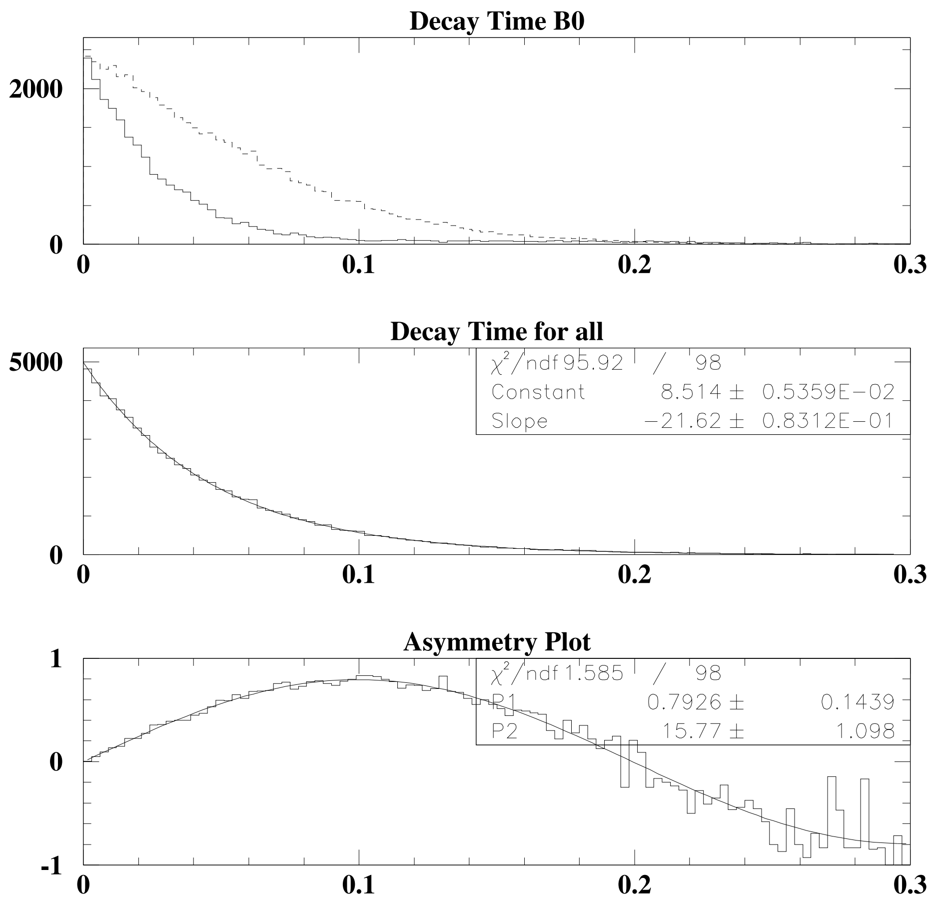

Figure 2: CP asymmetry for B's using EvtGen.

- The EvtGen-decayed $\Upsilon(4s)$ event summed up to 7.86×10^6 and were generated within the AC++ framework with no kinematic restrictions.

- The number of QQ-decayed $\Upsilon(4s)$ events generated is 8×10^6 .

A comparison of the absolute branching fractions for the different decays and multiplicity between EvtGen, QQ and CLEO-data is also shown(see table 3).

In table 3:

(0) The 1^{st} error is always statistical, the 2^{nd} is systematical

(i) The 3^{rd} error is due to uncertainty in $D_s^+ \to \phi\pi^+$ branching fraction

(ii) The 3^{rd} error is determined from the error on the product of branching fractions $B(D^0 \to K^-\pi^+) * B(D^{*0} \to D^0\pi^0)$

(iii) The 3^{rd} error is propagated from the errors on the $D^0 \to K^-\pi^+, D^+ \to K^-\pi^+\pi^+$ and

Branching ratio (%)	measured on CLEO	EvtGen	QQ
$B \to J/\psi$ X	1.12 $\pm$ 0.04 $\pm$ 0.06	1.01	1.16
$B \to \psi'$ X	0.34 $\pm$ 0.04 $\pm$ 0.03	0.29	0.35
$B \to \chi_{c1}$ X	0.40 $\pm$ 0.06 $\pm$ 0.04	0.35	0.40
$B \to D_s^+$ X	11.12$\pm$0.39$\pm$0.88$\pm$1.38(i)	11.55	10.14
$B \to D^{*0}$ X	24.7 $\pm$ 1.2$\pm$1.8$\pm$1.8(ii)	31.6	27.0
$B \to D^{*+}$ X	27.12$\pm$1.1$\pm$1.4$\pm$0.9(iii)	32.27	27.20
$B \to D^0$ X	63.6 $\pm$ 1.4$\pm$1.9$\pm$1.8(iv)	70.9	64.0
$B \to D^+$ X	23.5$\pm$0.9$\pm$0.9$\pm$2.4(v)	27.5	27.5
$B \to \Sigma_c^{++}$ X	0.67 $\pm$ 0.24 $\pm$0.21$\pm$0.15(vi)	0.40	0
$B \to \Sigma_c^0$ X	0.71 $\pm$ 0.26 $\pm$ 0.22 $\pm$ 0.16	0.63	0
$B \to e\nu$ X	10.49 $\pm$ 0.17 $\pm$ 0.43	10.49	10.91
$b \to c \to ye\nu$	7.8 $\pm$ 0.2 $\pm$ 1.2	7.6	8.5
$B \to \eta$ X	17.6 $\pm$ 1.1 $\pm$ 1.2	21.1	21.3
$B \to \phi$ X	3.65 $\pm$ 0.082 $\pm$ 0.31	4.13	3.41
$B \to K_s^0$ X	31.84	34.29	26.78
mean charge multiplicity	10.71 $\pm$ 0.02 $\pm$ 0.21	10.91	10.27

Table 3: *Comparison of different absolute branching ratios and multiplicity between EvtGen, QQ and CLEO-data.*

$D^{*+} \to D^0\pi^+$ decay branching fractions. These results combine $D^{*+} \to D^+\pi^0$ and $D^{*+} \to D^0\pi^+$ decay channels.

(iv) The 3^{rd} error is due to the error on the absolute branching fraction $B(D^0 \to K^-\pi^+)$

(v) The 3rd error is due to the statistical error in the branching fraction $B(D^+ \to pK^-\pi^+\pi^-)$.

(vi) The 3^{rd} error is due to the uncertainty in the branching fraction $B(\Lambda_c^+ \to pK^-\pi^+)$. Same in Σ_c^0.

This comparisons will be incorporated into EvtGenTestModule, to constitute a standard test. [3] The present set of plots were made with the standalone exe **EvtGen/testExe/getSpectra.cc**.

In all of the cases the normalization of the Monte Carlo is done by dividing over twice the total number of events times 100 (to present the branching ratios on %). Data errors are statistical only.

From the plots we can see EvtGen reproduces the spectra at least as well as QQ does. The places where the two decay programs differ the most are K_s^0 (and thus presumably also in K^+) in both spectrum as total branching fraction, and D_s. EvtGen does a better job in both. The total multiplicity distribution is also more accurately modeled by EvtGen while the spectrum of J/ψ from B's is surprisingly done better in QQ than in EvtGen. This is supposedly improved in the next version of EvtGen. So stay tuned!

J/ψ distribution (top left of figure 3) seems the most interesting one. EvtGen and QQ show

[3]The plots shown have been obtained with $CDFSOFT2_DIR/EvtGen/testRootMacro/evtgen.kumac

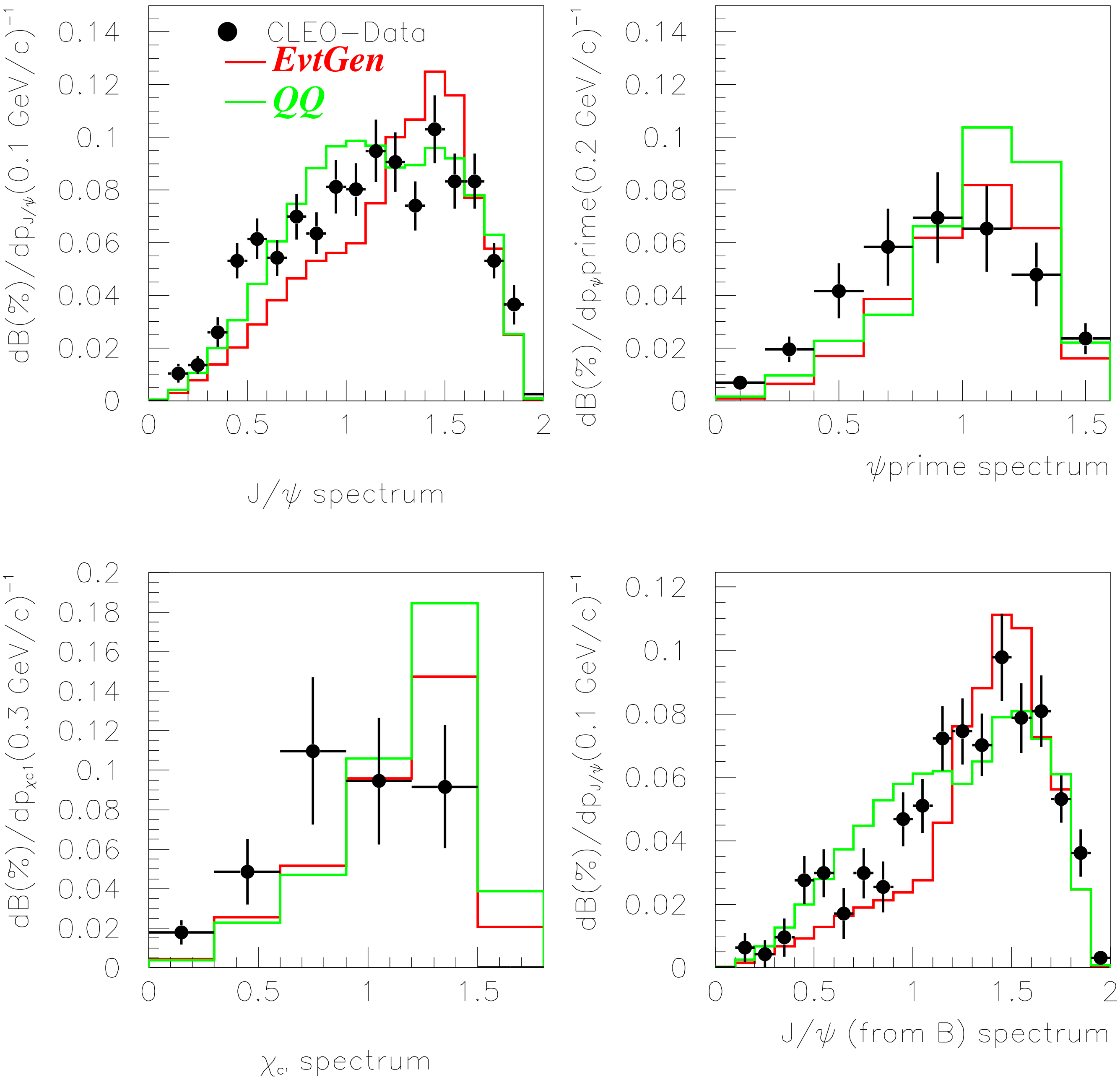

Figure 3: Momentum spectrum for inclusive J/ψ (top left), ψ' (top right), χ_c (bottom left) and J/ψ (bottom right) production from B decays overlaid with the expected momentum spectrum from EvtGen and QQ. Values are shown in %. In the bottom right plot the expected contributions to J/ψ comming from $B \to \psi' X, \psi' \to J/\psi \pi^+ \pi^-$ and $B \to \chi_c X, \chi_c \to J/\psi \gamma$ have been subtracted. See text for source details.

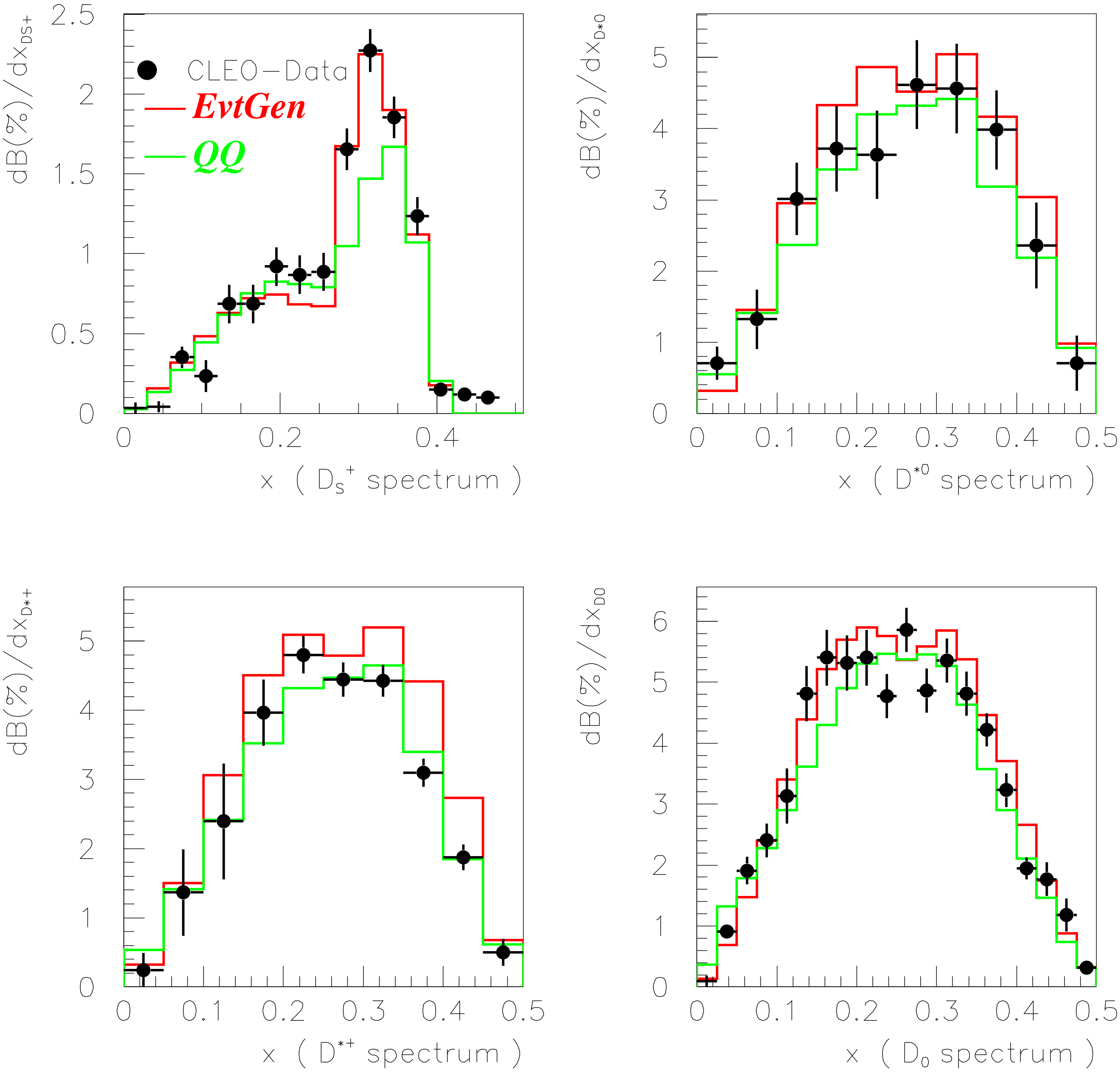

Figure 4: The spectrum $B \to D_s^+$ X (top left), $B \to D^{*0}$ X (top right), $B \to D^{*+}$ X (bottom left), $B \to D^0$ X (bottom right) in data compared to the EvtGen and QQ Monte Carlo. Values are shown in %. x is defined as the maximum momenta available, $x = p_D/p_{max}$, $p^2_{max} = E^2_{max} - m^2_D$, where D is the particular D meson.

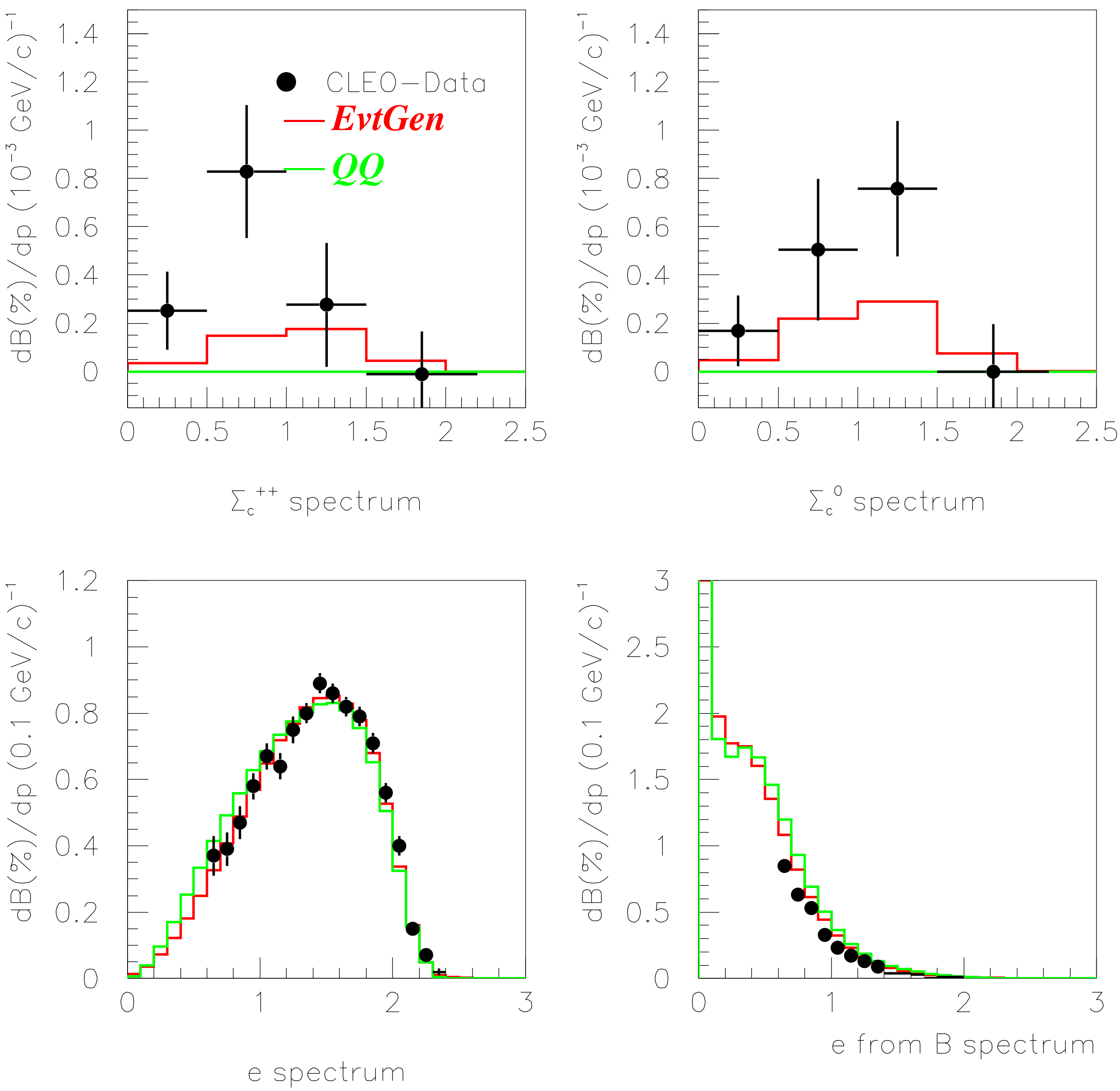

Figure 5: Momentum spectra of Σ_c^{++} (top left), Σ_c^0 (top right), electron (bottom left) and soft electron (bottom right) produced in B meson decays. EvtGen and QQ spectra is overlaid. Values are shown in %. See text for source details.

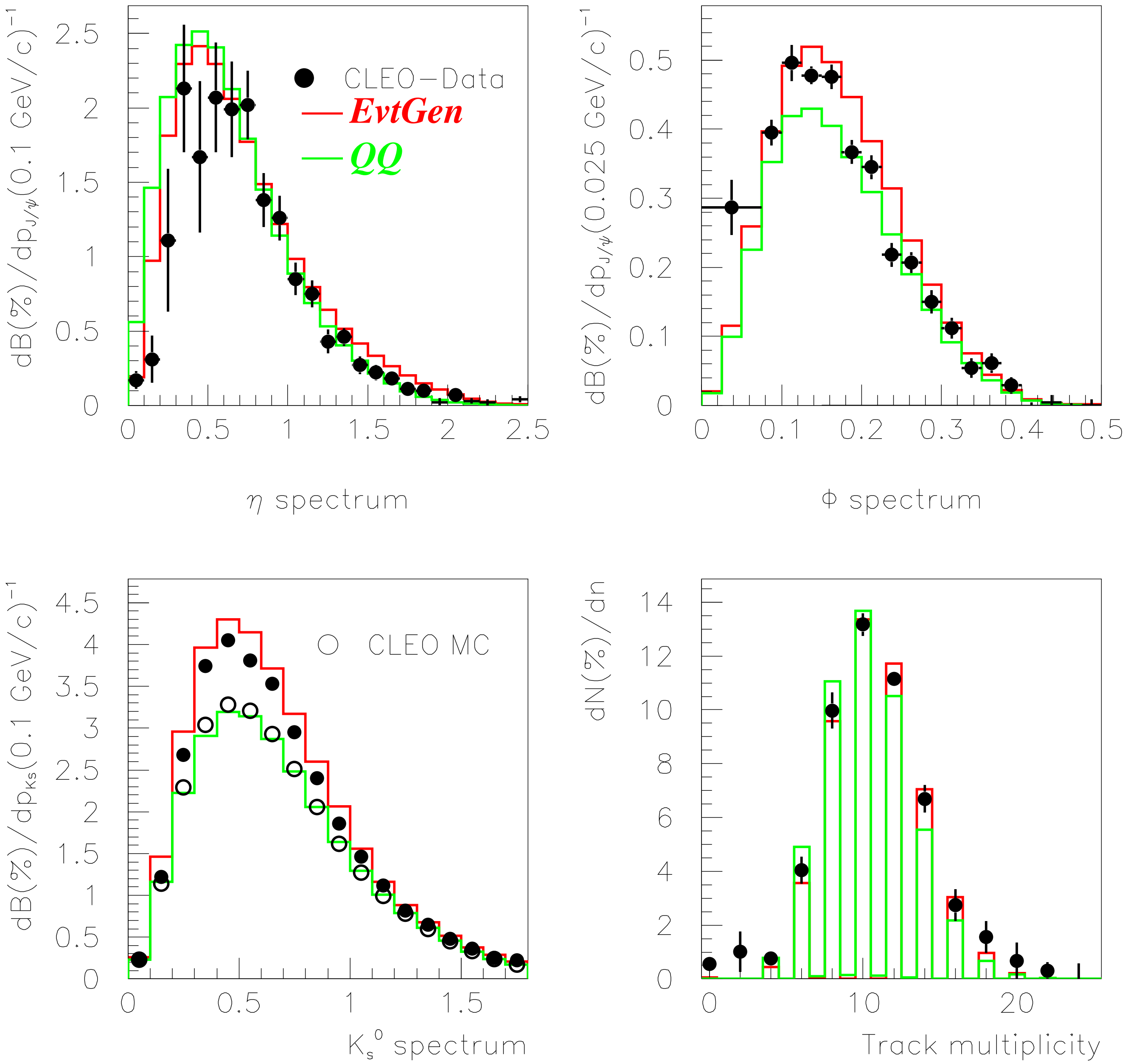

Figure 6: The $B(B \to \eta X)$ (top left), $B(B \to \phi X)$ (top right),$B(B \to K_s^0 X)$ (bottom left) spectra and the particle multiplicity compared with EvtGen and QQ. The CLEO Monte Carlo is also shown in the case of the K_s^0 spectrum. It should agree with QQ and it does.

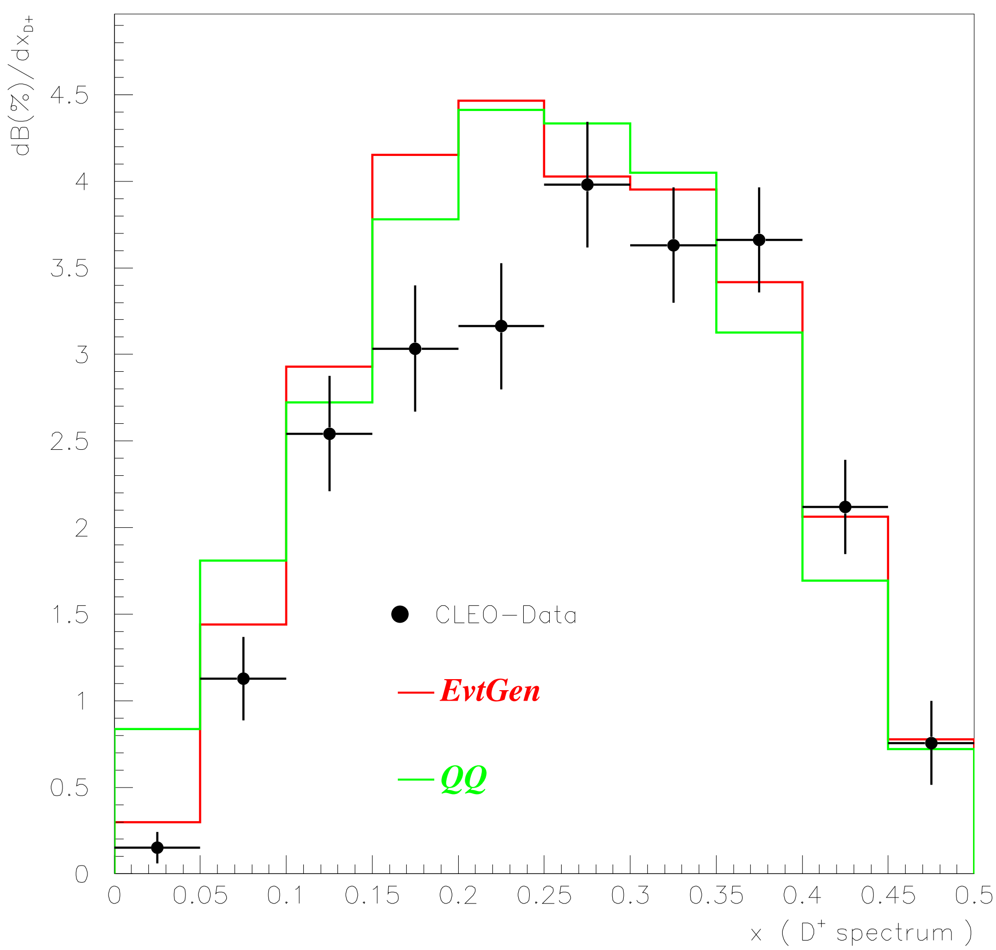

Figure 7: The $B(B \to D^+\mathrm{X})$ spectra compared with EvtGen and QQ.

a different behaviour. QQ spectra seems more symmetric than EvtGen. CLEO data lies between them.

This set of results is intended to be an automated procedure to validate EvtGen whenever it is brought over from BaBar.

4 Existing Limitations

The following physics topics are known to be either not implemented or very suspicious at this time.

- The implementation of time integrated rates for CP violating decays is limited to $B \to J/\psi K_s^0$ and closely related decays. *A fix is being worked on.*
- CP violation including $\Delta\Gamma$ in decays like $B_s \to K^+K^-$. *This requires writing a new decay model for EvtGen.*
- B_c decays. *Somebody needs to make up a decay table for B_c, and afterwards we need to add B_c to the list of mesons passed from the EvtGen interface to EvtGen to be decayed.*
- $b-$baryon decays as well as polarized production. *Same as for B_c.*
- The EvtGen interface only allows for decay of B_s, B_u, B_d. In particular, prompt charm is not selectable at this time. If that is produced then it is decayed by the generator that produced it, and NOT by EvtGen.*Adding prompt charm requires some amount of re-writing of the interface, unfortunately. It's on our to-do list.*

References

[1] CLEO Collaboration, M. Procario et al., Phys. Rev. Lett. 73 (1994) 1472;

[2] CLEO Collaboration, G. Brandenburg et al., Phys. Rev. D 61,072002.

[3] CLEO Collaboration, L. Gibbons et al., Phys. Rev. D 56 (1997) 3783.

[4] CLEO Collaboration, J.P. Alexander et al., CLEO-CONF-95-3 (1995) EPS0155.

[5] CLEO Collaboration, L. Gibbons et al., Phys. Rev. Lett. 76 (1996) 1570.

[6] CLEO Collaboration, D. Gibaut et al., Phys. Rev. D 53 (1996) 4734.

[7] CLEO Collaboration, R. Balest et al., Phys. Rev. D 52 (1995) 2661.

[8] CLEO Collaboration, B. Barish et al., Phys. Rev. Lett. 76 (1996) 1570.

[9] Private communication, F. Wuerthwein.

[10] CLEO Collaboration, Y. Kubota et al., Phys. Rev. D 53 (1996), 6033.