UED KK-quark decays in pandora

Michael Davenport, SLAC

with Michael Peskin, Josh Ruderman

MC4BSM
Princeton 2007

What is pandora?

Self contained parton level event generator, written in C++

Goals:

- Full treatment of polarization and spin correlations
  (origin in ILC studies)
- Efficiently handle large decay chains
- Provide a toolbox for implementing new physics models

Where is pandora?

Not at www.slac.stanford.edu/~mpeskin/pandora3.html but will be soon

Where is pandora?

Not at www.slac.stanford.edu/~mpeskin/pandora3.html

We decided to update the engine about 2 weeks ago, and haven’t quite hooked everything back together yet.

Treatment of Spin Correlations

pandora is based on helicity amplitudes for production and decay.

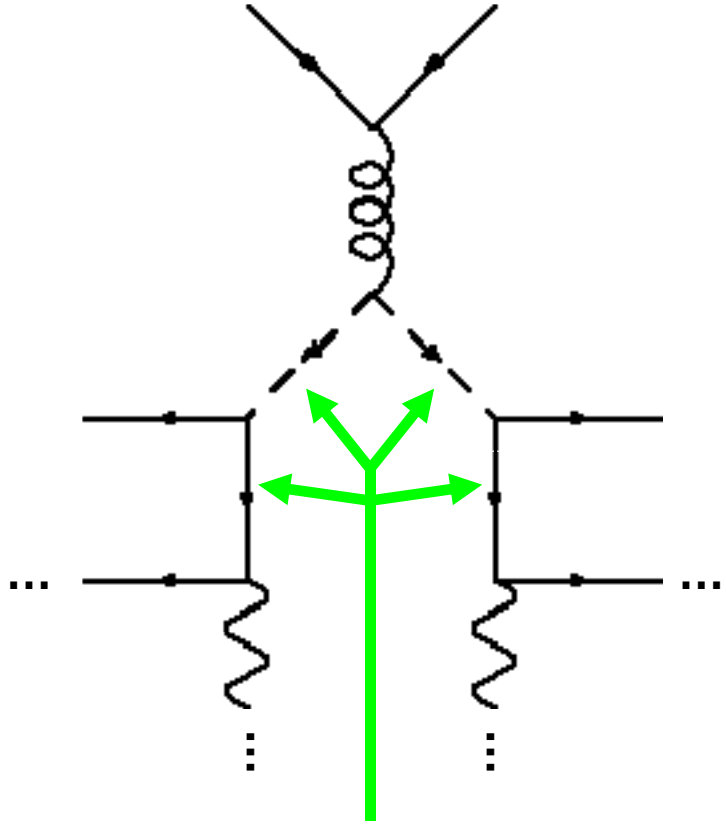
We keep complete quantum coherence by summing over helicity states of intermediate particles

$$\int dm_X^2 dm_Y^2 \cdots d\Pi \, | (pp \to X_{h_X} Y_{h_Y}) \times$$

$$M(X_{h_X} \to X_1 X_2) M(Y_{h_Y} \to Y_1 Y_2) \cdots |^2$$

More about the structure

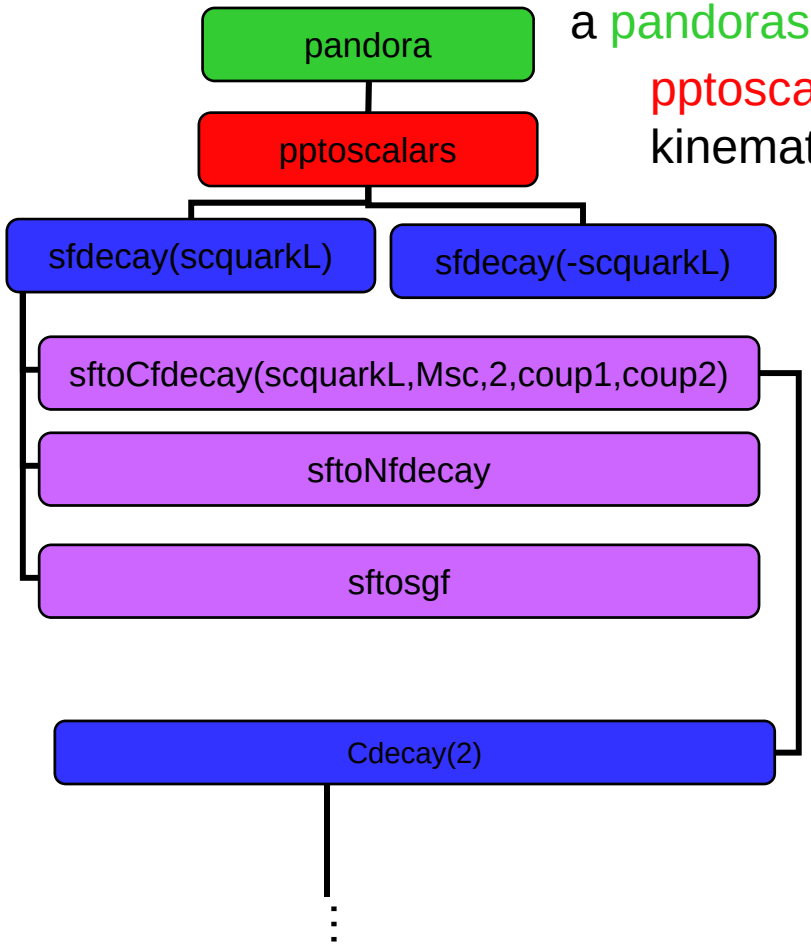
- Calculate 2->2 (and 2->3) helicity amplitudes
- Choose decay channels and attach 1->2 (and 1->3) amplitudes
- Continue down the chain until final massless particles are reached
- Vary decaying particle masses with Breit-Wigner distributions
- Square the constructed matrix elements and integrate to get cross section
- Use the optimized grid from the integral to generate events



$$\frac{1}{m^2 - M_{nom}^2 + iM_{nom}\Gamma}$$

C++ class structure

pandora class instantiates the Monte Carlo integral integral and links to a particular process (several can be linked together in a pandorasBox)



pptoscalars is a process class (inherits kinematics from twototwomm class

sfdecay is a complexdecay class, which stores pointers to all decay channels, and chooses the correct one based on the width

sftoCfdecay inherits helicity structure from StoFfdecay class, which inherits kinematics information from decaytotwomz, and it takes in links to massive particle decays as inputs.

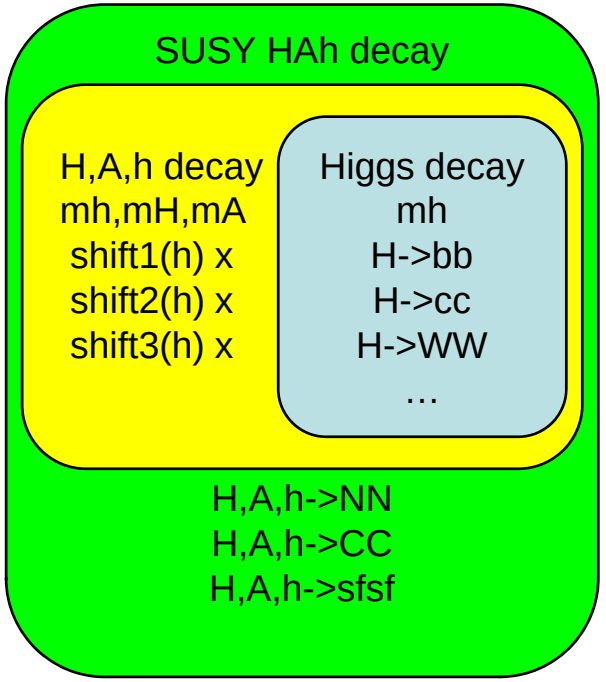
To add your own model only interact with highest level process classes, complexdecays, highest level decay classes (StoSSdecay, StoFfdecay etc.)

Inheritance

Using C++ class inheritance properties, new decays can be built up from SM particle decays or other implemented decays.

For instance to implement neutral parts of doublet Higgs, declare a class that inherits all the SM Higgs decays, simply shifting the mass and couplings appropriately.

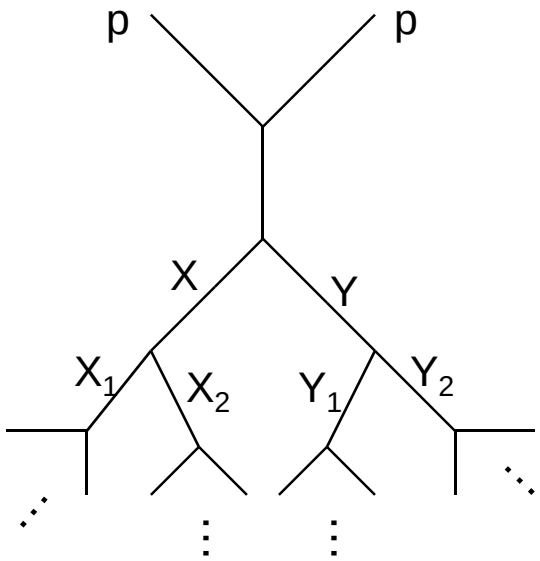
To get the SUSY Higgs, inherit all the doublet Higgs properties, and just add the decays to SUSY particles.



But, there is a problem with this approach …

$$\int dm_X^2 dm_Y^2 \cdots d\Pi \left| M(pp \to XY)\left(M(X \to X_1X_2)/\sqrt{2M\Gamma}\right)\left(M(Y \to Y_1Y_2)/\sqrt{2M\Gamma}\right)\cdots \right|^2$$

$$\times \frac{M\Gamma/\pi}{m_X^2 - M_X^2 + iM_X\Gamma} \times \frac{M\Gamma/\pi}{m_Y^2 - M_Y^2 + iM_Y\Gamma} \cdots$$



This full integral equals the cross section for XY production. It is important, to insure this, that the integral over the decay amplitudes and phase space gives exactly 1.

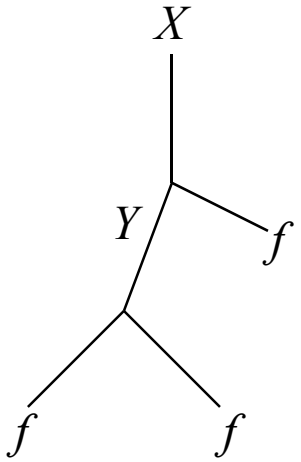
Let's ask if this is really true.

Focusing on the Decay Integral

The width of a channel is just the integral of its amplitude squared

$$\Gamma = \frac{1}{2m}\sum_f \int d\Pi_f \left|M(X \to f)\right|^2$$

$$\int dm_Y^2 d\Pi \left|\left(M(X \to Yf)/\sqrt{2M_X\Gamma_X}\right)\left(M(Y \to ff)/\sqrt{2M_Y\Gamma_Y}\right)\right|^2 \frac{M_Y\Gamma_Y/\pi}{m_Y^2 - M_Y^2 + iM_Y\Gamma} \quad =1$$



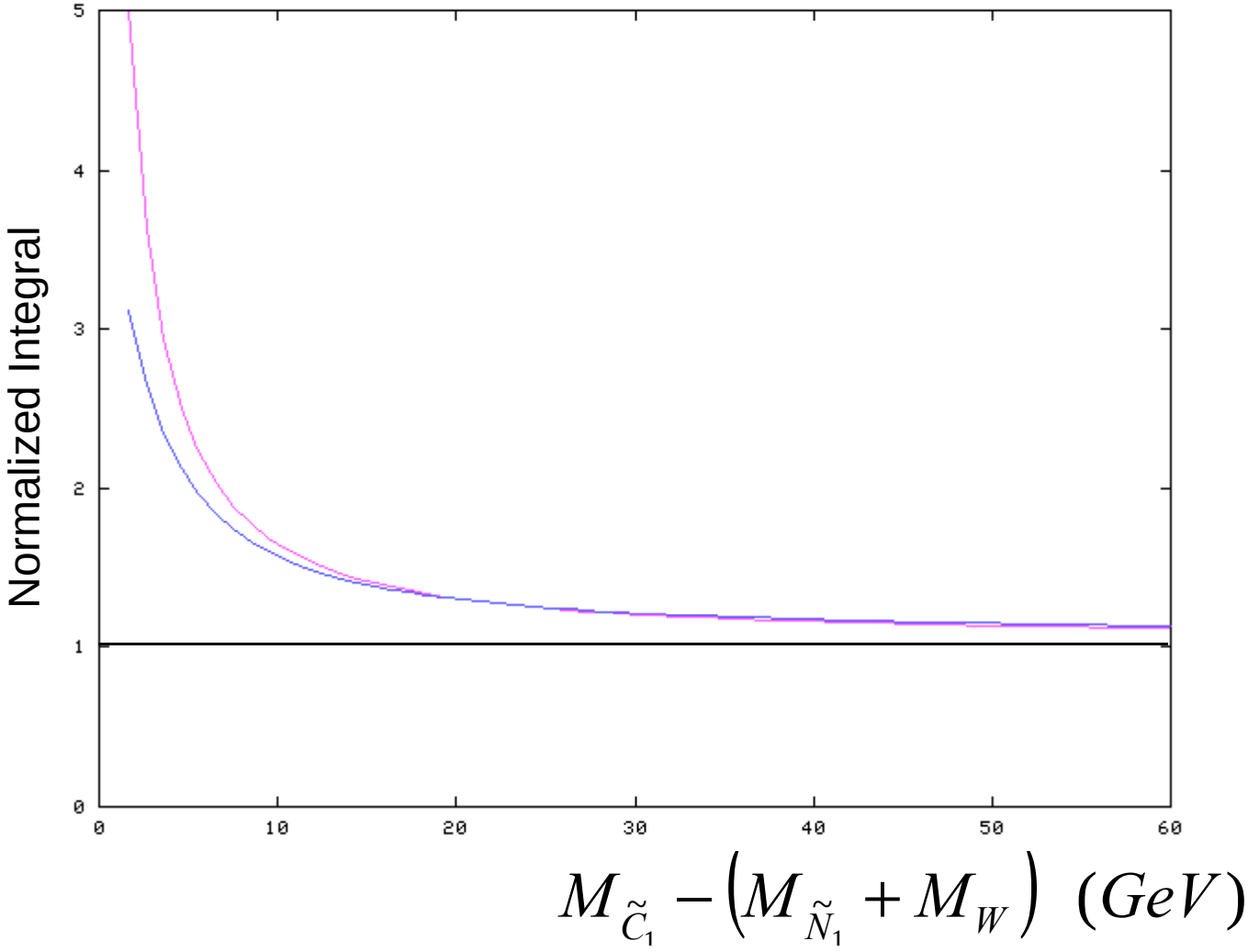
Ideally, normalizing the integral by dividing the amplitude by $\sqrt{\Gamma}$ this integrates to 1. For very narrow widths, this is correct.

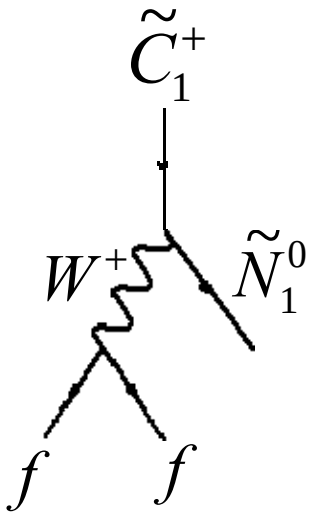
In practice, though, this is not so straightforward.

Not quite right

Red: normalize Gamma to nominal width with all particles on shell

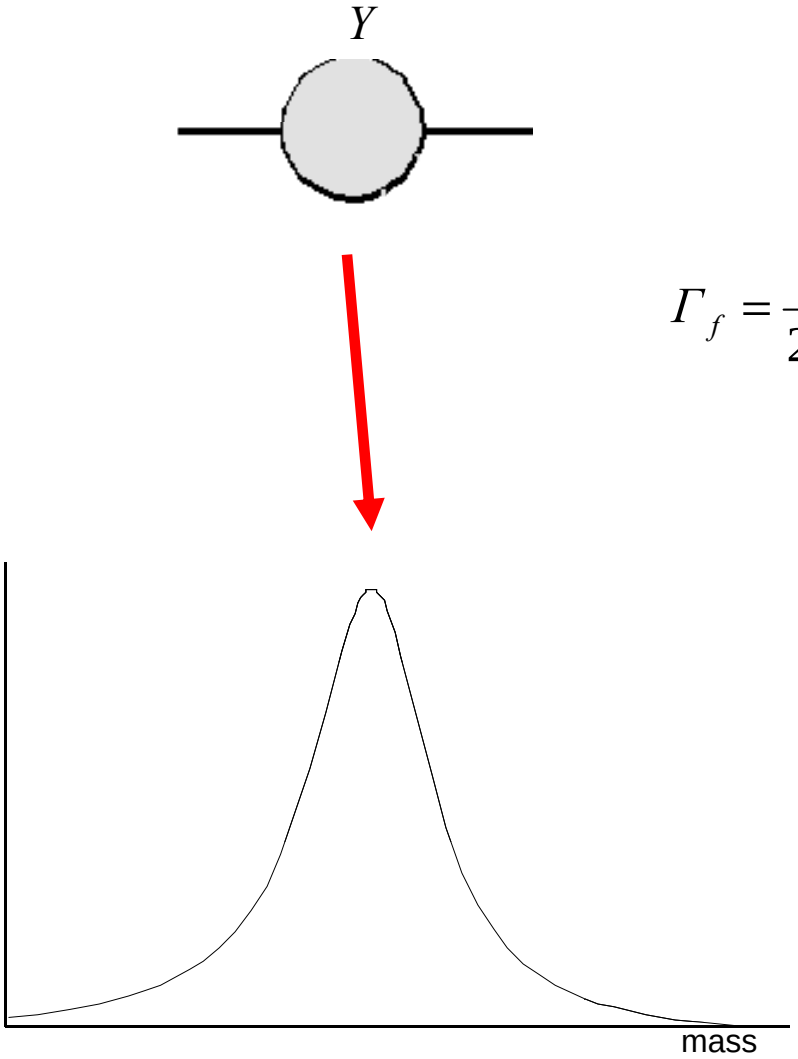
Blue: normalize Gamma to the width computed integrating over a Breit-Wigner distribution for daughter masses at each stage





The Chargino mass is fixed, and the Neutralino is stable, so the W is the only particle whose mass varies.

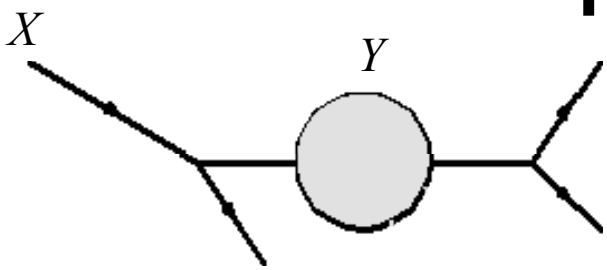
The Problem



The nominal width is an approximation based on a narrow width (almost a delta function)

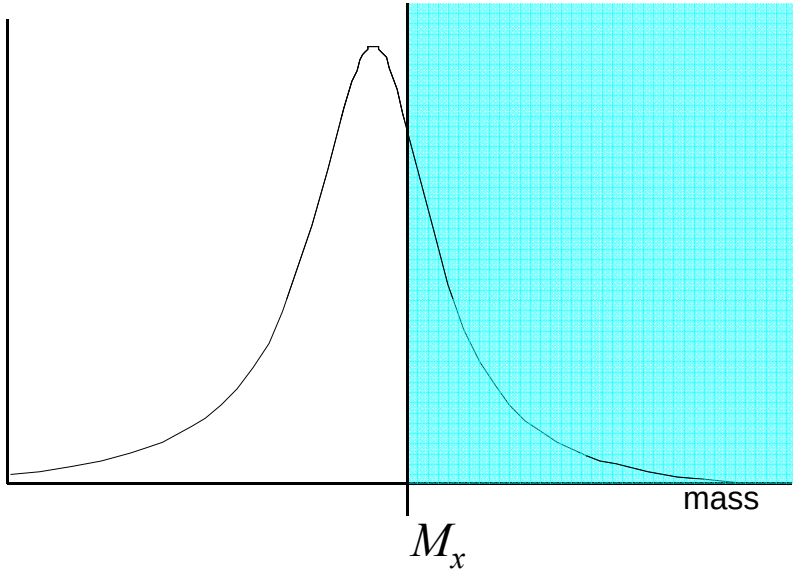
$$\Gamma_f = \frac{1}{2m}\int dm_Y^2 d\Pi_f \left|M(X \to f)\right|^2 \frac{M_Y\Gamma/\pi}{m_Y^2 - M_Y^2 + iM_Y\Gamma}$$

The Problem



The nominal width is an approximation based on a narrow width (almost a delta function)

$$\Gamma_f = \frac{1}{2m}\int dm_Y^2 d\Pi_f \left|M(X \to f)\right|^2 \frac{M_Y\Gamma/\pi}{m_Y^2 - M_Y^2 + iM_Y\Gamma}$$



In practice some portion of the Breit-Wigner distribution is kinematically disallowed.

The kinematic region will change from point to point in phase space, depending on variations in the parent mass, and other daughter masses. (Huge error on left)

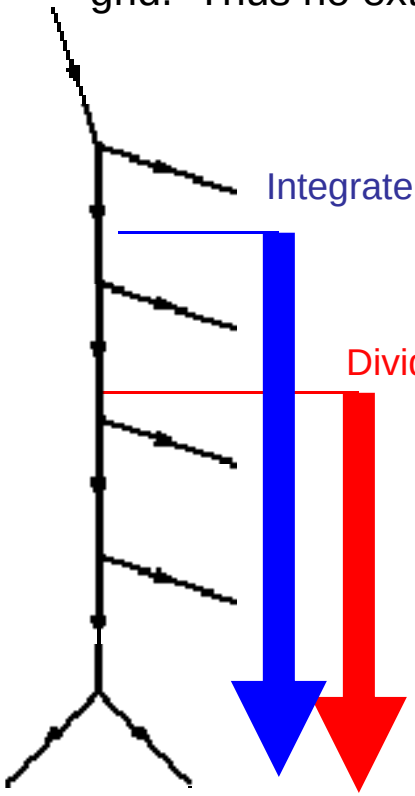
Even if the decay is open, particles move slightly away from mass shell to increase their phase space. (10% error on right)

A Solution?

What we really want is the full integral over the a given two-point function with all the kinematic constraints intact… but that's just a subset of the full pandora integral.

Reading up on Vegas Monte Carlo in Numerical Recipes, there is a formula to evaluate a separate function with similar shape to the main function, without any separate grid. Thus no extra computing time, just extra storage.

$$I_g = \sum_i w_i g(x)$$



Integrate everything below here

Divide by everything below here

Shift Γ to this result

$$\Gamma_{new} = \Gamma_{old} \left(\frac{I_{blue} / I_{red}}{\Gamma_{old}} \right)^{\alpha}$$

Only requires storing more data

Estimate the branching ratios using nominal width for a first pass. Starting at the bottom of the tree, compare the nominal width of the last step to the integral of the last step, and correct the width.

Above that integrate the entire chain from a given level, and divide by the entire integral from one level below it, and correct the width.

Repeat several times as grid refines.

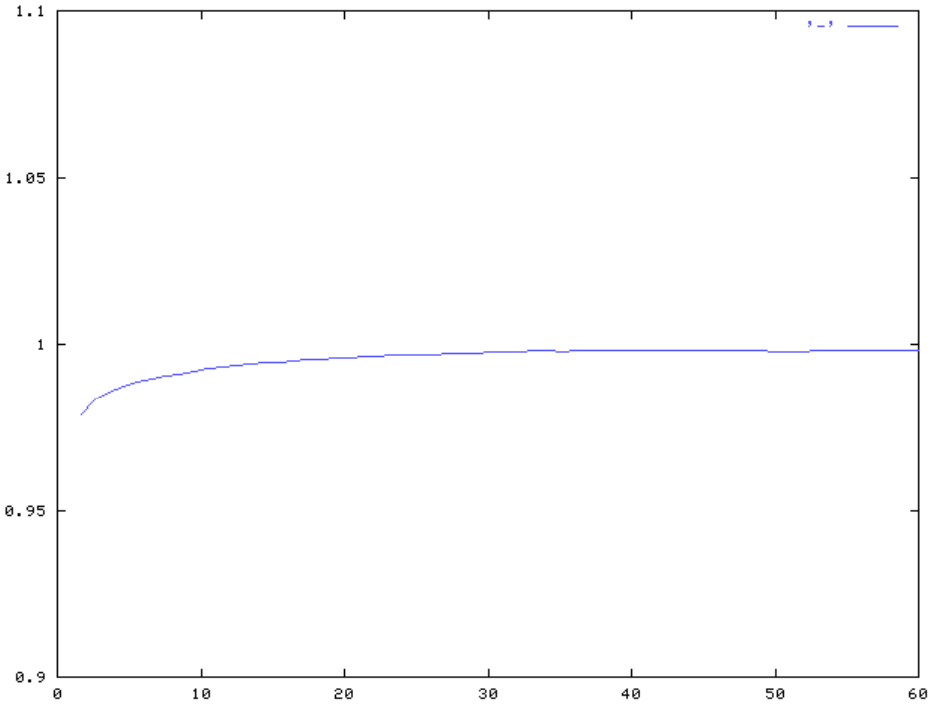
Do we know what's going on?

For a single decay chain this only effects the overall normalization of the integral.

For a complex decay this changes the branching ratios of the choices of the decay.

This changes the function you just adjust your grid to integrate more efficiently.

Is this stable? It seems to be but is not rigorously tested.



This Monte Carlo sampling and adaptation of the function integrated appears to have other physics applications.

Our underlying C++ integrator is built to make this adaptation.

How does this effect other event generators?

What can pandora do?

SUSY implementation is almost completed (but still needs to be verified)

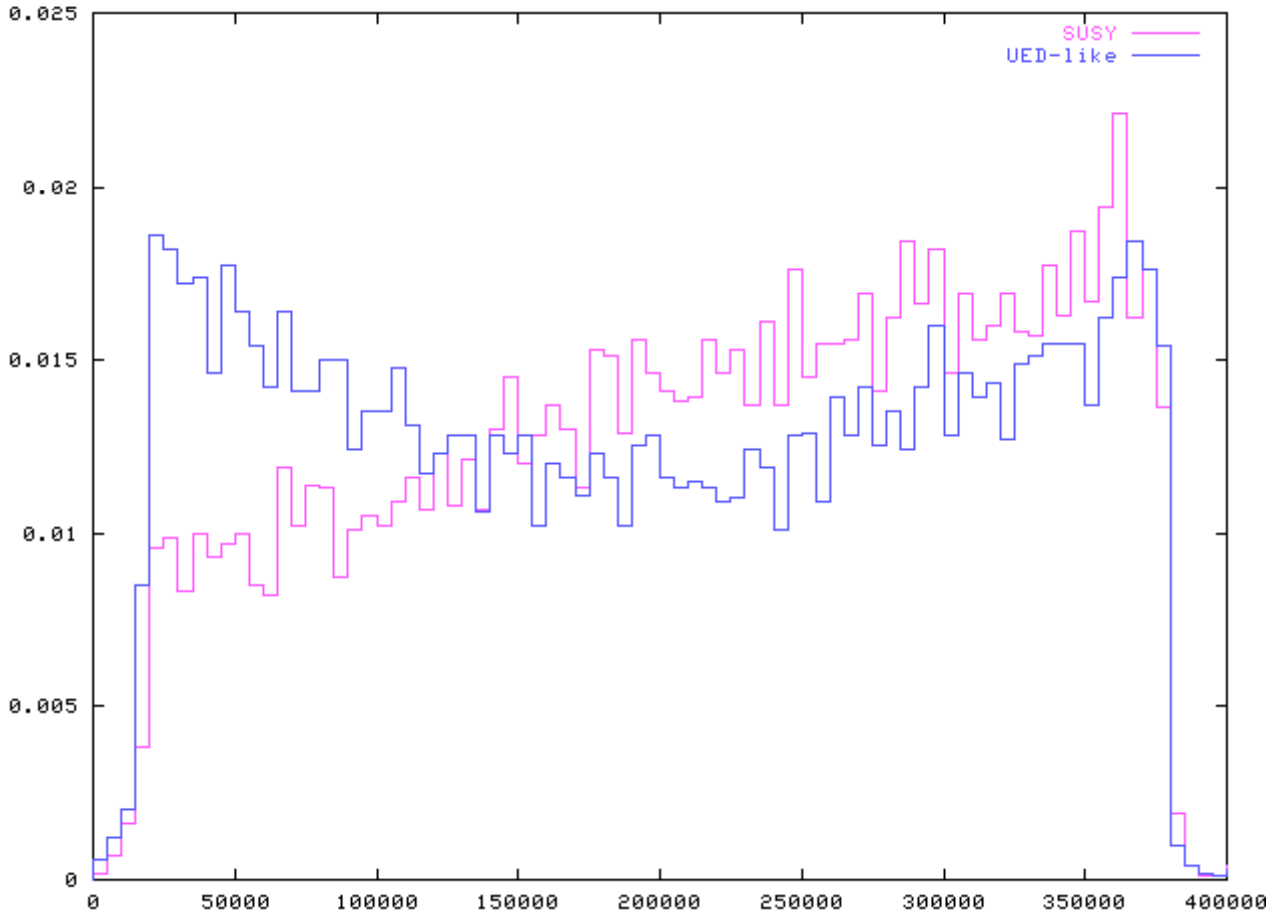
Les Houches accord output implemented, so we can pass pandora output to PYTHIA, PGS. We copy the modular structure of Madgraph.

Two Higgs doublet and OSET also implemented

UED calculator implemented by Josh Ruderman, still need to implement decay classes

Almost all the helicity structure decay classes implemented (FtoVFdecay, VtoVVdecay…). They are ready to be inherited into new physics decay classes.

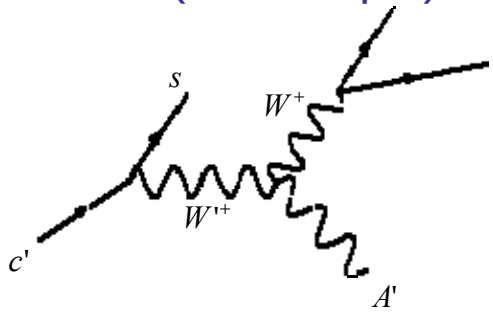
“UED” KK-quarks decays in pandora



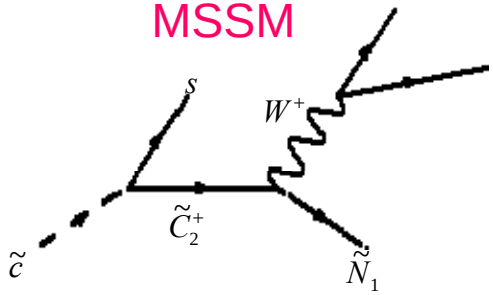
$(p_S + p_W)^2 \quad (GeV^2)$

see Wang, Yavin (2005)

“UED” (Same-Spin)



MSSM

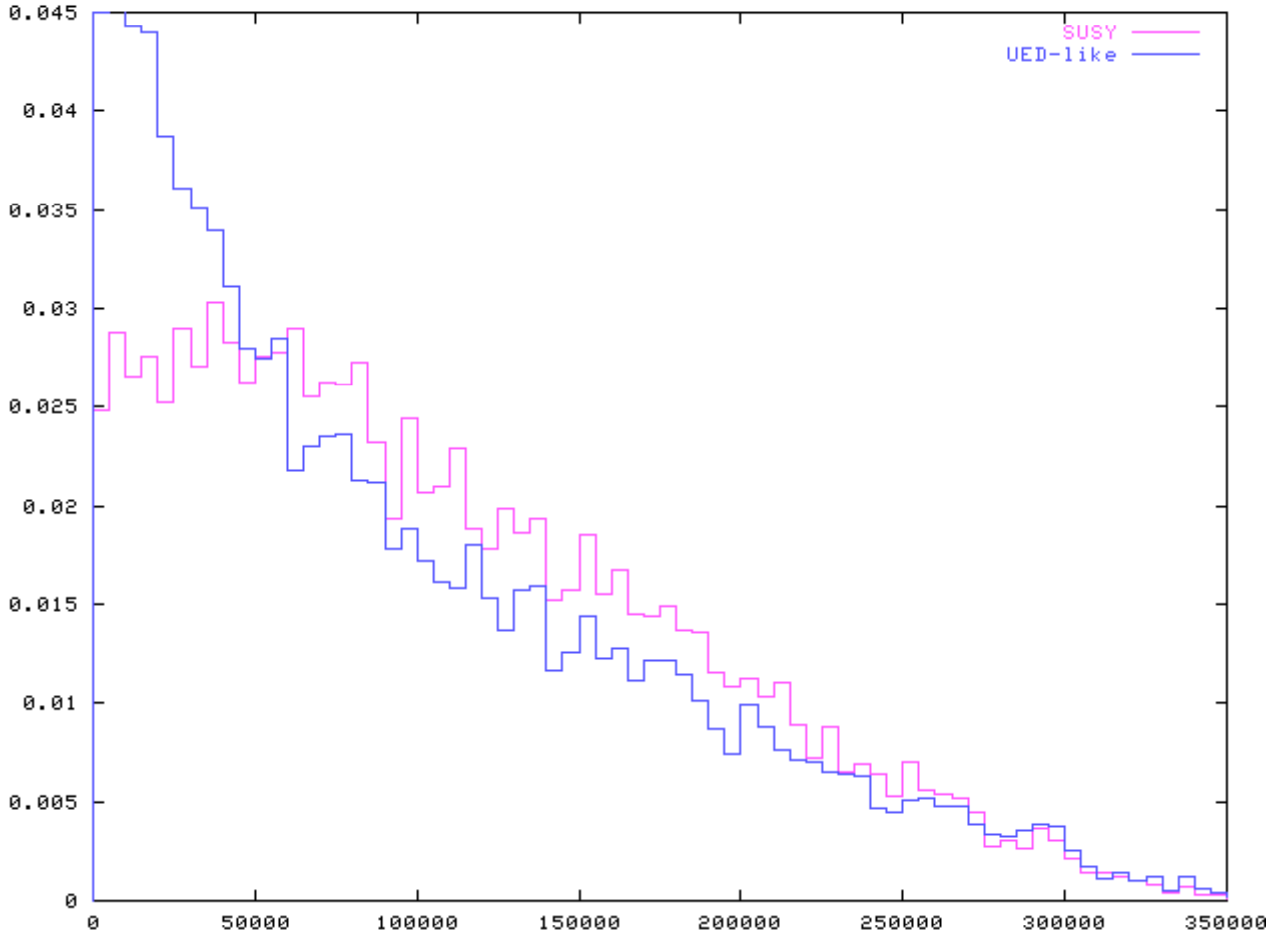


$M_{\tilde{c}} = 800$

$M_{\tilde{C}_2^+} = 500$

$M_{\tilde{N}_1} = 125$

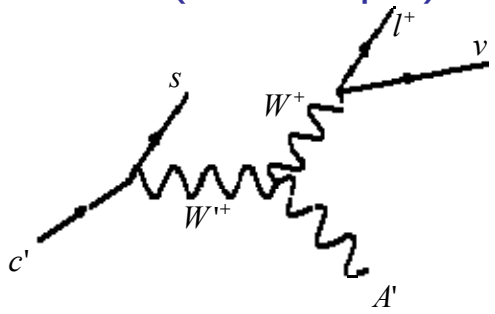
“UED” KK-quarks decays in pandora



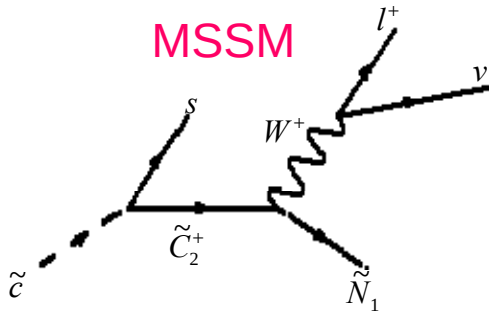
$(p_S + p_l)^2 \quad (GeV^2)$

see Wang, Yavin (2005)

“UED” (Same-Spin)



MSSM



$$M_{\tilde{c}} = 800$$

$$M_{\tilde{C}_2^+} = 500$$

$$M_{\tilde{N}_1} = 125$$

Where to go

Finish it and get it out there:

- Test and debug reworked engine
- Finish SUSY implementation (start testing)

From there:

- Complete UED implementation
- Understand the adaptive nature of the Monte Carlo
- Optimize the generator
- More user friendly model implementation (collaboration on a formalism)

Would love feedback and suggestions (Once we get it to you)

www.slac.stanford.edu/~mpeskin/pandora3.html