Simulation of QCD Showers in pandora

with Yue Chen

and reflecting many discussions
with Staszek Jadach

M. E. Peskin
May, 2003

Goals of the pandora event generator:

Practical tool for fast parton-level event generation
including beam polarization, beamstrahlung,
final state spin effects

Experience with object-oriented programming (C++)
in event generation

Experience with event generation based on helicity
amplitudes

Rethinking methods for event generation
$\rightarrow$ new techniques, new mistakes

pandora is not a precision tool !

It is an important problem to merge higher-order QCD calculations with Monte Carlo QCD showers.

PYTHIA, HERWIG generate QCD showers by a Markov process applied to each final (and initial) parton.

then,

include higher-order processes for $Q > Q_0$; shower for $Q < Q_0$

include higher-order corrections as a correction to the first step of the Markov process

(Frixione and Webber)

Negative weights appear; are these a problem ?

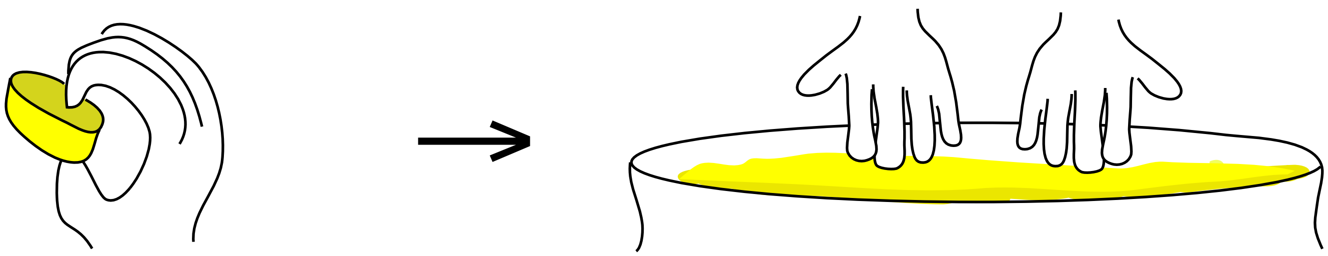
this strategy might provide a better starting point:

view LO QCD shower development as a integral

$$\int d\sigma = \int dx_i \, \frac{d^N\sigma}{\Pi \, dx_i} \qquad N = 5N_g$$

cover all of phase space

get all the random variables at once:

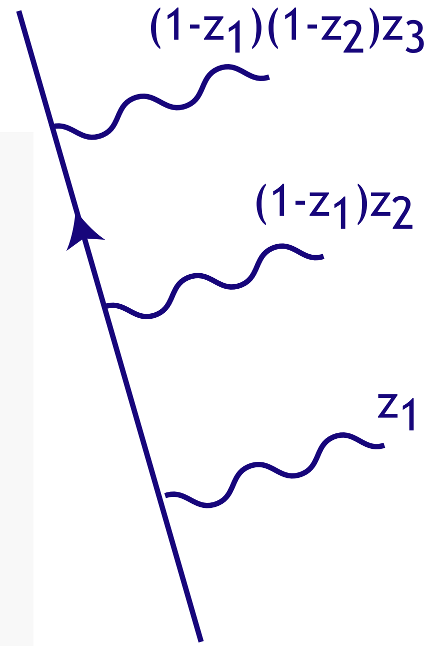


use an effective multi-dimensional integrator for MC

reweight events to implement higher-order corrections

LO multigluon emission:

$$\int d\sigma = \left[\int^{\sqrt{s}} \frac{dQ_1}{Q_1}\frac{\alpha_s(Q_1)}{\pi}\int_\epsilon^1 dz_1 \frac{1+(1-z_1)^2}{z_1}\right]$$
$$\left[\int^{Q_1} \frac{dQ_2}{Q_2}\frac{\alpha_s(Q_1)}{\pi}\int_\epsilon^1 dz_2 \frac{1+(1-z_2)^2}{z_2}\right]$$
$$\cdots \qquad \times e^{-S}$$



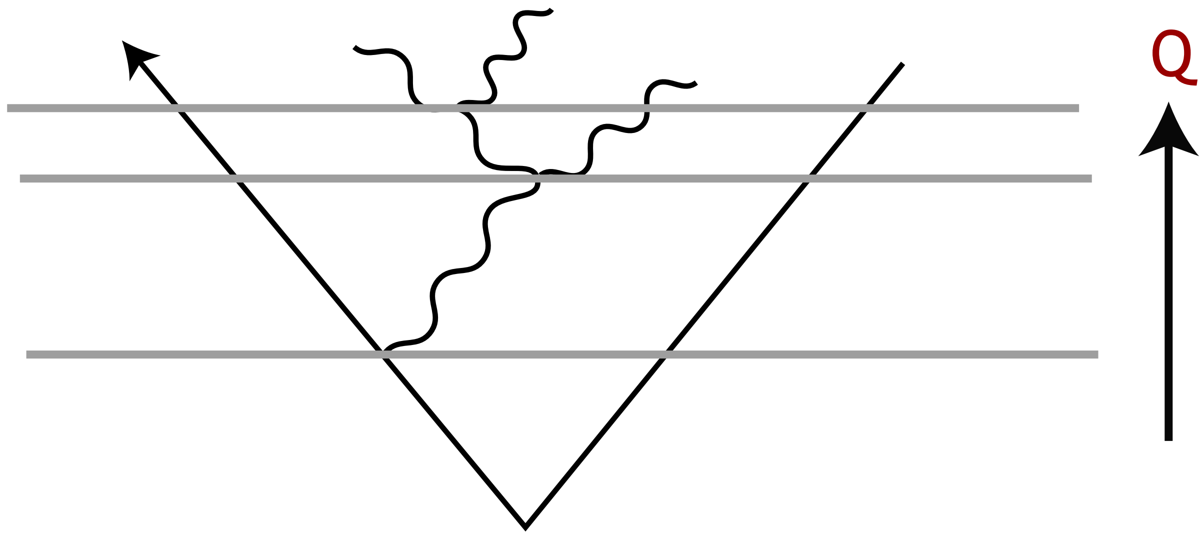
where
$$S = \int^{\sqrt{s}} \frac{dQ}{Q}\frac{\alpha_s(Q)}{\pi}\int_\epsilon^1 dz \frac{1+(1-z)^2}{z}$$
$$= \int dt \int d\xi (1+(1-e^{-\xi})^2)$$

then , summing over n, $\int d\sigma = 1$ exactly

5 variables/gluon:

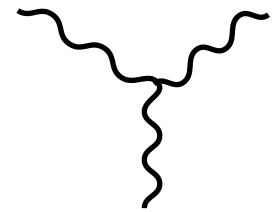
begin with Q_i, z_i, $\square_i$

b_i determines the position of the next branching:



Even with a cutoff on z, this algorithm generates many soft gluons. Use s_i to decide which not to instatiate as final-state particles.

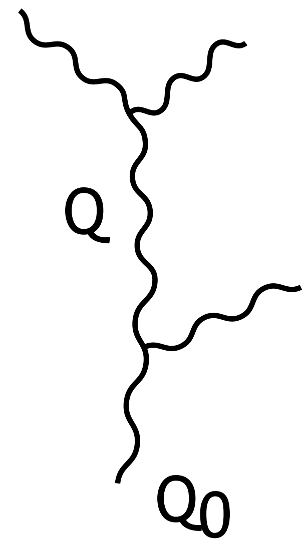
fill all of phase space:



Q = mass of a 2-parton system or
mass of a virtual parton

treat these masses exactly in
kinematics

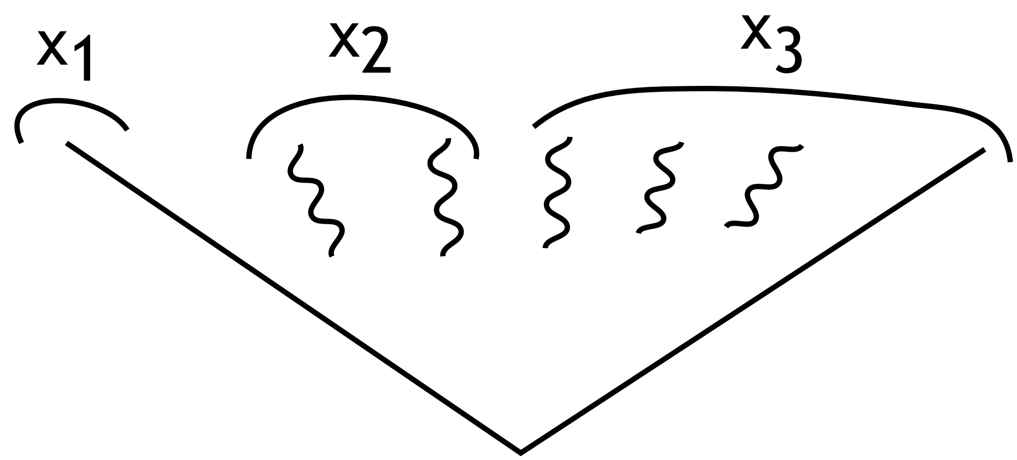
$z = (1 - \cos\theta)/2$ in parton-parton
CM frame



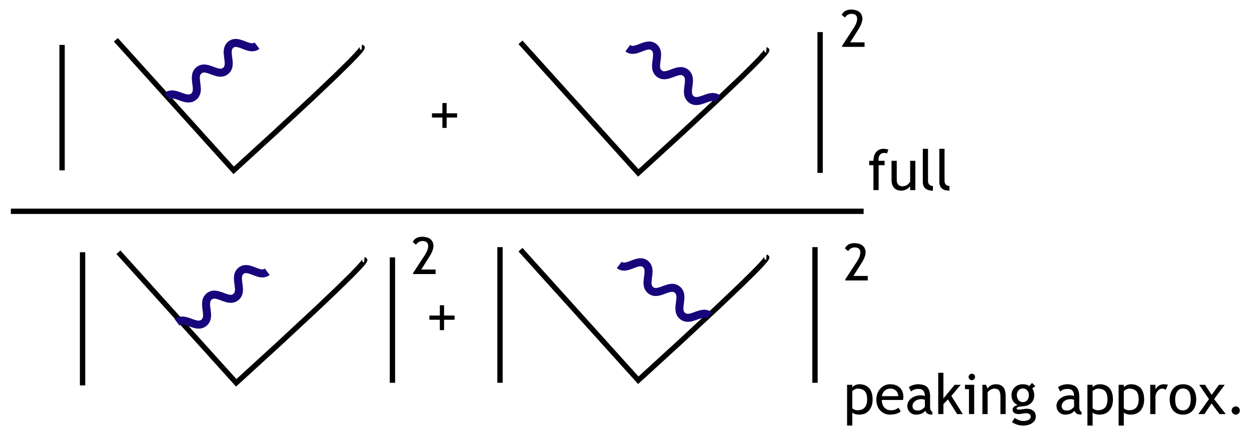
Suppress realization of gluons in the
backward direction: $z < Q^2/Q^2_0$.

Inclusion of exact fixed-order results:

e.g., cluster to 3 jets



reweight by the ratio:



w. the denominator computed without the Sukakov factor.

Can we avoid using negative weights ?

In this approach to QCD showers, one readily generators integrals of 40 dimensions or more.

In FSR, the cross section is relatively smooth in these variables.

However, in ISR, the cross section can vary rapidly with some combinations of variables, e.g., in radiation down to a resonance.

To deal with this, we need a more powerful method for Monte Carlo integration and event selection.

Jadach chooses an adaptive fractal integrator: FOAM.

We are trying an adaptive multigrid strategy: MAVEN.

general principle of Monte Carlo event selection under a probability distribution p(x):

construct a model g(x), let w(x) = p(x)/g(x);
generate points with g(x), keep with prob = w(x)/w(max)

pandora uses a single predetermined coordinate system, with a grid adapted by the VEGAS algorithm (Lepage), as the model g(x). This is also the principle of BASES/SPRING (Kawabata).

For complex reactions, no one coordinate system is adequate; better to use multiple systems (Berends, Kleiss, Pittau).

Even better is an adapted grid in each system:
Ohl's VAMP, the underlying engine in WHIZARD

How do we choose the coordinate systems ?

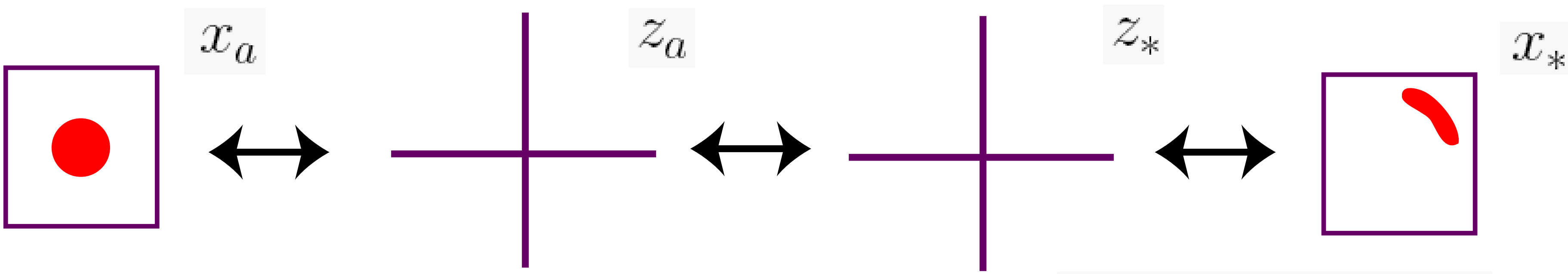
Choose systems appropriate to particular Feynman diagrams.
(Berends et al., Kurihara, Stelzer, ..)

But, this becomes complicated when

computation of p(x) is done module by module

mappings between diffferent coordinate systems are used in the integrator a very different approach (Y. Chen) :

represent the function in terms of features,
introduce a coordinate system for each feature



$$x_i = \frac{1}{1 - e^{-z_i}} \qquad z_a = A_a(z_* - B) \qquad z_i = log\frac{x_i}{(1 - x_i)}$$

giving each feature weight p_a, the model is

$$g(x) = \sum_a p_a \left[\prod_i \frac{x_{ai}(1 - x_{ai})}{x_{*i}(1 - x_{*i})} det A_a \right]$$

Adapt features using the EM algorithm:

sample under g(x), collect points, divide among features according to the weight coming from that feature.

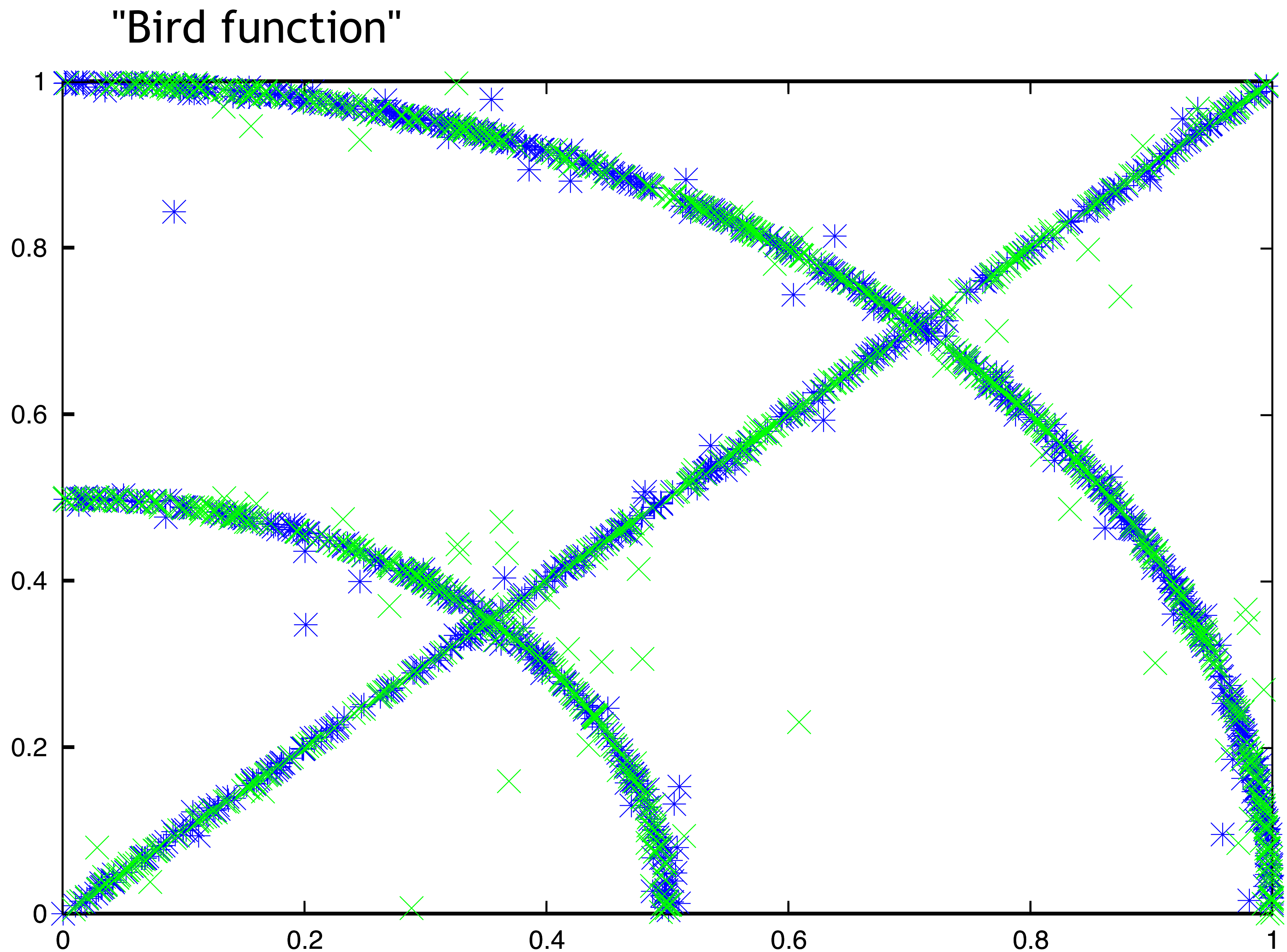
use $<Z_i>$, $<Z_i Z_j>$ to construct new positions and widths of features.

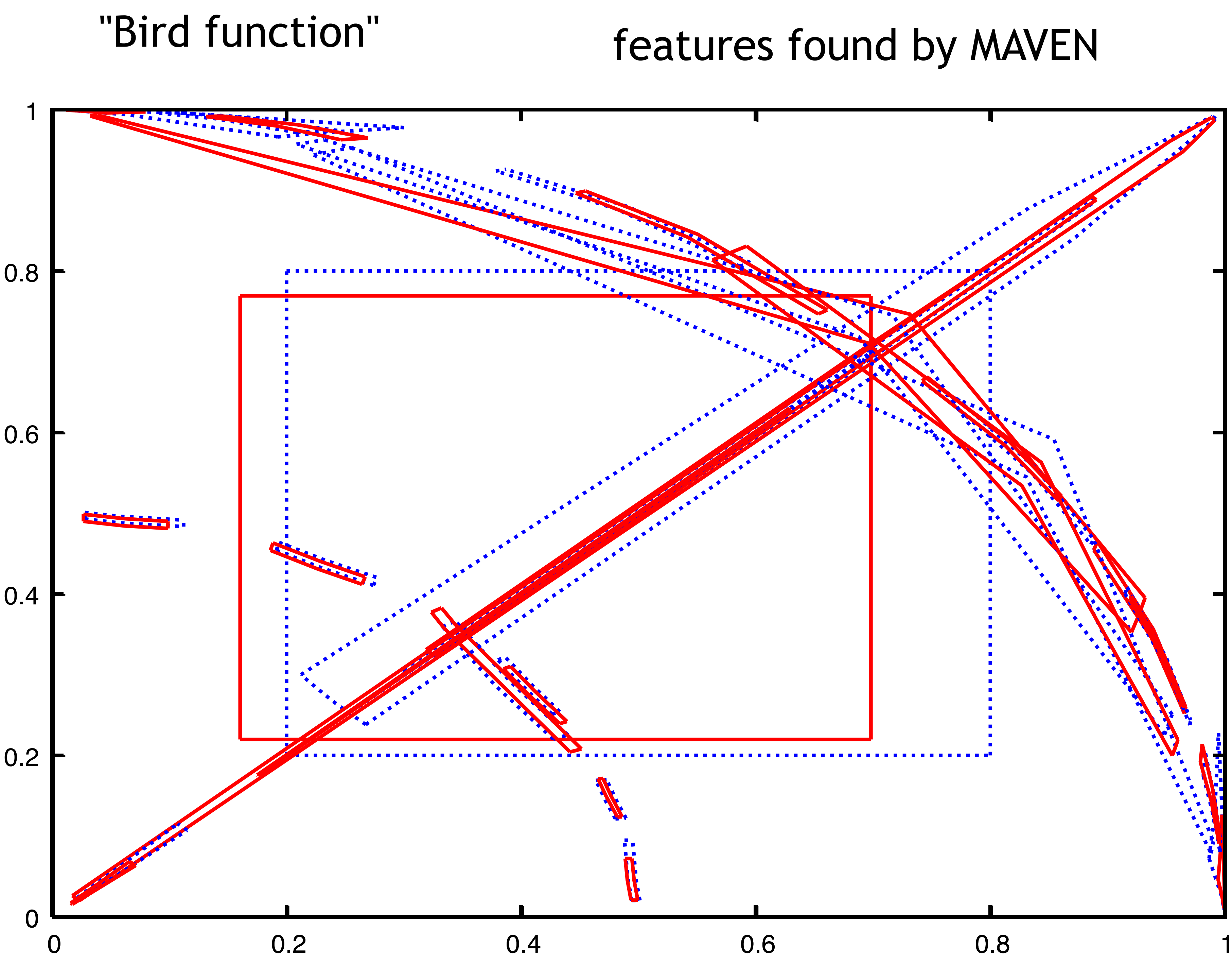
this gives a good model of the function, but one with large (p/g) in some regions.

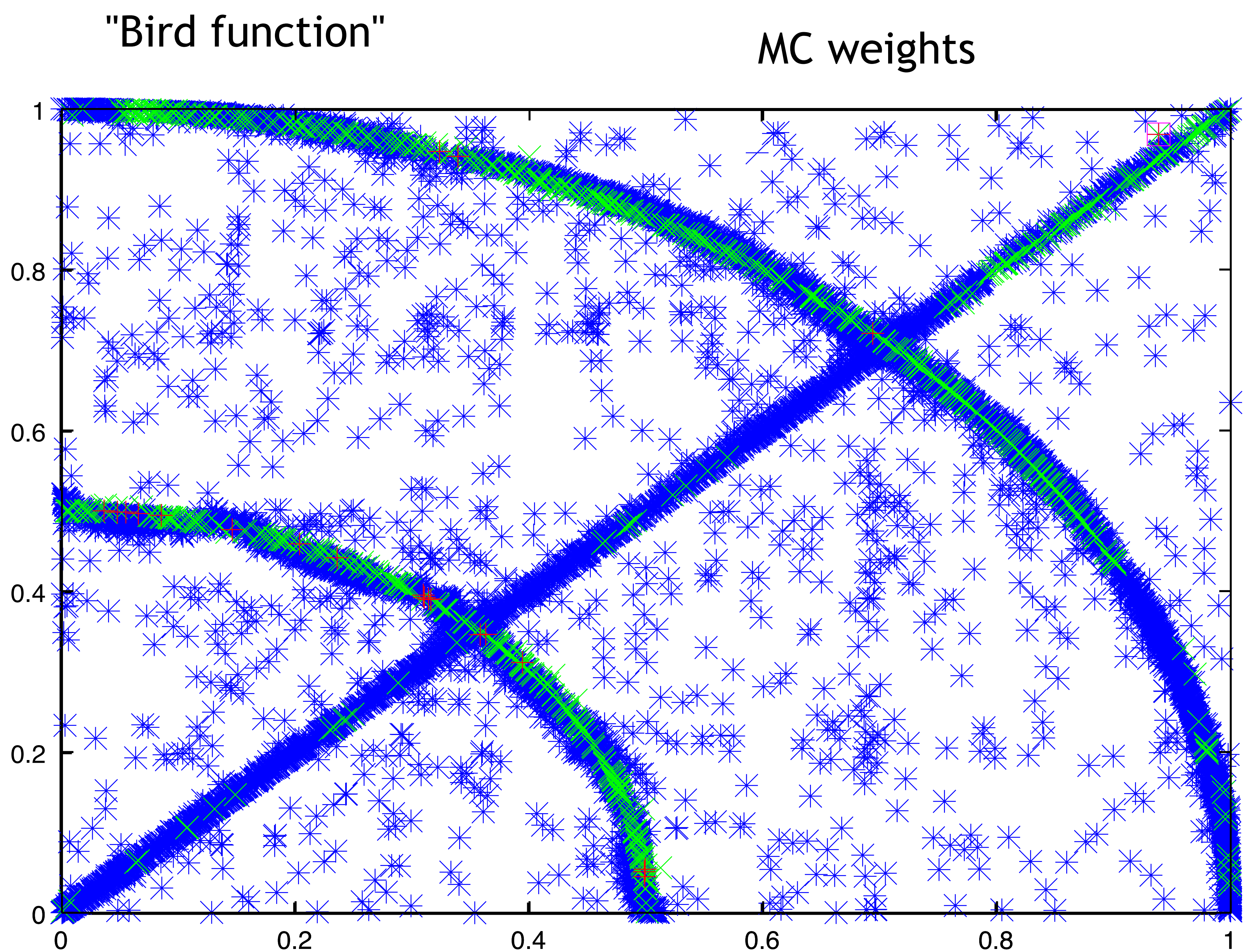
Adapt a grid for each feature:

Each feature is a coordinate system on the integration domain. Associate a VEGAS grid and adapt.

In all, multigrid adaptive integration with VEGAS enhancement (MAVEN)


"Bird function"
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1


"Bird function"
features found by MAVEN
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1


"Bird function"
MC weights
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1

Comparison of VEGAS and MAVEN : 10^6 function calls

	integral	max weight/int.	rms/int.
$e^+e^- \to \mu^+\mu^-$: 2-d integral over ISR			
VEGAS:	1072 (3)	66	2.3
MAVEN:	1073 (1)	18	0.75
$e^+e^- \to \mu^+\mu^-$: full beam effects			
VEGAS:	1141 (8)	562	6.0
MAVEN:	1151 (5)	223	3.2
$e^+e^- \to \nu\overline{\nu}h^0$: no beam effects			
VEGAS:	124.1 (7)	3289	18.2
MAVEN:	126.3 (5)	478	5.7
$e^+e^- \to \nu\overline{\nu}h^0$: full beam effects			
VEGAS:	107.5 (0.1)	3777	17.5
MAVEN:	104.2 (1.2)	1034	12.6

Conclusions ?

These ideas need to be tested in a working code.

I hope to report on that experience soon.