Modern Methods for Simulation & Fitting

Daniel Greenwald

Technische Universität München
Physics Department, E18

International Workshop on Hadron Structure and Spectroscopy
September 5–7, Kloster Seeon

What is our goal?

We want to **efficiently** sample points
in a multidimensional space according to some distribution.

It may be the distribution

- of data simulated according to some model and its parameters

- of parameters within a model explaining data

Regardless of the distribution,
the sampling techniques are the same.

What is our goal?

We want to **efficiently** sample points
in a multidimensional space according to some distribution.

It may be the distribution

- of data simulated according to some model and its parameters

  $\Longrightarrow$ **Monte Carlo Simulation**

- of parameters within a model explaining data

  $\Longrightarrow$ **Analysis**

Regardless of the distribution,
the sampling techniques are the same.

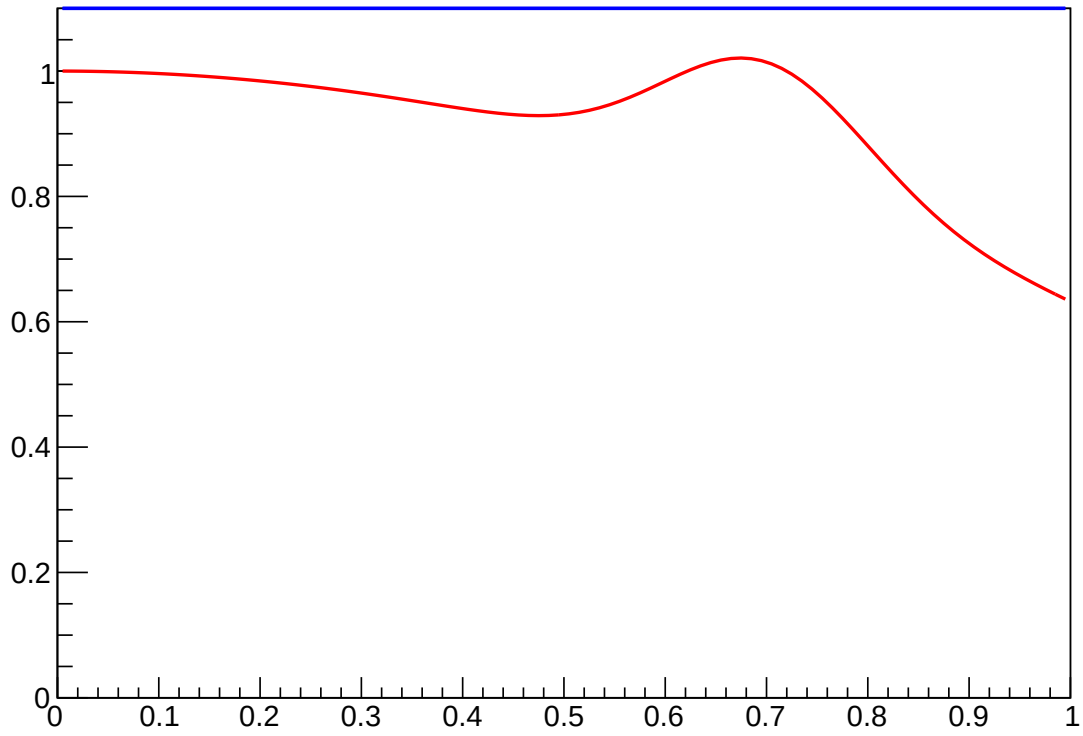
Most Basic Sampling

The two most basic methods of sampling one can—and commonly does—use:

Given desired function f (e.g. model of particle decay)

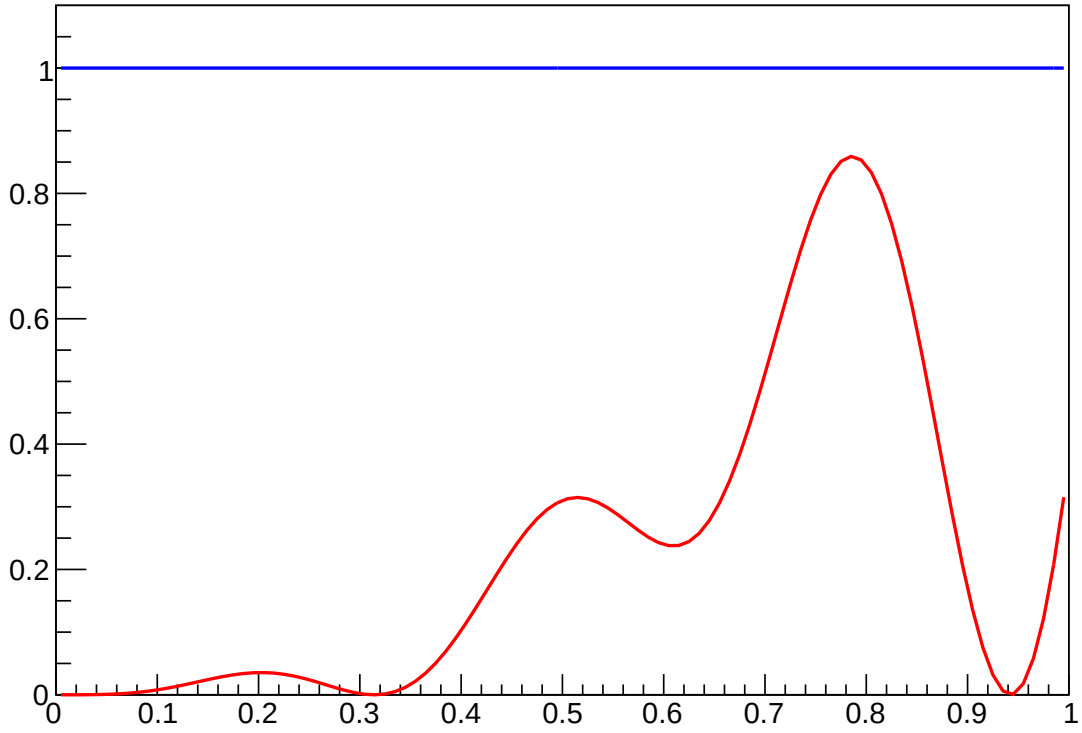
- Importance Sampling
  1. Sample a point according to some cover function, g: $\vec{\mu} \sim g$
  2. Weight point: $w = f(\vec{\mu})/g(\vec{\mu})$
  3. Repeat.
- Hit-or-Miss Sampling
  1. Sample a point according to some cover function, g: $\vec{\mu} \sim g$
  2. Generate uniform random value, r, between 0 and $g(\vec{\mu})$
  3. Add point to data set if $r \leq f(\vec{m})$
  4. Repeat.

Hit-or-Miss / Importance Sampling



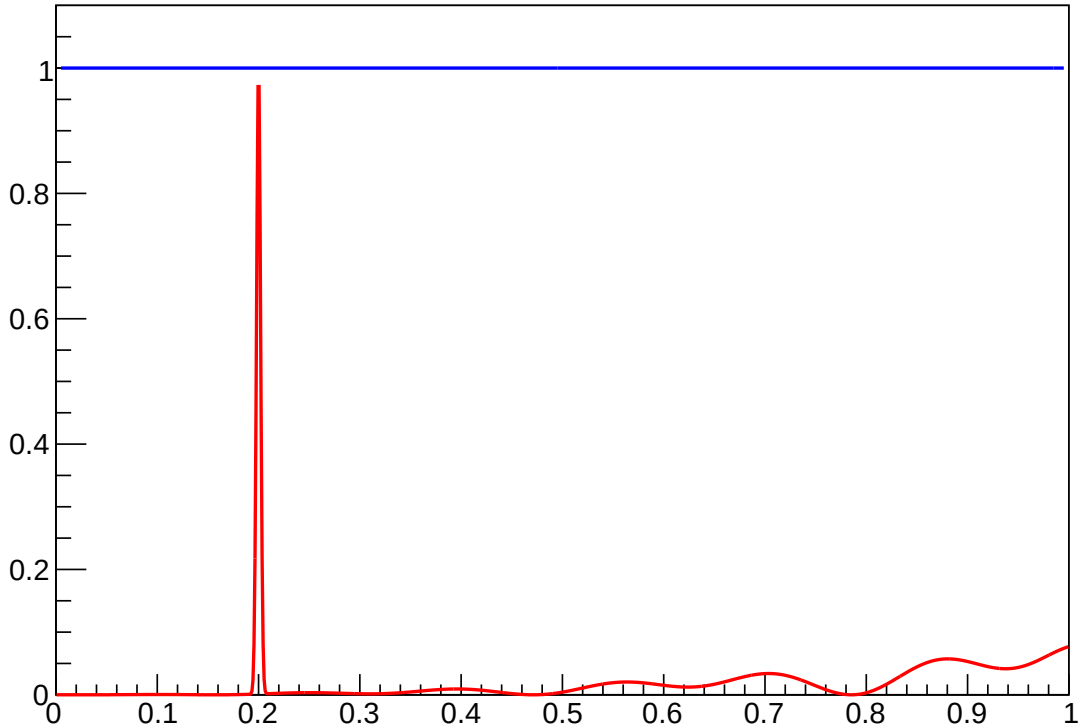
H.o.M. Efficiency $= 84\%$ $=$ Average I.S. Weight

Hit-or-Miss / Importance Sampling



H.o.M. Efficiency = 23% = Average I.S. Weight

Hit-or-Miss / Importance Sampling



H.o.M. Efficiency = 2% = Average I.S. Weight

Hit-or-Miss / Importance Sampling

Yes, that example was rather artificial.

Either your problem will be more well behaved,
or you can choose a wiser cover function.

But, it demonstrates how even in a one dimensional problem,
these methods can easily become ineffecient.

Imagine sampling this way in an n-dimensional problem:

You need to surf an n-dimensional manifold
of an $(n+1)$-dimensional space!

Efficiencies / Average Weights quickly drop to tiny fractions.

Example: Using ROOT-PWA[1] to model

$$\pi + \mathrm{p} \to \mathrm{p} + \pi^+\pi^-\pi^+$$

with $\mathcal{O}(100)$ waves

Generate 170,000 events for COMPASS analysis of F. Krinner, using hybrid of Hit-or-Miss and Importance Sampling:

1. Hit-or-Miss sample (with flat cover) to generate according to 5D-phase-space distribution.
2. Weight all events by model intensity (Importance Sampling)
3. Hit-or-Miss (with flat cover) on weights to cull set.

$\Longrightarrow$ Requires 10,000,000 calls to the model calculation.
$\Longrightarrow$ Requires 8 hours to run.
$\Longrightarrow \approx 60$ calls to model to generate an event.
$\Longrightarrow$ 1.7% efficiency

We can greatly speed this up with a more efficient sampling method.

[1] github.com/ROOTPWA-Maintainers/ROOTPWA

First generalization of simple sampling

The above algorithms are types of **Monte Carlo Markov Chain**:

A sequence of points in which the conditional probability for point given all those before it, depends only on the point directly preceding it:

$$P(\vec{x}_{n+1}|\vec{x}_1, \ldots, \vec{x}_n) = P(\vec{x}_{n+1}|\vec{x}_n)$$

As well, $P(\vec{x}_{n+1}|\vec{x}_n)$ need not depend on $\vec{x}_n$:
(This is typical of the most naive algorithms. For example: Hit or Miss.)

Let's focus on the **Metropolis-Hastings algorithm**

Metropolis-Hastings algorithm

We want to sample according to a function: $f(\vec{x}) : \mathbb{X} \to \mathbb{R}_{\geq 0}$
(for simplicity assume $\mathbb{X} \subset \mathbb{R}^n$)

Random Walk MCMC:

1. from a current point $\vec{x}_i$ propose a new point $\vec{y}$

   Denote the probability of selecting $\vec{y}$ given $\vec{x}_i$ by $T(\vec{y}|\vec{x}_i)$

Metropolis-Hastings algorithm

We want to sample according to a function: $f(\vec{x}) : \mathbb{X} \to \mathbb{R}_{\geq 0}$
(for simplicity assume $\mathbb{X} \subset \mathbb{R}^n$)

Random Walk MCMC:

1. from a current point $\vec{x}_i$ propose a new point $\vec{y}$

   Denote the probability of selecting $\vec{y}$ given $\vec{x}_i$ by $T(\vec{y}|\vec{x}_i)$

2. calculate the **Hastings ratio**:

$$r(\vec{x}_i, \vec{y}) = \frac{f(\vec{y})}{T(\vec{y}|\vec{x}_i)} \Big/ \frac{f(\vec{x}_i)}{T(\vec{x}_i|\vec{y})} = \frac{f(\vec{y}) \cdot T(\vec{x}_i|\vec{y})}{f(\vec{x}_i) \cdot T(\vec{y}|\vec{x}_i)}$$

Metropolis-Hastings algorithm

We want to sample according to a function: $f(\vec{x}) : \mathbb{X} \to \mathbb{R}_{\geq 0}$
(for simplicity assume $\mathbb{X} \subset \mathbb{R}^n$)

Random Walk MCMC:

1. from a current point $\vec{x}_i$ propose a new point $\vec{y}$

   Denote the probability of selecting $\vec{y}$ given $\vec{x}_i$ by $T(\vec{y}|\vec{x}_i)$

2. calculate the **Hastings ratio**:

$$r(\vec{x}_i, \vec{y}) = \frac{f(\vec{y})}{T(\vec{y}|\vec{x}_i)} \Big/ \frac{f(\vec{x}_i)}{T(\vec{x}_i|\vec{y})} = \frac{f(\vec{y}) \cdot T(\vec{x}_i|\vec{y})}{f(\vec{x}_i) \cdot T(\vec{y}|\vec{x}_i)}$$

3. accept/reject $\vec{y}$ with probability $\min(1, r)$

Metropolis-Hastings algorithm

We want to sample according to a function: $f(\vec{x}) : \mathbb{X} \to \mathbb{R}_{\geq 0}$
(for simplicity assume $\mathbb{X} \subset \mathbb{R}^n$)

Random Walk MCMC:

1. from a current point $\vec{x}_i$ propose a new point $\vec{y}$

   Denote the probability of selecting $\vec{y}$ given $\vec{x}_i$ by $T(\vec{y}|\vec{x}_i)$

2. calculate the **Hastings ratio**:

$$r(\vec{x}_i, \vec{y}) = \frac{f(\vec{y})}{T(\vec{y}|\vec{x}_i)} \Big/ \frac{f(\vec{x}_i)}{T(\vec{x}_i|\vec{y})} = \frac{f(\vec{y}) \cdot T(\vec{x}_i|\vec{y})}{f(\vec{x}_i) \cdot T(\vec{y}|\vec{x}_i)}$$

3. accept/reject $\vec{y}$ with probability $\min(1, r)$

   3.1 if $r \geq 1$, $\vec{x}_{i+1} = \vec{y}$

Metropolis-Hastings algorithm

We want to sample according to a function: $f(\vec{x}) : \mathbb{X} \to \mathbb{R}_{\geq 0}$
(for simplicity assume $\mathbb{X} \subset \mathbb{R}^n$)

Random Walk MCMC:

1. from a current point $\vec{x}_i$ propose a new point $\vec{y}$

   Denote the probability of selecting $\vec{y}$ given $\vec{x}_i$ by $T(\vec{y}|\vec{x}_i)$

2. calculate the **Hastings ratio**:

$$r(\vec{x}_i, \vec{y}) = \frac{f(\vec{y})}{T(\vec{y}|\vec{x}_i)} \Big/ \frac{f(\vec{x}_i)}{T(\vec{x}_i|\vec{y})} = \frac{f(\vec{y}) \cdot T(\vec{x}_i|\vec{y})}{f(\vec{x}_i) \cdot T(\vec{y}|\vec{x}_i)}$$

3. accept/reject $\vec{y}$ with probability $\min(1, r)$

   3.1 if $r \geq 1$, $\vec{x}_{i+1} = \vec{y}$

   3.2 else throw uniform random number in unit interval:

$$a \sim U(0, 1)$$

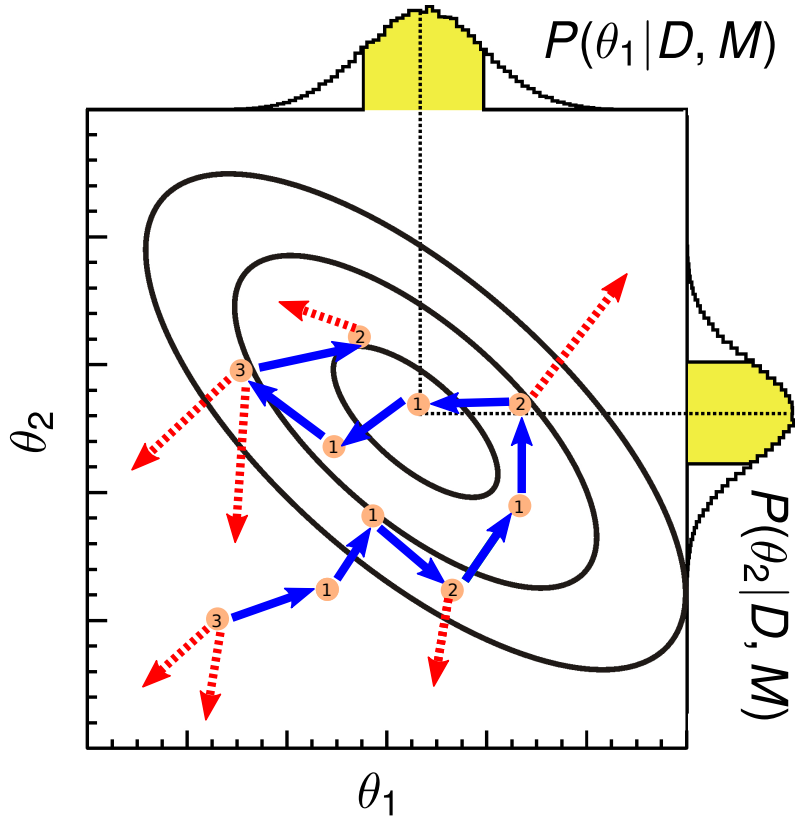
   if $a \leq r$, $\vec{x}_{i+1} = \vec{y}$;

   else $\vec{x}_{i+1} = \vec{x}_i$

Metropolis-Hastings algorithm

In this way we scan our parameter space, spending our time wisely:

- concetrating on areas of interest
- but not completely avoiding areas of less interest



Metropolis-Hastings algorithm

Some things to note:

1. In the accept/reject state we **always produce a new point**!

   If we reject $\vec{y}$, we accept $\vec{x}_i$ as $\vec{x}_{i+1}$

   This of course introduces auto-correlations,
   which we avoid by applying a lag: taking every n'th sample.

   We cannot simply change the accept/reject step
   It is vital for the functioning of the algorithm.

Metropolis-Hastings algorithm

Some things to note:

2. The algorithm can propose a $\vec{y}_0$ for which $f(\vec{y}_0) = 0$,

   but it will never go to it, since:

   $$r(\vec{x}_i, \vec{y}_0) \propto f(\vec{y}_0) = 0$$

   So from a point for which $f(\cdot) \neq 0$
   we can never reach a point for which $f(\cdot) = 0$,
   regardless of the proposal function

3. $r(\vec{x}_i, \vec{y})$ is **undefined** if $f(\vec{x}_i) = 0$

   but since we can never accept a new point for which $f(\cdot) = 0$,
   we need only insure:

   $$f(\vec{x}_0) \neq 0$$

   That is: we need a suitable starting position.

MCMC in practice

1. We will need to tune the proposal function so that the algorithm is efficient.

   Most common:

   - Gaussian,
   - Cauchy,
   - Student's t

   All require radius:

   - If radius is too large: too often select unlikely points $\rightarrow$ chain becomes ineffecient
   - If radius is too small: though efficient, we take small steps, move too slowly, see only part of parameter space

   So we monitor efficiency:

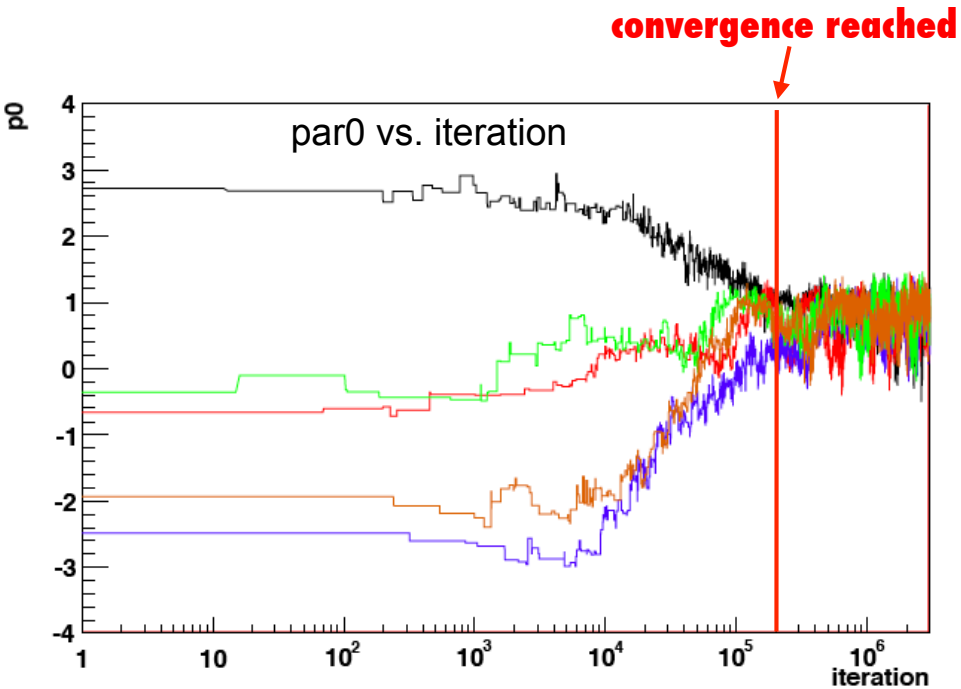
   - if efficiency is too low, decrease radius
   - if efficiency is too large, increase radius

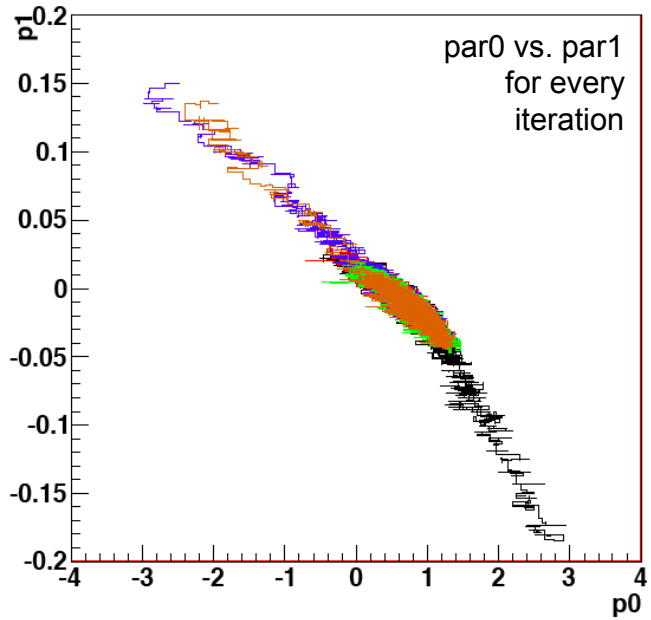
MCMC in practice

2. It takes some initial number of iterations before the Markov Chain **converges** to its equilbrium distribution

   There are many methods for judging whether a chain has converged.

   Many judge graphically.





MCMC in practice

2. It takes some initial number of iterations before the Markov Chain **converges** to its equilbrium distribution

   There are many methods for judging whether a chain has converged.

   Many judge graphically.

   Or judge by modified R values (Brooks & Gelman, 1998)

   1. run several chains for many iterations
   2. for each parameter, calculate

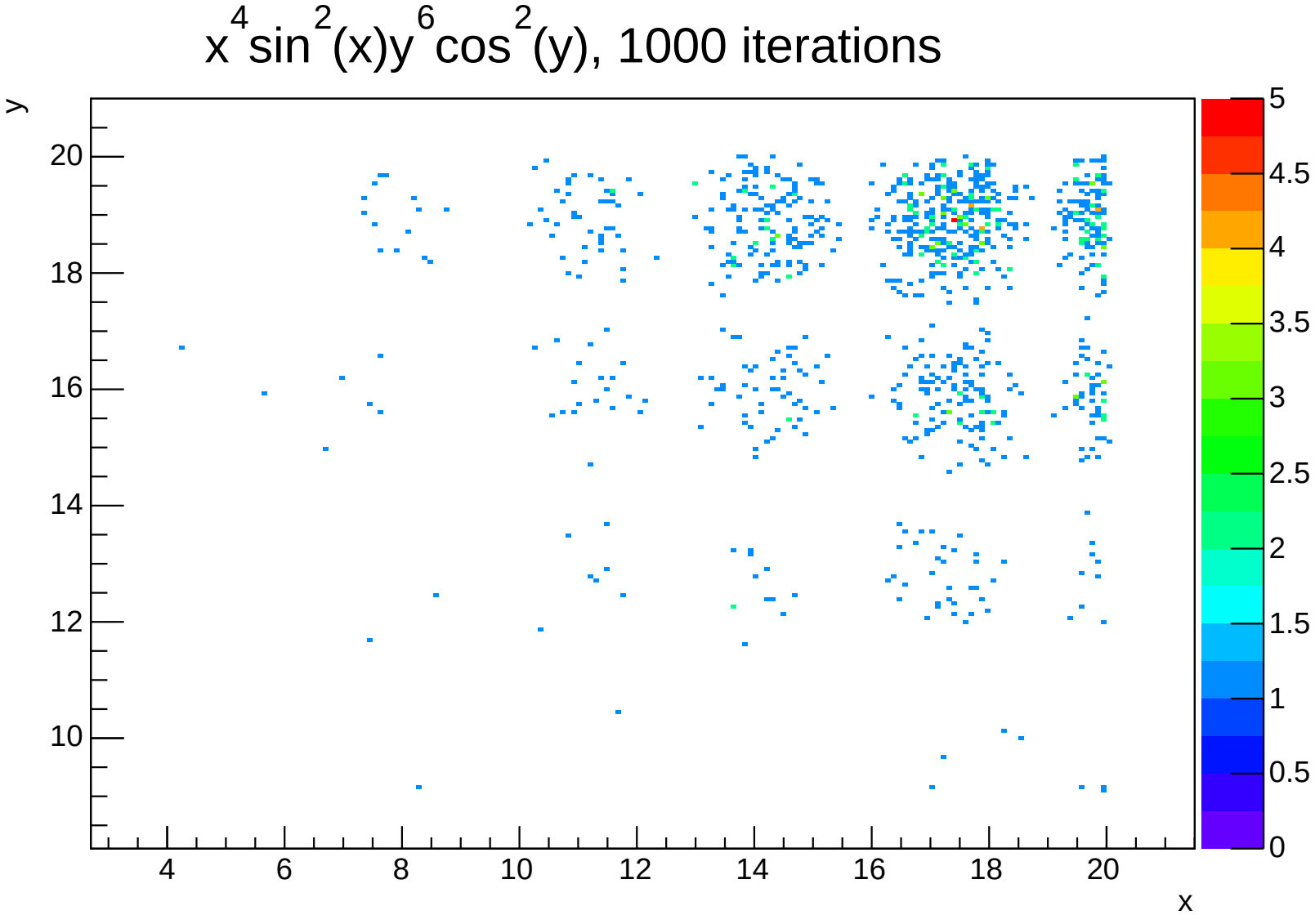
$$R'(\lambda) \propto R(\lambda) = \frac{\text{variance over all samples in all chains for } \lambda}{\text{mean of chains' individual variances for } \lambda}$$

   3. chains have converged when R' is below threshold for all parameters, indicating chains are sampling from true distribution.

MCMC demonstration

Let us look at how well the algorithm attacks equations with multiple peaking structures and dead spaces:

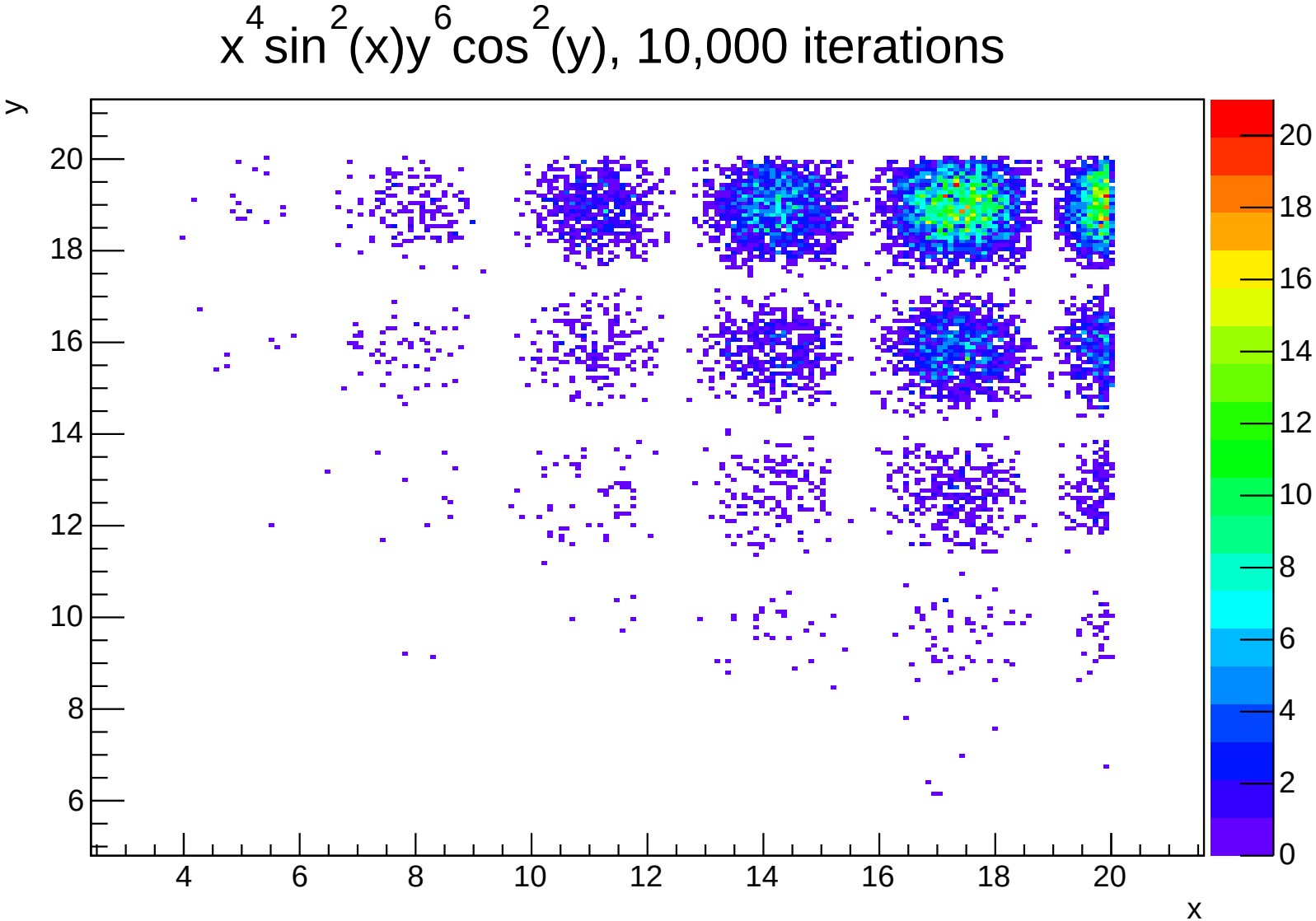
$$f(x,y) = x^4 \sin^2(x) y^6 \cos^2(y), \quad 0 \leq x, y \leq 20$$



MCMC demonstration

Let us look at how well the algorithm attacks equations with multiple peaking structures and dead spaces:

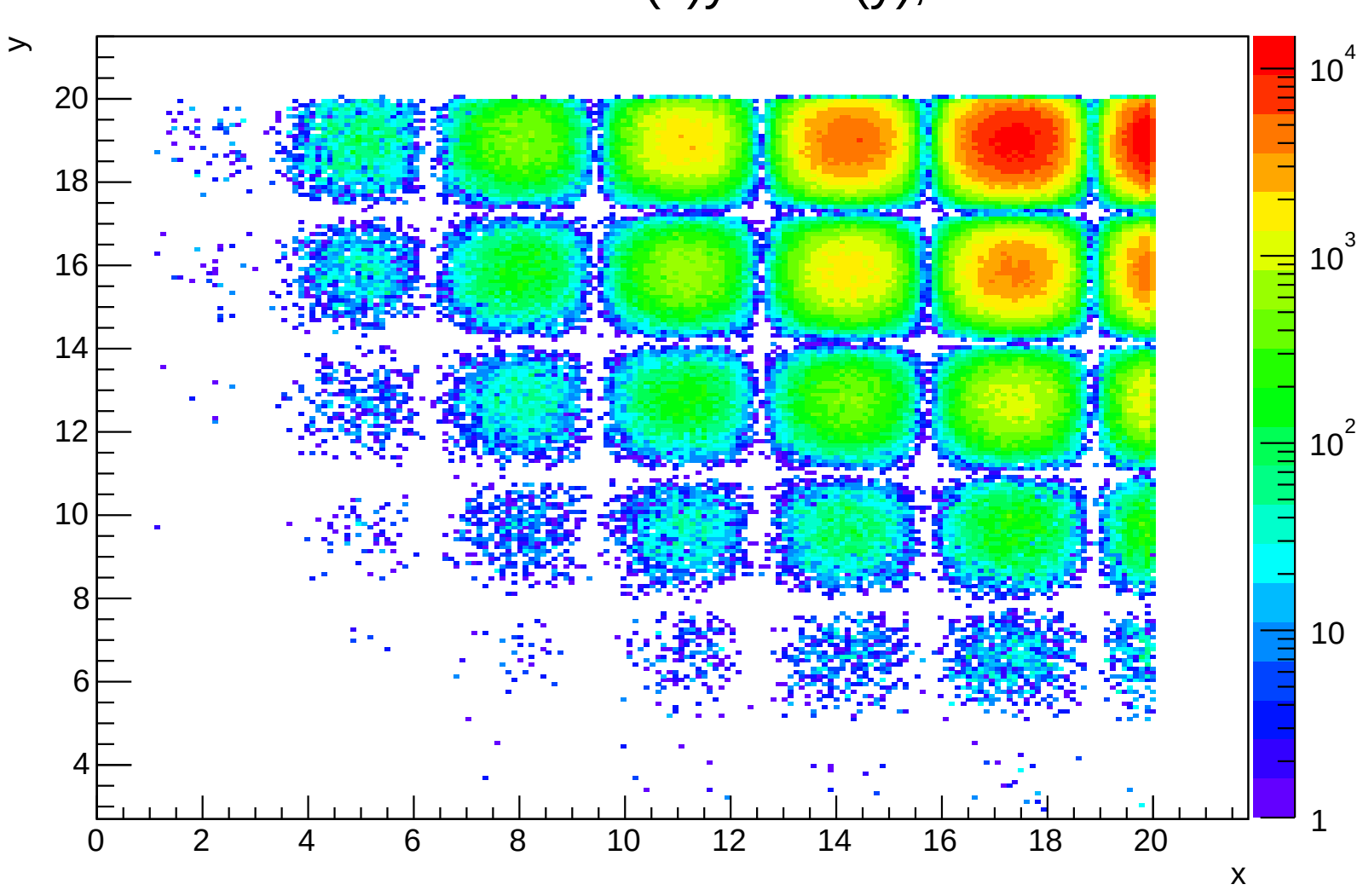
$$f(x,y) = x^4 \sin^2(x) y^6 \cos^2(y), \quad 0 \le x, y \le 20$$



MCMC demonstration

Let us look at how well the algorithm attacks equations with multiple peaking structures and dead spaces:

$$f(x,y) = x^4 \sin^2(x) y^6 \cos^2(y), \quad 0 \le x, y \le 20$$



The Bayesian Analysis Toolkit

Rather than reinventing the wheel, you can do random walk MCMC using BAT:
The Bayesian Analysis Toolkit

BAT is a C++ library developed in the particle-physics community

- It relies on ROOT for data handling and display:
  - So many structures are already familiar to ROOT users
- Being a C++ library, interface to to any other existing code is easy:
  - BAT's structure encourages code separation!

PWA MC in BAT

Using the example from earlier:
Using ROOT-PWA[2] to model

$$\pi + \mathrm{p} \rightarrow \mathrm{p} + \pi^+\pi^-\pi^+$$

with $\mathcal{O}(100)$ waves
to generate 170,000 events

F. Krinner implemented this in BAT:
simply call ROOT-PWA function from inside BAT;
literally a handful of lines of code.

Random-walk MCMC in BAT with a lag of 15:
$\Longrightarrow$ Requires 420,000 calls to model function
$\Longrightarrow$ Requires 20 minutes to run
$\Longrightarrow \approx 2.5$ calls to model per event
$\Longrightarrow 40\%$ efficiency
Reminder: Hit-or-Miss/Importance Sampling—
10,000,000 calls; 8 hours; 60 calls per event; $< 1.7\%$ efficiency

[2] github.com/ROOTPWA-Maintainers/ROOTPWA

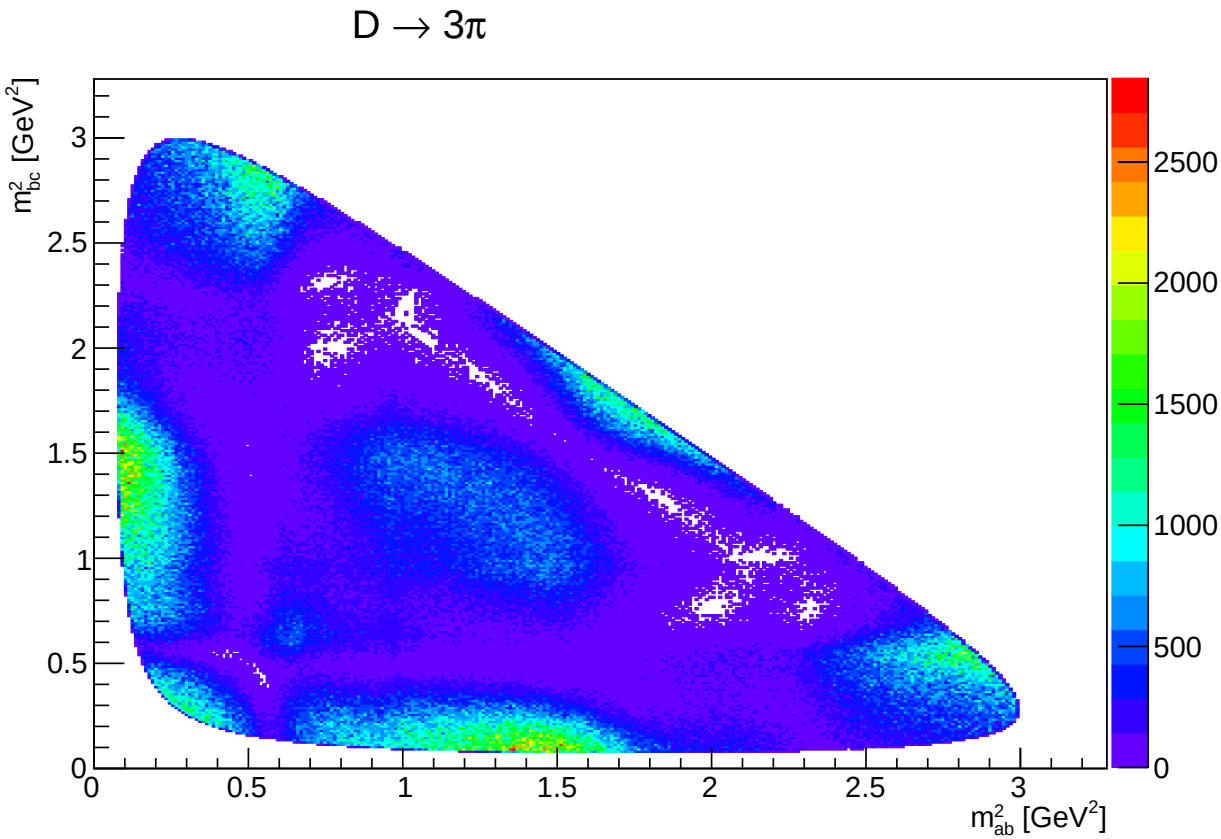
Another Example

Another example from hadron physics: heavy meson decay.

Using YAP[3] to calculate

$$D^+ \to \pi^+\pi^-\pi^+$$

with 6 waves. Sampling with BAT:



12,000,000 events in 59.5 seconds on my laptop.

[3] http://github.com/YAP/YAP

And now for something a little different

Bayes' Theorem:

1. Relate the **joint probability** to the **conditional probability** axiomatically:

$$P(M \cap D) = P(M|D)P(D)$$

2. The joint probability is **commutative**:

$$P(M \cap D) = P(D \cap M)$$

3. From which Bayes' theorem practically falls out:

$$P(M|D)P(D) = P(D|M)P(M) \quad \Longrightarrow \quad P(M|D) = \frac{P(D|M)P(D)}{P(B)}$$

And for the parameters within a model:

$$\underbrace{P(\vec{\lambda}|D;M)}_{\text{posterior}} \propto \underbrace{L(D|\vec{\lambda};M)}_{\text{likelihood}} \cdot \underbrace{P_0(\vec{\lambda}|M)}_{\text{prior}}$$

And for a subset of parameters ($\vec{\theta} \subset \vec{\lambda}$, $\vec{\nu} \equiv \vec{\lambda} \setminus \vec{\theta}$)

$$\text{marginalized posterior: } P(\vec{\theta}|D) \propto \int P(D|\vec{\lambda})P_0(\vec{\lambda})\mathrm{d}\vec{\nu}$$

What tools do you need?

If you only want to optimize your likelihood, you only need an optimizer. For example:

- Gradient following
- Simulated Annealing/Tempering

If you want to

- integrate out "nuisance" parameters,
- compare models
- give non-approximate credibility ranges

you will need a sampler. For example:

- Random-walk MCMC
- Hamiltonian MCMC

MCMC Fitting Example

The same example as previously:
Using YAP to fit to

$$D^+ \to \pi^+\pi^-\pi^+$$

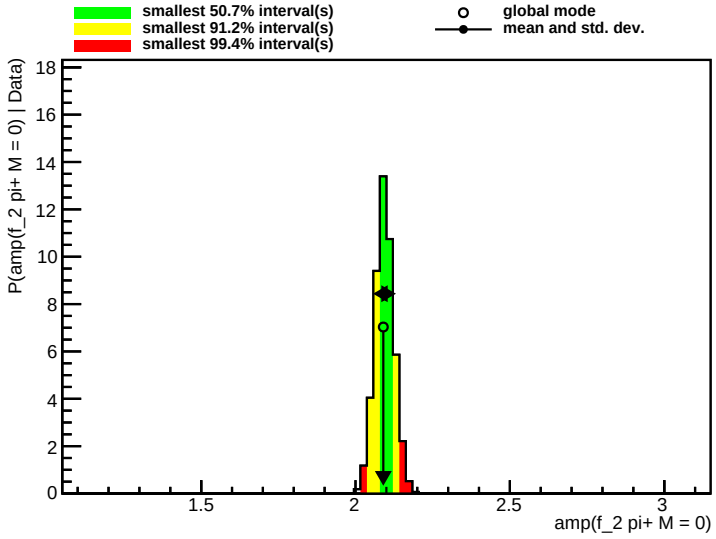
with 6 waves:

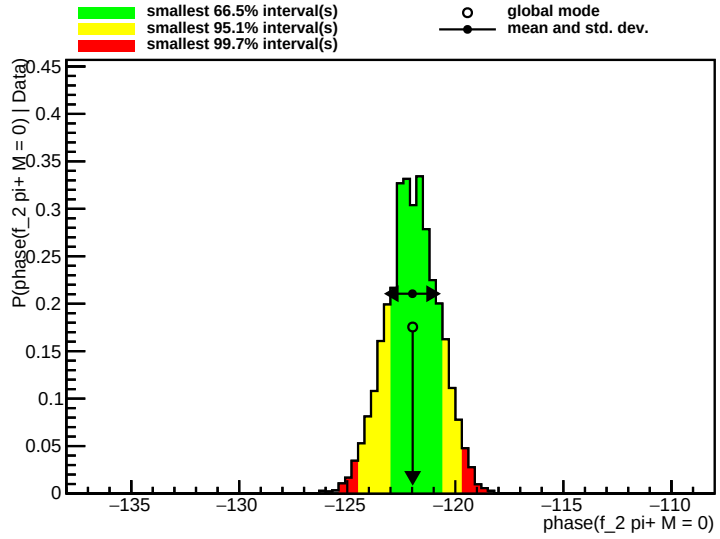
$$\sigma\pi^+,\ f_0(980)\pi^+,\ f_0(1370)\pi^+,\ f_0(1500)\pi^+,\ \rho\pi^+,\ f_2(1270)\pi^+$$

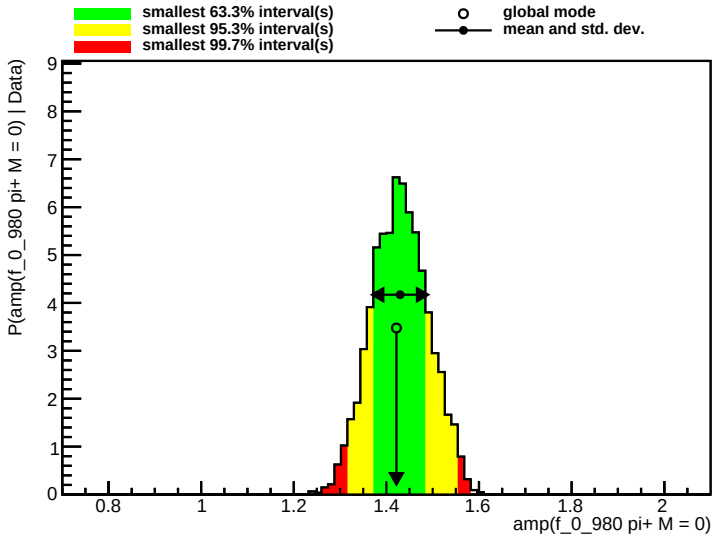
$\Longrightarrow$ 10 free real parameters (5 amplitudes and phases) Fitting 20,000 data points unbinned, sampling with BAT:
$\Longrightarrow$ Requires 12,500 iterations (per chain) for "burn-in."
Then sampled 20,000 parameter points: $\Longrightarrow$ Required 21 minutes for full fitting (including burn-in).
(Again, just run single core on my laptop.)

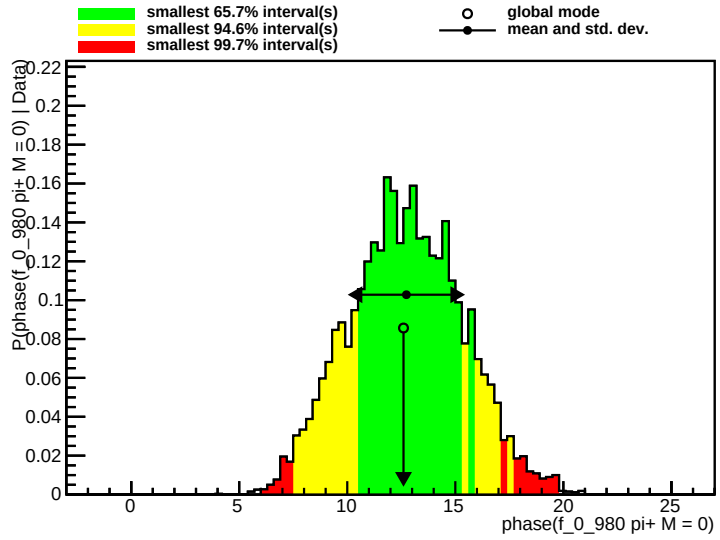
And we get back more than just a best fit:

$D^+ \to \pi^+\pi^-\pi^+$ Posterior



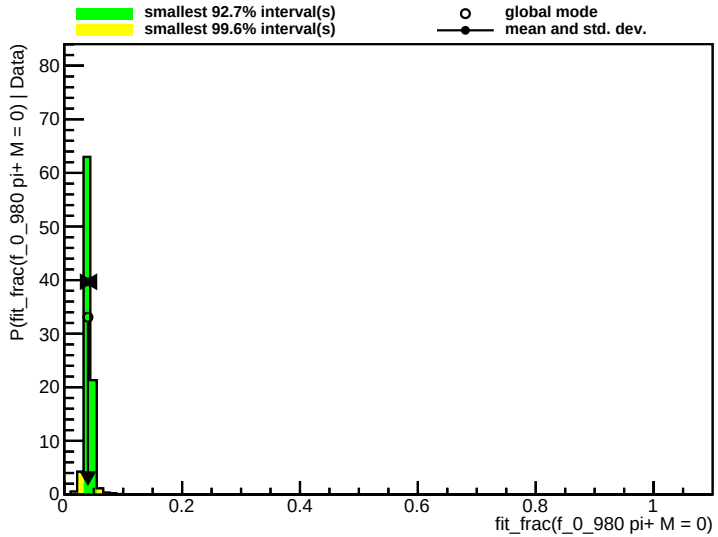


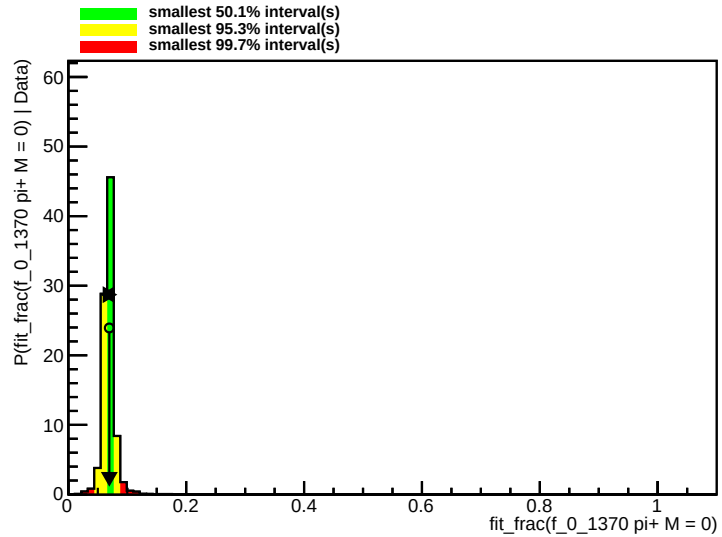


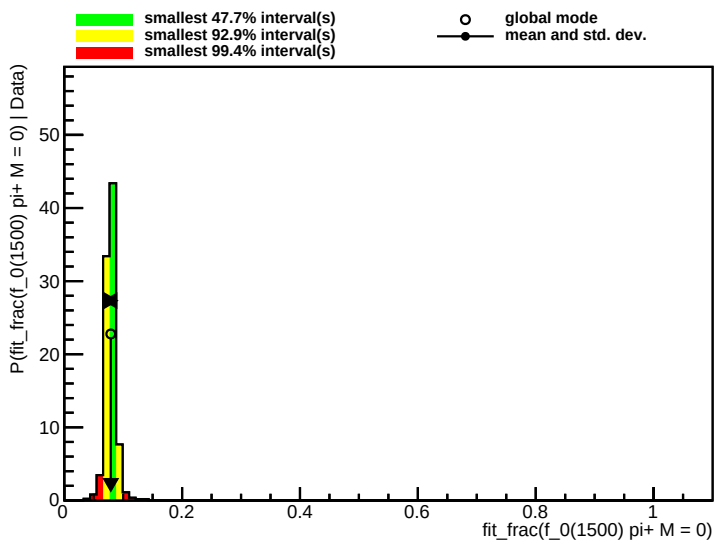


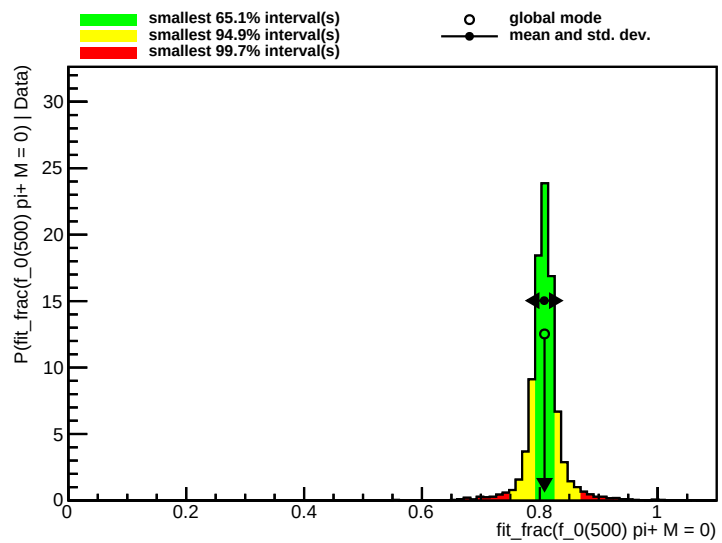
$D^+ \to \pi^+\pi^-\pi^+$ Posterior

We can also plot posterior distributions of functions calculable from the model parameters. For example: fit fractions.



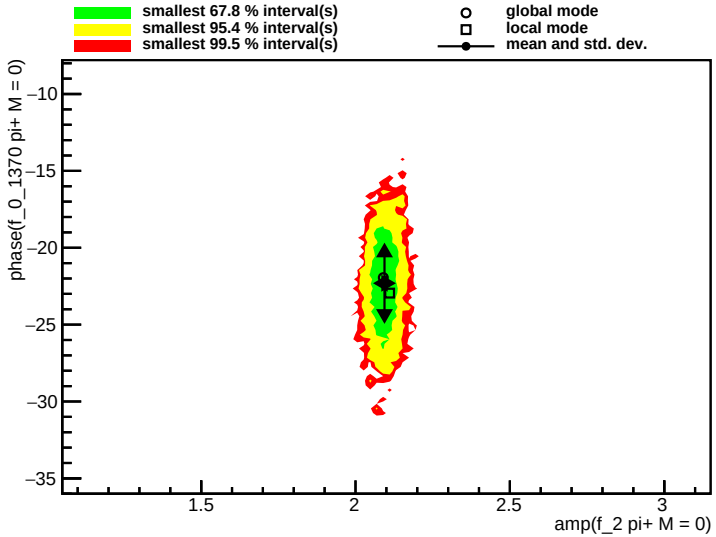


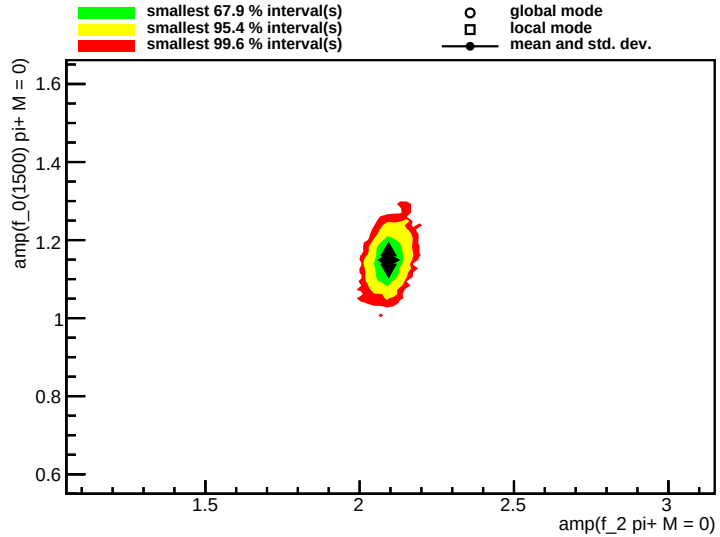


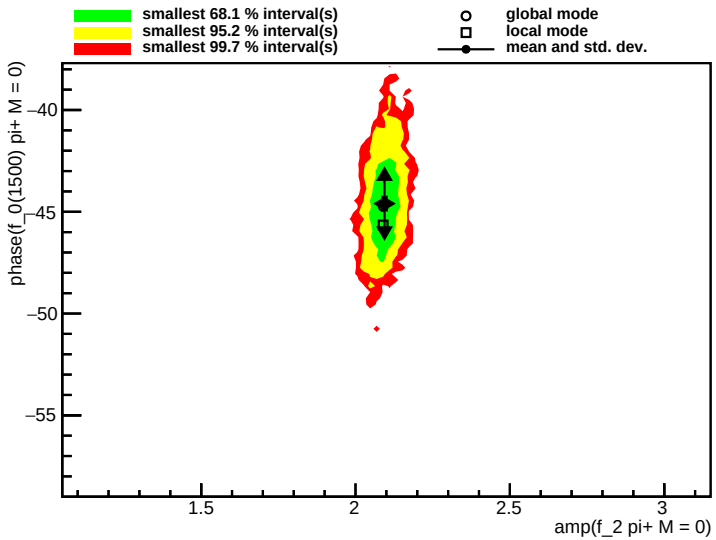


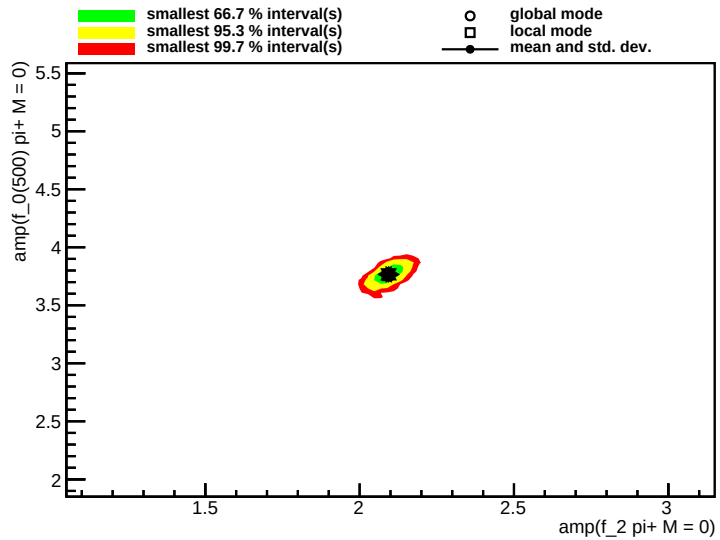
$D^+ \to \pi^+\pi^-\pi^+$ Posterior

Or look at correlations:



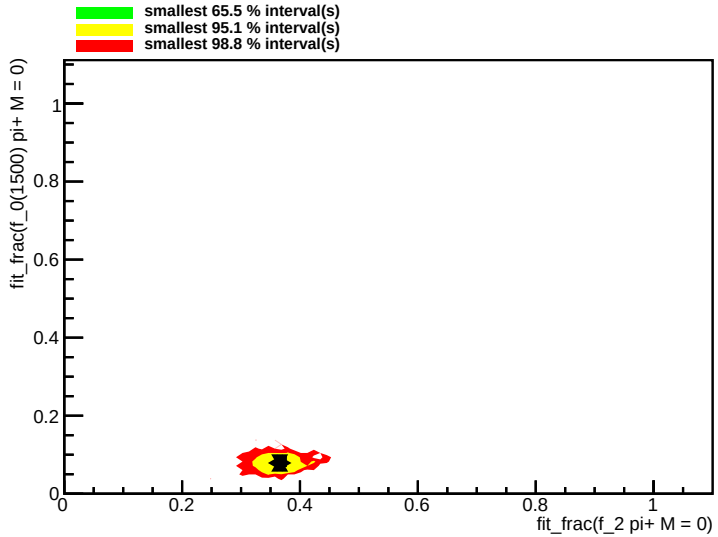


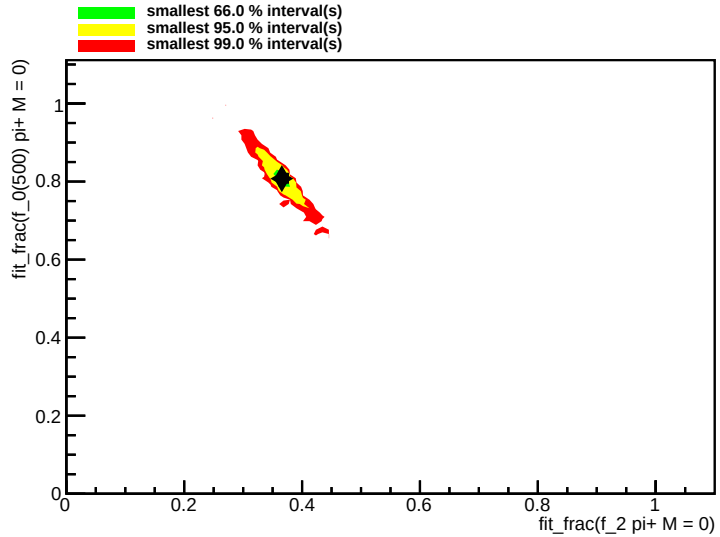


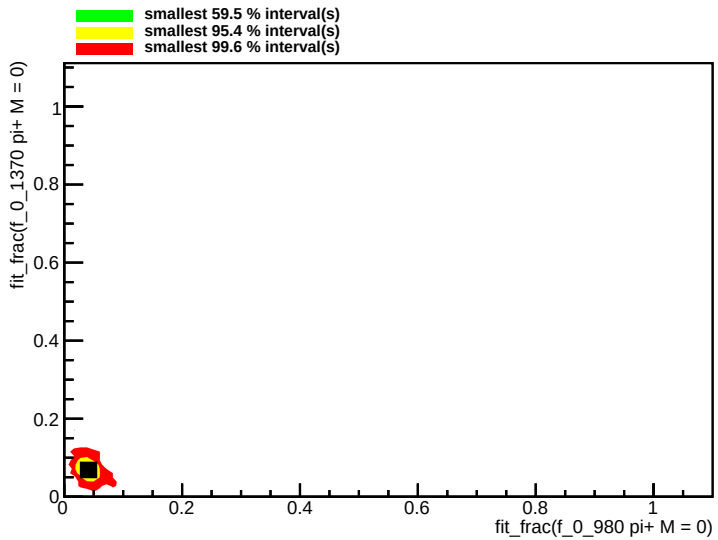


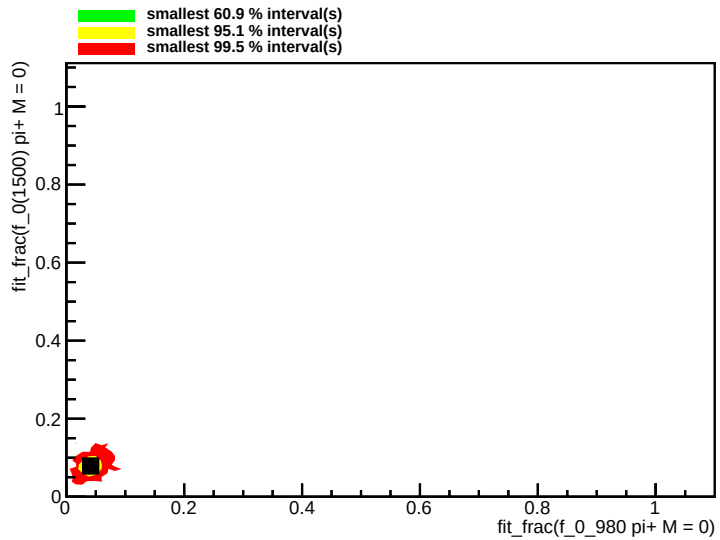
$D^+ \to \pi^+\pi^-\pi^+$ Posterior

Or look at correlations:









Summary

Sampling is important for both

- **generating simulated data** and
- **fitting a model to data**.

The most basic techniques are very ineffecient.

Replacing them is not difficult,
since **many ready-made tools for sampling exist**.

It is worth your time to think about
the optimal sampling method for your problem.

(If your problem is hairier than the examples I've given,
pay close attention to Fred's talk!)