Metropolis Algorithm in P0D NC1Pi0 Analysis

Shilin Liu

NNGroup Meeting

Jul. 8, 2020

Goal:

- Introduce a MCMC method (Metropolis Algorithm) from the perspective of my analysis.
- Will focus on the confusions I had when I learnt it
- It won't be fully rigorous for some contents, but the main intuitions will be provided.
- Hopefully everyone will have some ideal how and why MCMC works before the incoming seminar

Contents

- Why we use Metropolis Algorithm in the NC1Pi0 Analysis
- How Metropolis Algorithm works
- Example of Metropolis Algorithm sampling
- Why Metropolis Algorithm works
- How step size affects the sampling
- Adaptive MCMC

Bayes' Theorem and P0D FV Water Mass

What is the P0D FV water mass

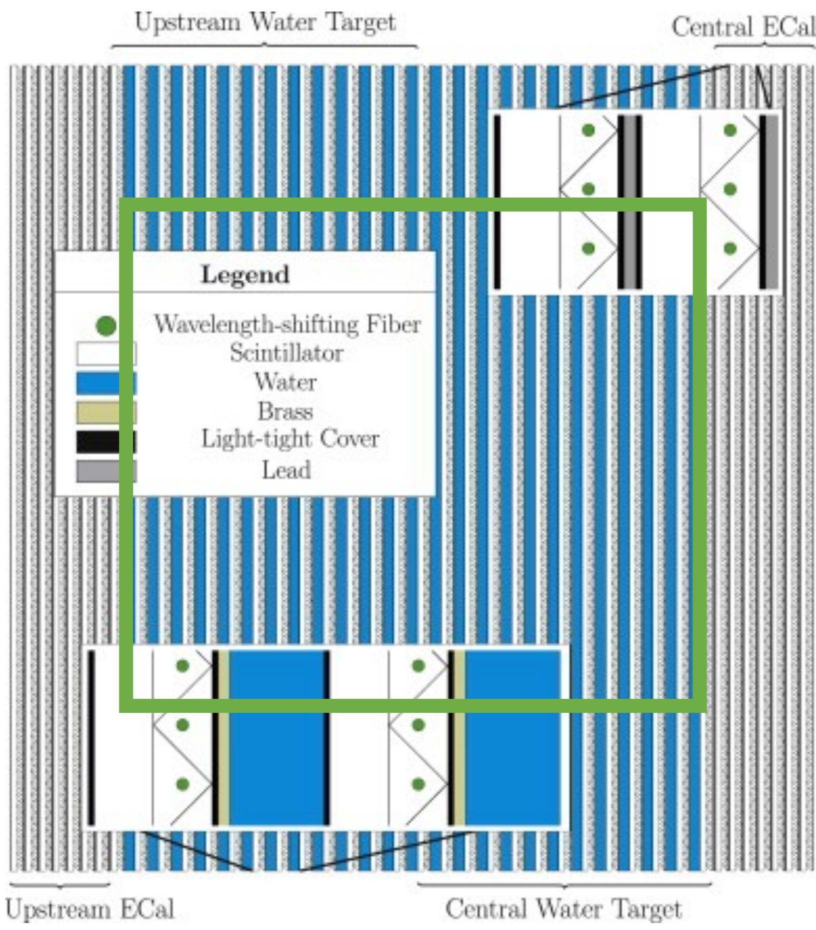
- Yue and I measured the fiducial volume (FV) water mass with scale to be $1910.4 \pm 10.8\ kg$

- In the NC1Pi0 analysis, we measure # of NC1Pi0 interactions.
- The data we observe (denoted by x) can further constrain the FV water mass (denoted by θ) by Bayes' Theorem:

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$$

$P(\theta)$: Prior distribution of θ, before seeing the data x, N(1910,10.8) in this case

$P(\theta|x)$: Posterior distribution of θ, in presence of data x. It is the information we have on θ after seeing the data



P0D Detector

Bayes' Theorem and P0D FV Water Mass

- The data we observe (denoted by x) can further constrain the FV water mass (denoted by θ) by Bayes' Theorem:

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$$

$P(x|\theta)$: Likelihood function, the conditional probability of x happening in presence of θ.

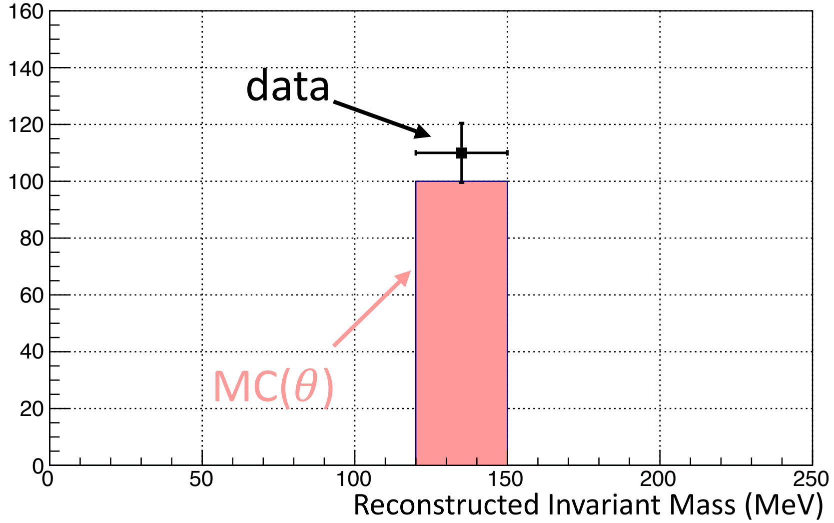
Treat data as an incident from a Poisson distribution with expected rate MC(θ):

$$P(x|\theta) = \frac{MC(\theta)^{data}e^{-MC(\theta)}}{data!}$$

P(x): constant, from law of total probability

$$P(x) = \int P(x|\theta)\,P(\theta)d\theta$$

Posterior distribution $P(\theta|x)$ is obtained!



- Observed data and monte carlo prediction MC as a function of θ
- Only 1 bin shown here as an example for likelihood

Bayes' Theorem and P0D FV Water Mass

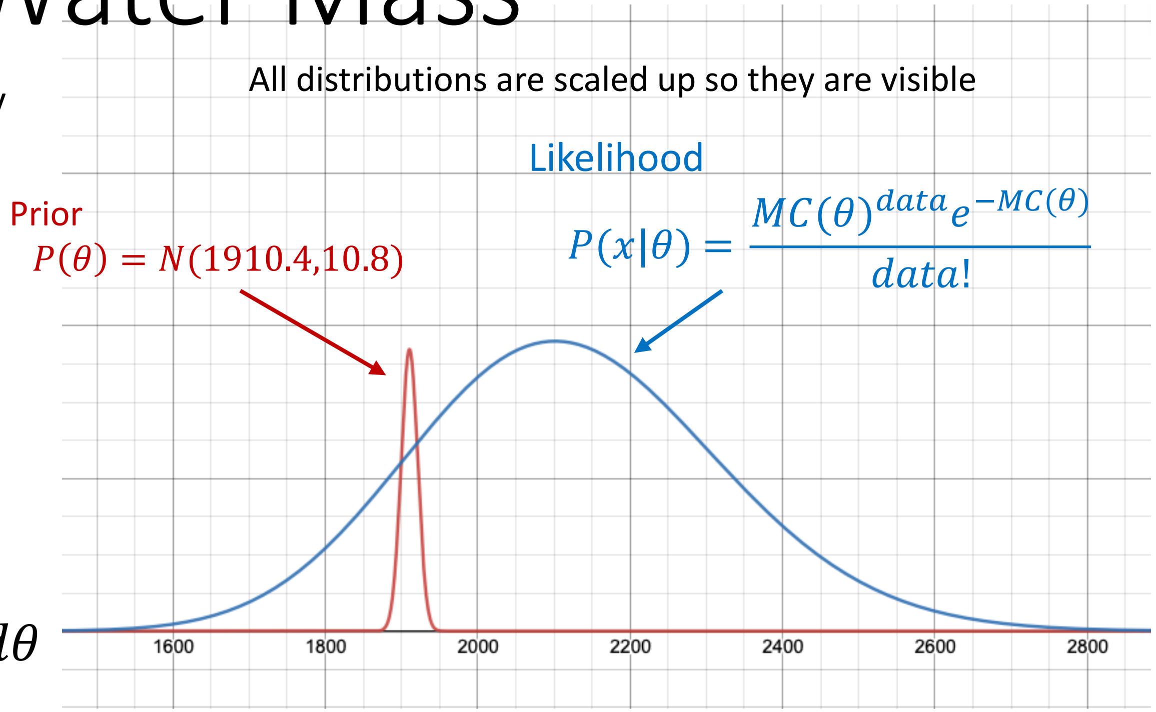
The data we observe (denoted by x) can further constrain the FV water mass (denoted by θ) by Bayes' Theorem:

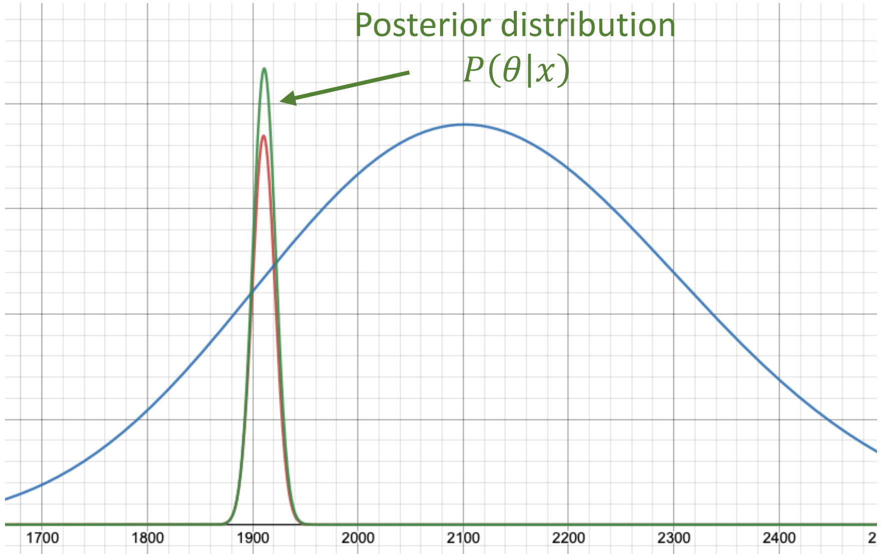
$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$$

Posterior distribution $P(\theta|x)$ is obtained!

What is the P0D FV water mass

Estimate the posterior distribution by $E[\theta] = \int \theta \cdot P(\theta|x) d\theta$





In real case, there are much more parameters (135 in my analysis)

- Cross-section of NC1Pi0 interaction
- Neutrino flux
- …

Monte carlo prediction is affected by $\vec{\theta} = (\theta_1, \ldots, \theta_{135})$.

The affect on MC($\vec{\theta}$) is correlated from all the parameters

Posterior distribution $P(\vec{\theta}|x)$ is a multi-dimensional distribution

Curse of High Dimensionality and Monte Carlo Integration

Suppose multi-dimensional posterior distribution $P(\vec{\theta}|x)$ is obtained

What is the P0D FV water mass (cross section of NC1Pi0)

Estimate the posterior distribution by

$$E[\theta_1] = \int \theta_1 \cdot P(\vec{\theta}|x) d\vec{\theta}$$

- There's no analytic form of $P(\vec{\theta}|x)$
- Numerically if only 2 points chosen for each parameter, 2^{135}

It is impossible to do this integration

Solution: Monte Carlo integration

- Sample from posterior distribution $P(\vec{\theta}|x)$ for a set of samples $(\vec{\theta}^1, \dots, \vec{\theta}^n)$
- Use sample mean $\bar{\theta}_1 = \frac{1}{n}\sum_{i=1}^{n} \theta_1^i$ as an approximation of $E[\theta_1]$
- Kolmogorov's Strong Law of Large Numbers applies and $\bar{\theta}_1$ converges almost surely to $E[\theta_1]$ as n becomes large
- The estimation of error of $\bar{\theta}_1$ is proportional to $\frac{1}{\sqrt{n}}$, regardless of the dimension

Example of Simple Monte Carlo

We need to sample from posterior distribution $P(\vec{\theta}|x)$ to estimate $E[\theta_1] = \int \theta_1 \cdot P(\vec{\theta}|x) d\vec{\theta}$ by sample mean $\bar{\theta}_1 = \frac{1}{n}\sum_{i=1}^{n} \theta_1^i$

1D example: $P(\theta) = \sqrt{1-\theta^2}$

Sample from a distribution:

- Generate samples from a process
- Putting samples into histogram
- Histogram converge to the distribution

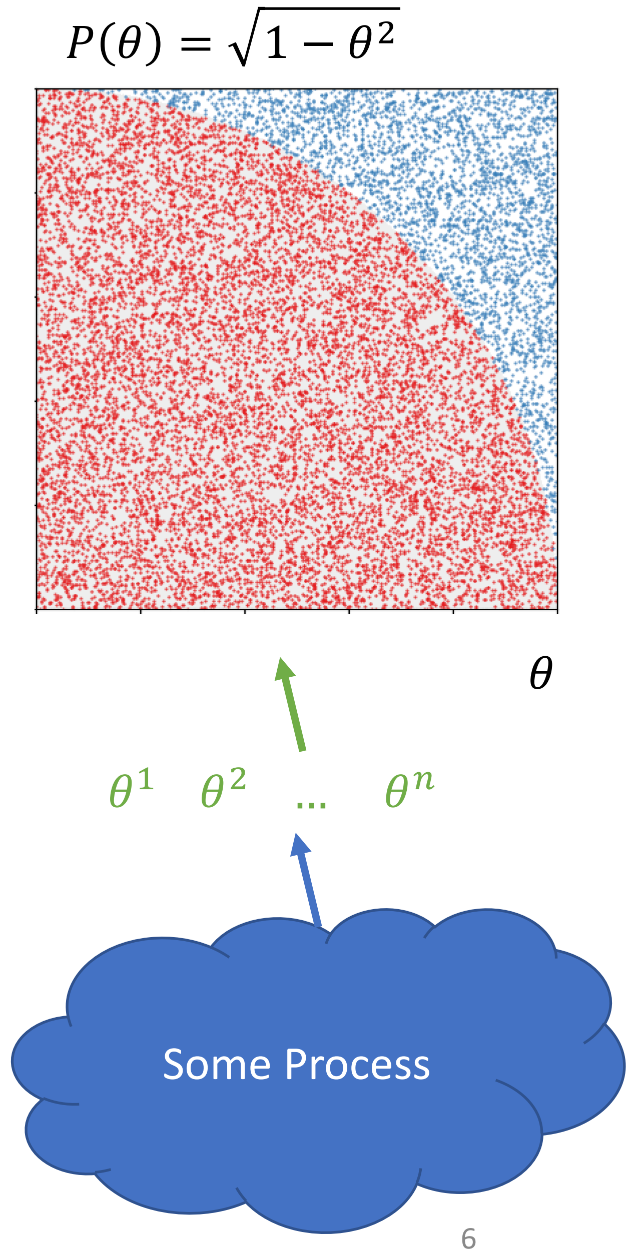
Rejection Sampling (low efficiency):

- Randomly generate samples in square
- Reject samples above the distribution

Inversion Sampling

Importance Sampling

…



Advantage of Metropolis Algorithm

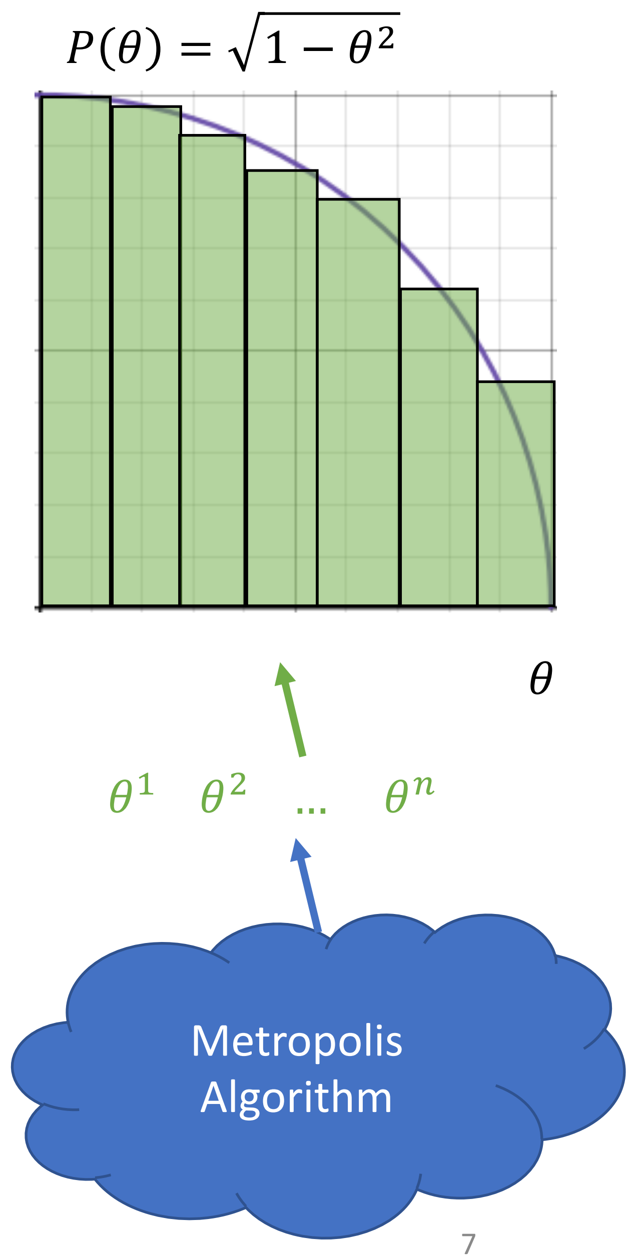
We need to sample from posterior distribution $P(\vec{\theta}|x)$ to estimate $E[\theta_1] = \int \theta_1 \cdot P(\vec{\theta}|x)d\vec{\theta}$ by sample mean $\bar{\theta}_1 = \frac{1}{n}\sum_{i=1}^{n}\theta_1^i$

$$P(\vec{\theta}|x) = \frac{P(x|\vec{\theta})P(\vec{\theta})}{P(x) = \int P(x|\vec{\theta})\,P(\vec{\theta})d\vec{\theta}}$$

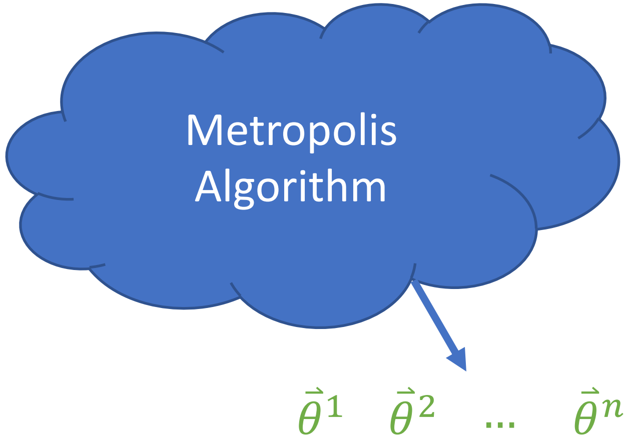
- P(x) is also a multi-dimensional integration thus unknow.
- We don't know the normalization constant of $P(\vec{\theta}|x)$.

- Previous sampling method mostly require full knowledge of target distribution
- They sometimes can be inefficient

- Metropolis Algorithm can sample from distribution without knowledge of the normalization constant



How Metropolis Algorithm Works

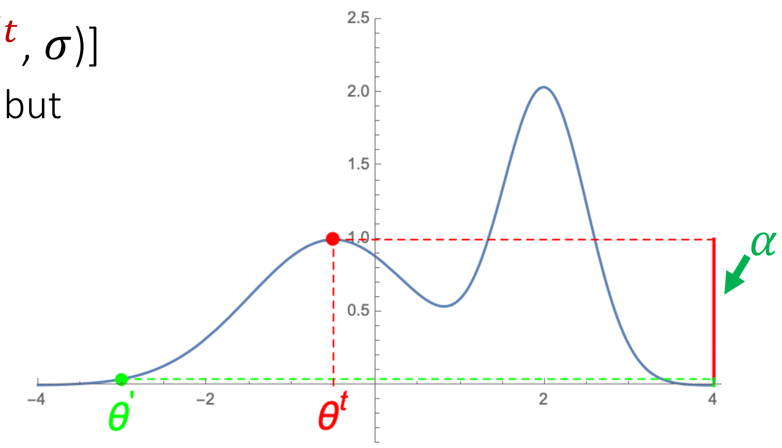


Metropolis Algorithm is a process devised to generate samples that will converge to a target distribution $P(\vec{\theta}|x)$ without knowing its normalization constant

$$P(\vec{\theta}|x) = \frac{P(x|\vec{\theta})P(\vec{\theta})}{P(x)}$$

- Choose a starting point $\vec{\theta}^0$ randomlly
- At step t+1, generate $\vec{\theta}^{t+1}$ by:
  1. Propose this step $\vec{\theta}'$ by random sampling from a distribution q($\vec{\theta}'$ | $\vec{\theta}^t$) [e.g.N($\vec{\theta}^t$, σ)]
     ($\vec{\theta}^t$ =-0.5, $\vec{\theta}'$ =2), q($\vec{\theta}'$ | $\vec{\theta}^t$) proposal distribution, doesn't have to be normal distribution, but has to be symmetric, q($\vec{\theta}'$ | $\vec{\theta}^t$) = q($\vec{\theta}^t$ | $\vec{\theta}'$)
  2. Calculate acceptance ratio $\alpha = \frac{P(\vec{\theta}'|x)}{P(\vec{\theta}^t|x)} = \frac{P(x|\vec{\theta}')P(\vec{\theta}')}{P(x|\vec{\theta}^t)P(\vec{\theta}^t)}$
     a. If α >1, accept. $\vec{\theta}^{t+1} = \vec{\theta}'$
     b. If α <1, generate random number r - Uniform[0,1]
        i. If r< α, accept. $\vec{\theta}^{t+1} = \vec{\theta}'$
        ii. If r> α , reject. $\vec{\theta}^{t+1} = \vec{\theta}^t$

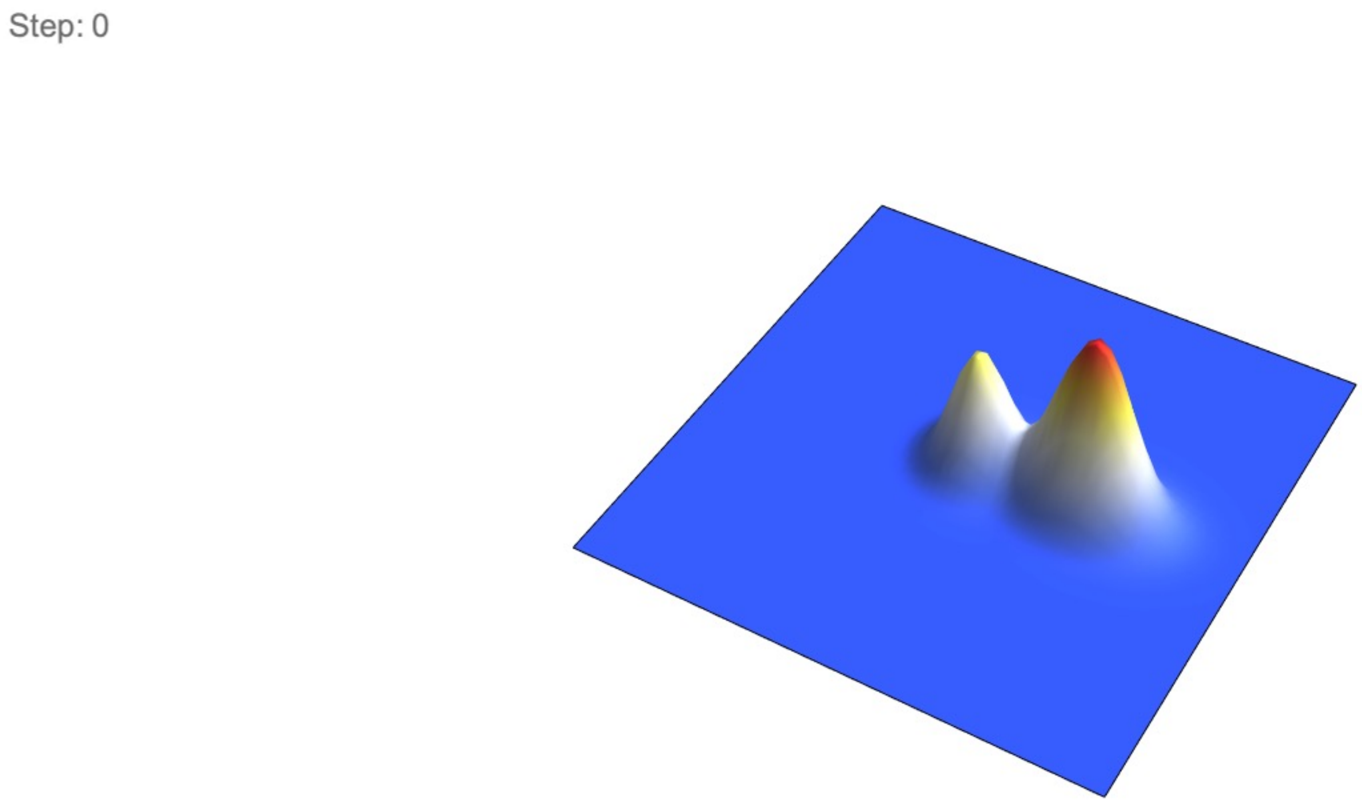
$P(x|\vec{\theta})P(\vec{\theta})$

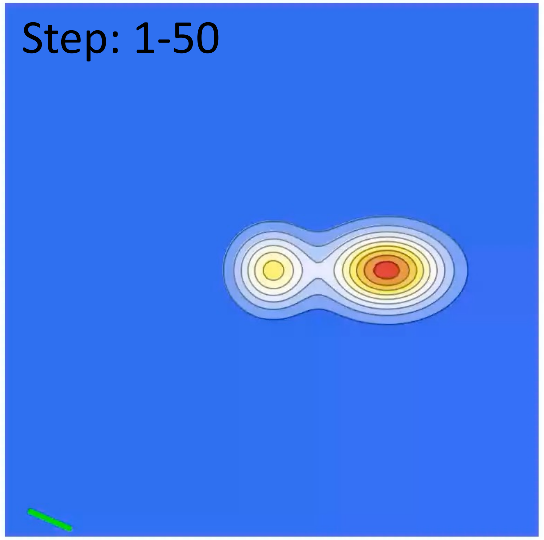


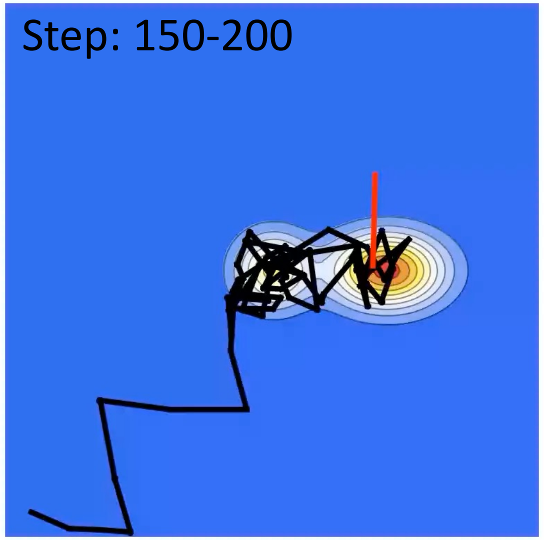
Example Sampling Steps

Rejected Steps
Accepted Steps
Previous Steps

- This is a random distribution to be sampled with Metropolis Algorithm
- Starting from (-10, -10), q($\vec{\theta}'$ | $\vec{\theta}^t$) taken as N($\vec{\theta}^t$,1.5)







Metropolis Algorithm Samples' Properties

Metropolis Algorithm is one of the most popular Markov Chain Monte Carlo (MCMC) algorithms

- The samples generated ($\vec{\theta}^1, \ldots, \vec{\theta}^n$) forms a Markov Chain, since $\vec{\theta}^{t+1}$ is and is only determined by $\vec{\theta}^t$
- The samples generated $\vec{\theta}^t$ and $\vec{\theta}^{t+m}$ are not independent, but they will become closer and closer to being independent as m increase. The correlation between them can be determined by a quantity "autocorrelation"
- It usually can be shown that the sample mean $\bar{\theta}_1 = \frac{1}{n}\sum_{i=1}^{n} \theta_1^i$ converges to the expected value $E[\theta_1] = \int \theta_1 \cdot P(\vec{\theta}|x)d\vec{\theta}$ with a Law of Large Numbers for dependent samples
- The samples generated ($\vec{\theta}^1, \ldots, \vec{\theta}^n$) will converge to the target distribution

What Causes the Samples to Converge to the Target Distribution

The samples generated ($\vec{\theta}^1, \ldots, \vec{\theta}^n$) will converge to the target distribution

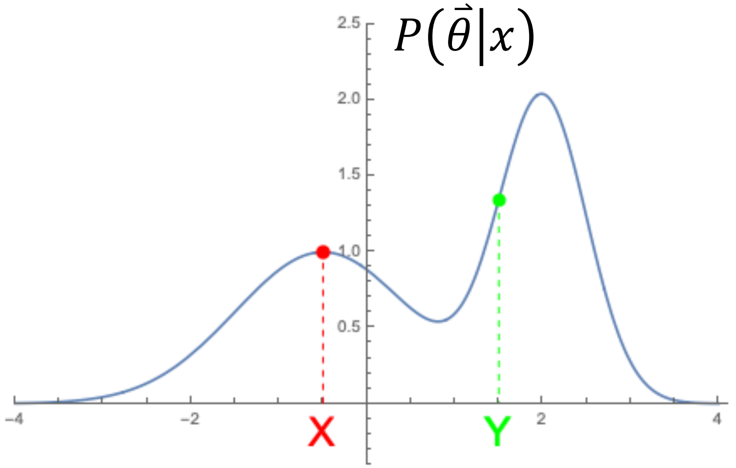
- The main intuitions will be provided below, it is not a rigorous proof

The crucial step is to prove that the target distribution is a stationary distribution of the Markov Chain

- Taking steps ($\vec{\theta}^t, \vec{\theta}^{t+2}, \vec{\theta}^{t+4} \ldots, \vec{\theta}^{t+2m}$), suppose they form the target distribution $P(\vec{\theta}|x)$
- The next steps ($\vec{\theta}^{t+1}, \vec{\theta}^{t+2+1}, \vec{\theta}^{t+4+1} \ldots, \vec{\theta}^{t+2m+1}$) will also form the target distribution
- (This is not rigorous, it is usually introduced directly in terms of applying Markov Chain transition kernel to a probability density distribution. But in the algorithm, the Markov Chain transition is from a sample step to another sample step, so in this way it is easier to explain)

What Causes the Samples to Converge to the Target Distribution

- Taking steps ($\vec{\theta}^{t}, \vec{\theta}^{t+2}, \vec{\theta}^{t+4} \ldots, \vec{\theta}^{t+2m}$), suppose they form the target distribution $P(\vec{\theta}|x)$
- The next steps ($\vec{\theta}^{t+1}, \vec{\theta}^{t+2+1}, \vec{\theta}^{t+4+1} \ldots, \vec{\theta}^{t+2m+1}$) is a transition of each of previous $\vec{\theta}^{t} = \vec{X}$ to another point $\vec{\theta}^{t+1} = \vec{Y}$ ($\vec{X}$ and $\vec{Y}$ here denotes random points in parameter space)



- Take 2 random point X, Y in $P(\vec{\theta}|x)$.
- Probability density of a transition from X to Y:
  $P(X \rightarrow Y) = P(X|x) * q(Y|X) * 1$ ($\alpha > 1$ so always accept)
- Probability density of a transition from Y to X:
  $$P(Y \rightarrow X) = P(Y|x) * q(X|Y) * \left(\alpha = \frac{P(X|x)}{P(Y|x)}\right) = P(X \rightarrow Y) \qquad {**}q(Y|X) = q(X|Y)$$
- There's no transition between X and Y. Same can be shown for all random points. This is called detailed balance. And thus $P(\vec{\theta}|x)$ is a stationary state.
- It can be shown that if $q(Y|X)$ can propose any point in parameter space with a positive probability density, the Markov Chain will converge to the stationary distribution.

Summary:

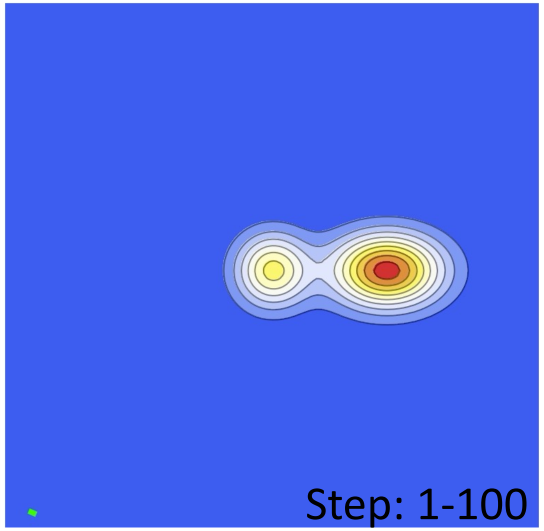
- Choice of $q(Y|X) = q(X|Y)$ and acceptance ratio α ensures the target distribution is the stationary distribution of Markov Chain
- Choice of proposal distribution q also ensures that the Markov Chain will converge to the stationary distribution
- Taking ratio of target distribution $\alpha = \frac{P(X|x)}{P(Y|x)}$ allows us to sample without normalization constant

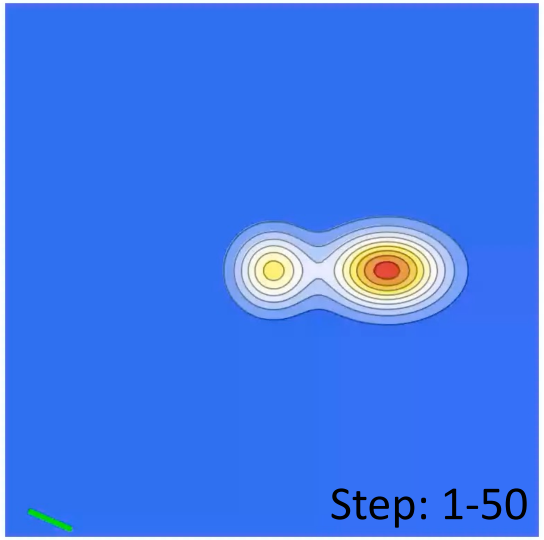
Why Step Size Matters

- This is a random distribution to be sampled with Metropolis Algorithm
- Starting from (-10, -10)

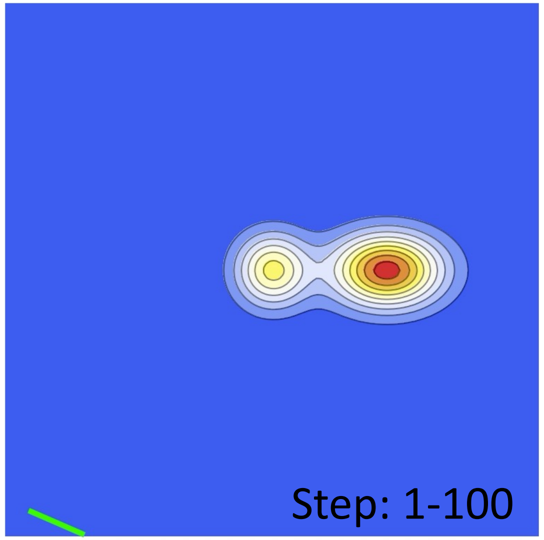
Rejected Steps
Accepted Steps
Previous Steps

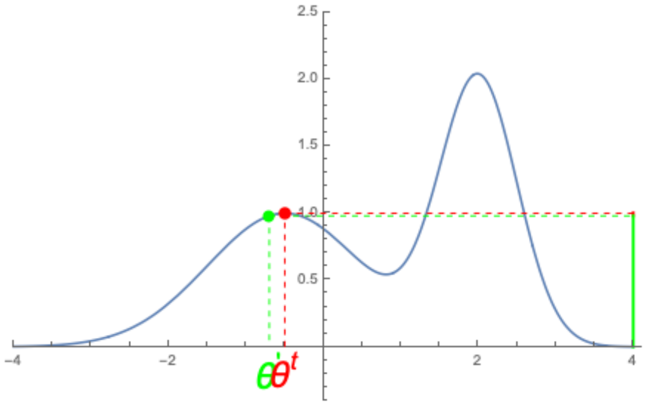
Step size too small

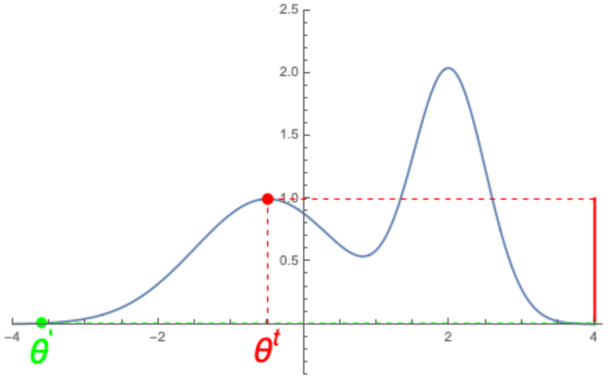




Step size too big







Posterior Predictive Distribution

$$P(\tilde{X}|X) = P(\tilde{X}|\vec{\theta})P(\vec{\theta}|X)$$

Posterior predictive distribution:

- Predictive new data $\tilde{X}$ given observed data X
- What we want

Posterior distribution:

- Sampled from MCMC

Predictive data distribution:

- Give model parameter $\vec{\theta}$
- $\tilde{X}$: predictive data

Example:

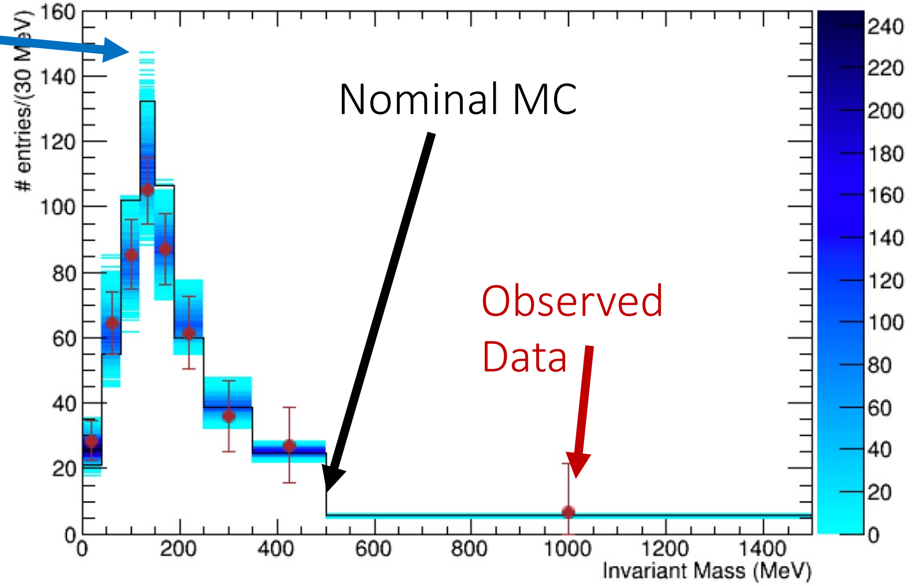
- $\tilde{X}_i$ ~ Poisson(MC_X($\vec{\theta}_\mathrm{i}$))

Take from MCMC output:

- $\vec{\theta}_\mathrm{i}$

Obtain predictive data:

- $\tilde{X}_i$



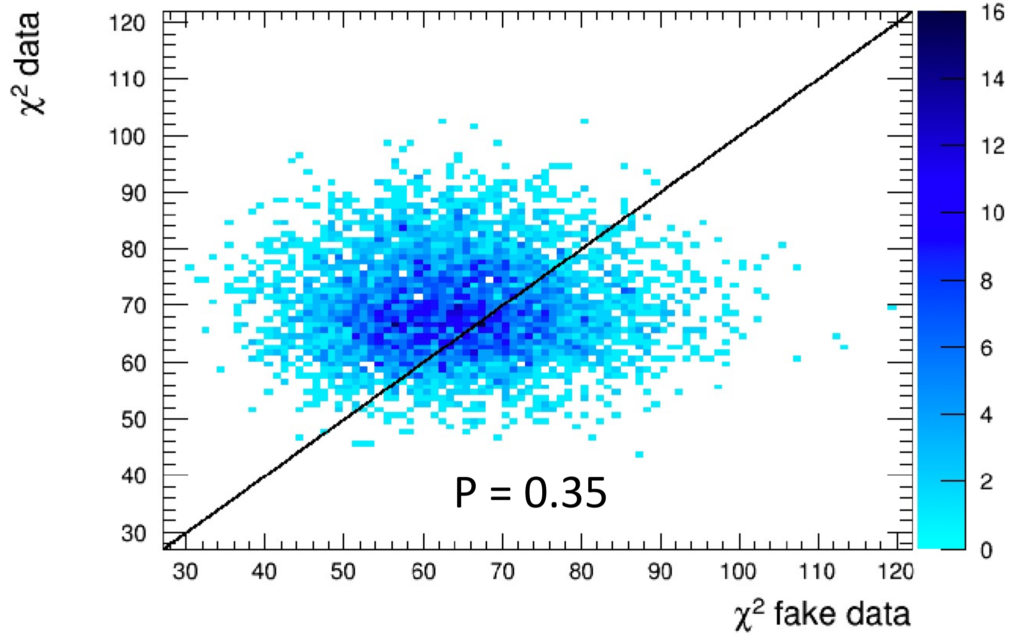
Posterior Predictive Checks and Bayesian P-value

Need to compare observed data X ~ posterior predictive distribution $(\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_n)$

- If ~ comparable: model fit ok
- Else: check model

Quantitatively:

- Calculate test statistics $T(\tilde{X})$ and T(X)
- Bayesian p-value $P = \Pr(T(\tilde{X}) > T(X))$
- For example: T can be Likelihood used in MCMC sampling



If Bayesian p-value is near 0 or 1 → This is bad, model misfit

- Observed data → extrema of fake simulated data

Note:

This method tells if a model misfit

This method doesn't support the model

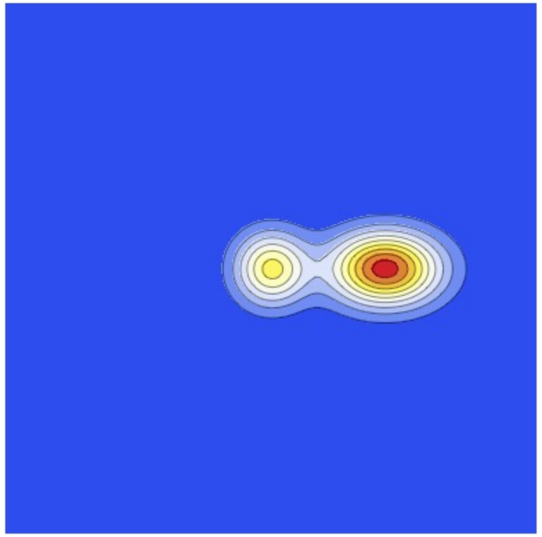
Summary

- Bayes’ Theorem can be used in extraction of xsec
- Posterior distribution is multi-dimensional and hard to integrate over, and the normalization constant is always unknow
- Use Metropolis Algorithm to sample from posterior distribution, use sample mean to approximate the expectation value of parameters
- Metropolis Algorithm is a process devised to generate samples that will converge to a target distribution without knowing the distribution’s normalization constant
    - Step size is important in sampling speed
- Posterior predictive checks can tell if model misfit

Backup

Adaptive Metropolis Algorithm (Clark's Code)

- Both Yue and I use Clark's Adaptive Metropolis Algorithm
- Auto tune step size so the overall acceptance rate of all sample is 44% for one parameter or 23.4% for five or more parameters
- It uses the covariance matrix of historical and accepted samples in the multivariate normal distribution to propose the next step.
- The proposal distribution becomes closer to the target distribution comparing to multivariate normal distribution without covariance, the proposal will be more efficient.



Reference

- Taboga, Marco (2017). "Metropolis-Hastings algorithm", Lectures on probability theory and mathematical statistics, Third edition. Kindle Direct Publishing. Online appendix. https://www.statlect.com/fundamentals-of-statistics/Metropolis-Hastings-algorithm.

- Ben Lambert. A Student's Guide to Bayesian Statistics

- Robert, C. P. and G. Casella (2013) Monte Carlo Statistical Methods, Springer Verlag.