

# Gravitation Wave Identification

Using Metropolis Hastings Algorithm

---

Group 1

Anirudh Bhat

Samyak Rai

Shanmukh Machiraju

# Index

1. Problem Statement
2. Methodology
3. Results
4. Optimization
5. Thank You

# Problem Statement

---

# Problem Statement

Given a time series strain data with added noise with the structure of a gravitational wave as given below

$$h(t) = \alpha e^t [1 - \tanh\{2(t - \beta)\}] \sin(\gamma t)$$

$\alpha, \beta, \gamma$  are parameters that signify the physical properties of the given wave.  
Their value ranges are

$$0 < \alpha < 2$$

$$1 < \beta < 10$$

$$1 < \gamma < 20$$

We need to determine the parameter values using a **Metropolis Hastings** Random walk algorithm in the 3 dimensional space.

# Understanding Wave Parameters

Let us visualize how the parameters  $\alpha$ ,  $\beta$ , and  $\gamma$  influence the waveform

1.  $\alpha$  controls the amplitude of the signal
2.  $\beta$  shifts the signal in time
3.  $\gamma$  controls the oscillation frequency

## **Animation Paramater effects**

# Methodology

---

# Random Walks

1. **Initialization:** We start with initial parameter values at the midpoints of the given ranges so

$$\alpha = 1, \beta = 5, \gamma = 10$$

2. **Random Walk:**

For each iteration we propose a new set of parameters using

$$\theta_{\text{new}} \sim N(\theta_{\text{initial}}, \sigma^2) \quad \text{where } \sigma = [0.005, 0.081, 0.2]$$

The new value is discarded or chosen based on an **Acceptance Probability** defined as

$$A(\theta_{\text{new}}, \theta_{\text{initial}}) = \min\left(1, \frac{\text{Posterior}(\theta_{\text{new}})}{\text{Posterior}(\theta_{\text{initial}})}\right)$$

The **Posterior** function is defined as the following

$$\text{Posterior}(\theta_i) = P(\theta_i \mid \text{data}) \propto P(\text{data} \mid \theta_i)P(\theta_i)$$

Due to the assumptions taken,  $P(\theta)$  has no effect on the acceptance ratio and our algorithm as a whole.

$$P(\theta) := \begin{cases} 1 & \text{if } \theta \in \theta_{\text{constraint}} \\ 0 & \text{everywhere else} \end{cases}$$

The only significant metric to consider is now the **Likelihood Function**  $L(\theta)$  which we define as

$$L(\theta) = \exp\left(-\frac{1}{2N} \sum_i \frac{(y_i - f(\theta_i))^2}{\sigma_i^2}\right)$$

# Results

---

# Numerical Analysis

## Parameter Values

Parameter	$\alpha$ (alpha)	$\beta$ (beta)	$\gamma$ (gamma)
Median Value	1.38	3.91	10.00
95% Credibility Interval	0.92 - 1.91	3.63 - 4.19	9.92 - 10.08
Effective Sample Size	25.0	61.1	1800.0
MC Standard Error	0.049	0.019	0.001

The MCMC Algorithm ran with **Acceptance Ratio** of **0.222**.

The Global **Signal to Noise Ratio** was **0.97**, with Local SNR of **0.99**

# Measurement Metrics for Metropolis-Hastings

- **Signal-to-Noise Ratio (SNR):**

Measures how strongly the true signal stands out from the noise. A Global SNR of 0.97 indicates a moderately clean signal, while a Local SNR of 0.99 shows that the oscillatory region is highly informative.

- **Effective Sample Size (ESS):**

MCMC samples are correlated, so the **true** number of independent samples is smaller. ESS quantifies this:

$$\text{ESS} = \frac{N_{\text{samples}}}{1 + 2 \sum_{k=1}^{\infty} \rho_k}$$

where  $\rho_k$  is the autocorrelation at lag  $k$ .

- Monte Carlo Standard Error (MCSE):

Estimates the uncertainty in the posterior mean due to sampling noise. It is defined as

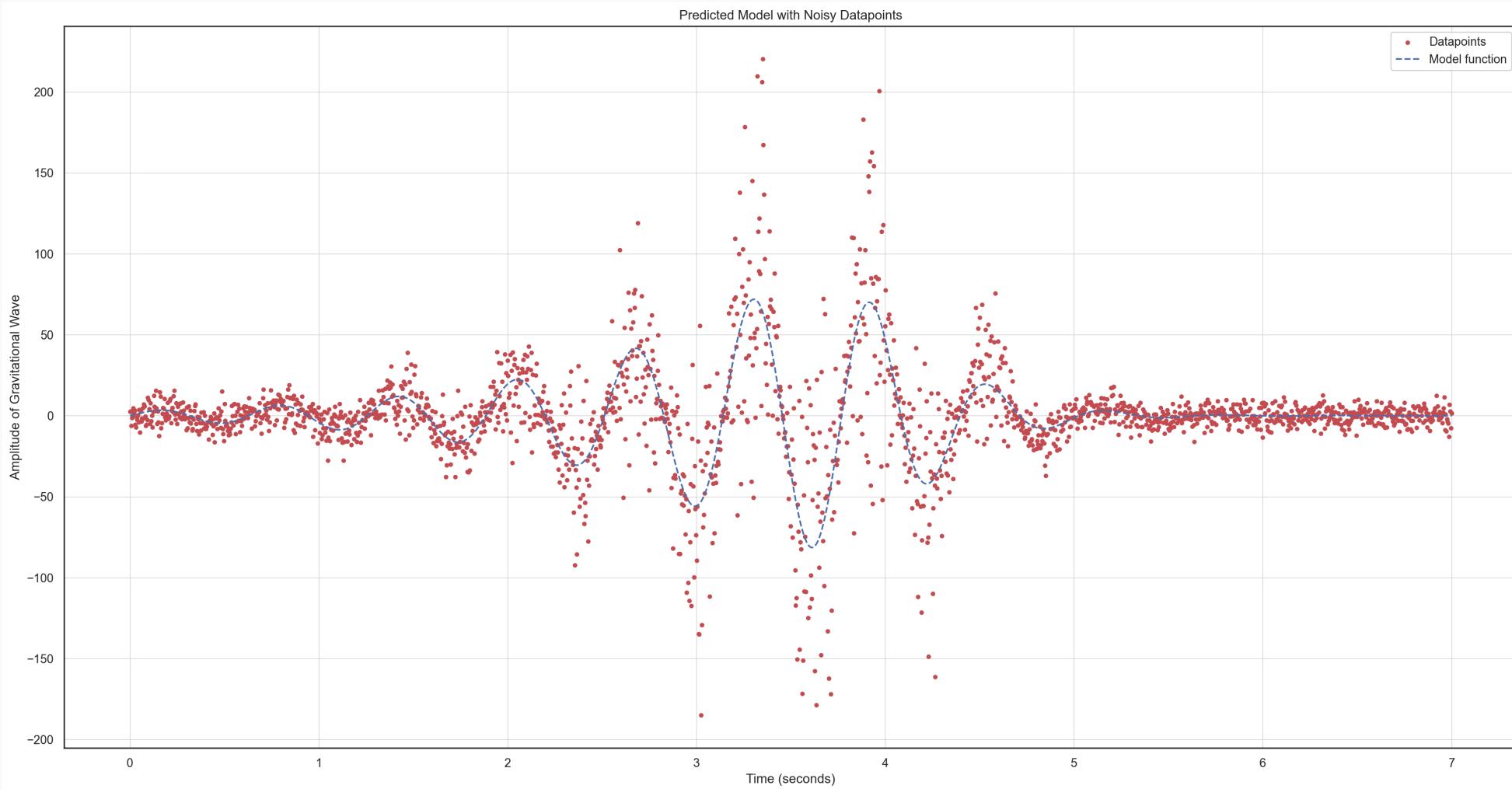
$$\text{MCSE} = \frac{\sigma}{\sqrt{\text{ESS}}}$$

Low MCSE indicates that the chain produced enough effective samples for reliable estimation of parameter statistics.

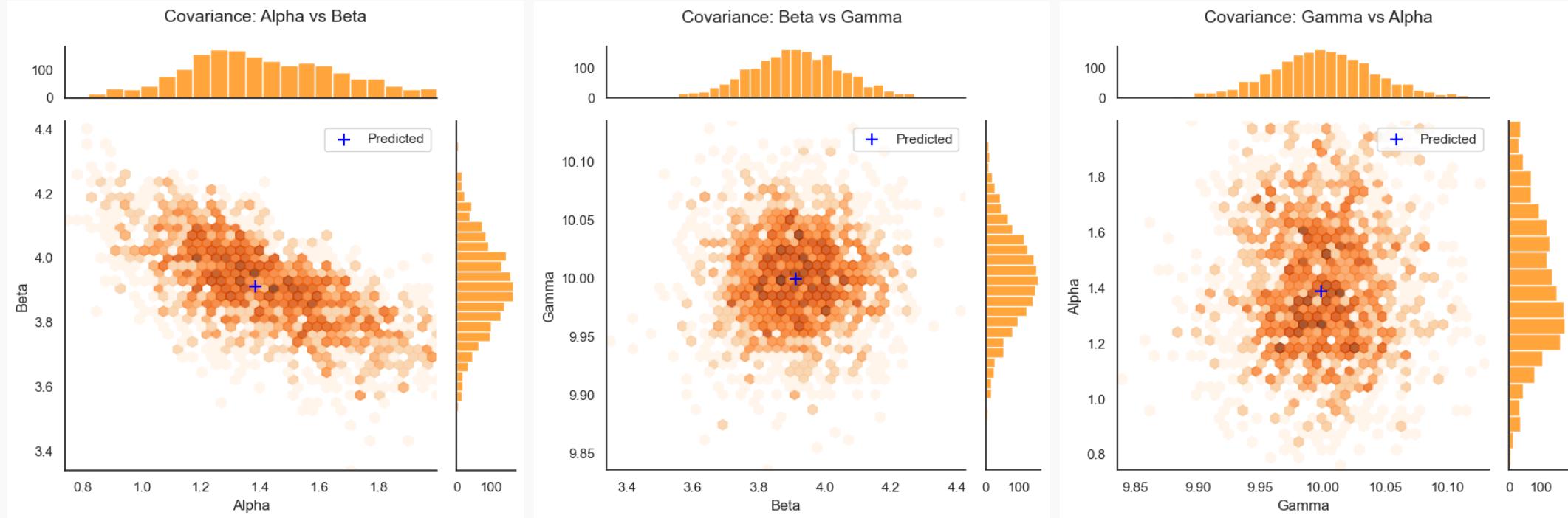
- Autocorrelation:

Measures how strongly each MCMC sample depends on earlier samples. High autocorrelation means slow exploration and fewer effectively independent samples, directly reducing ESS and increasing MCSE.

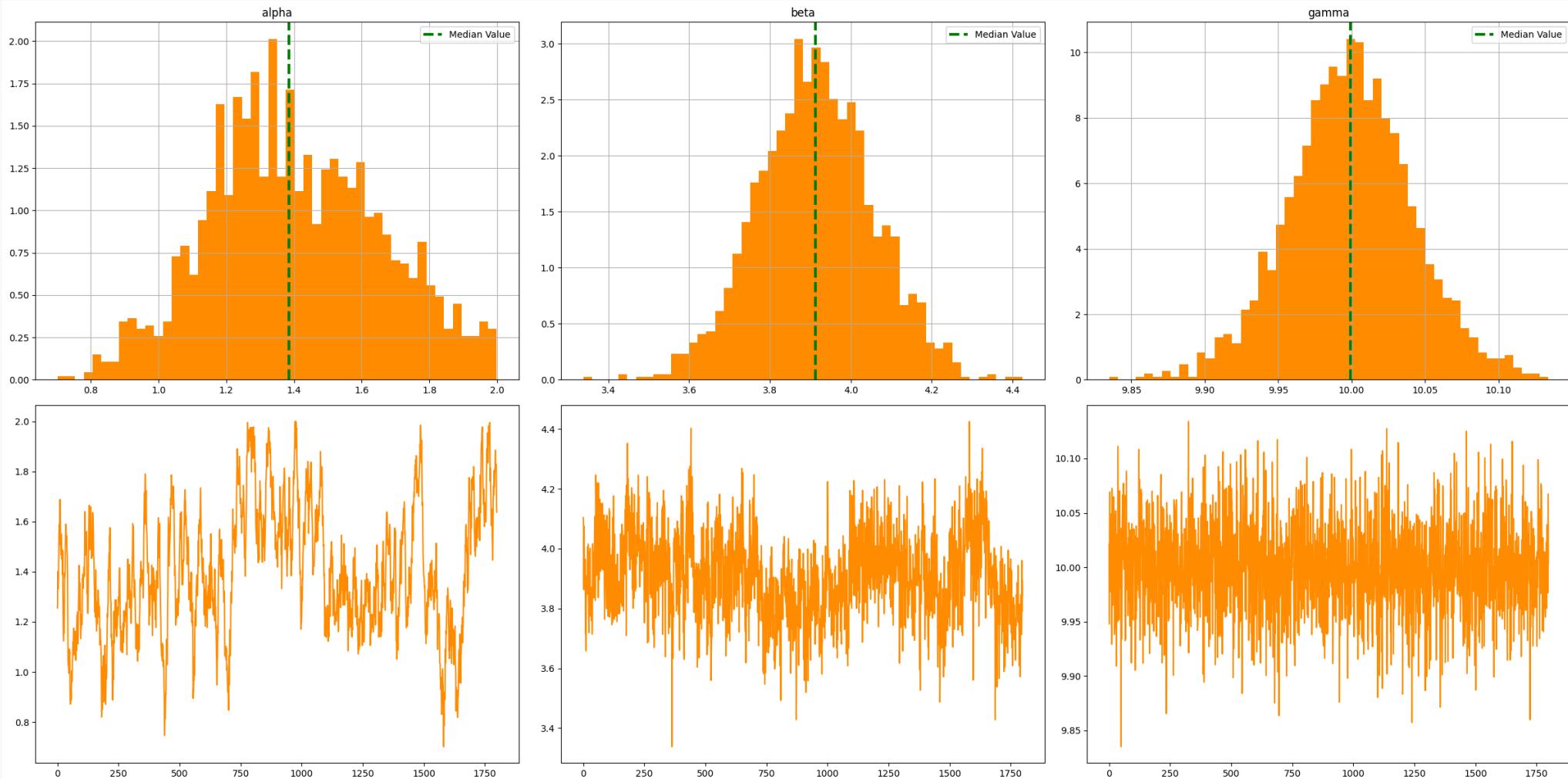
# Prediction vs Data



# Covariance Scatter Plots



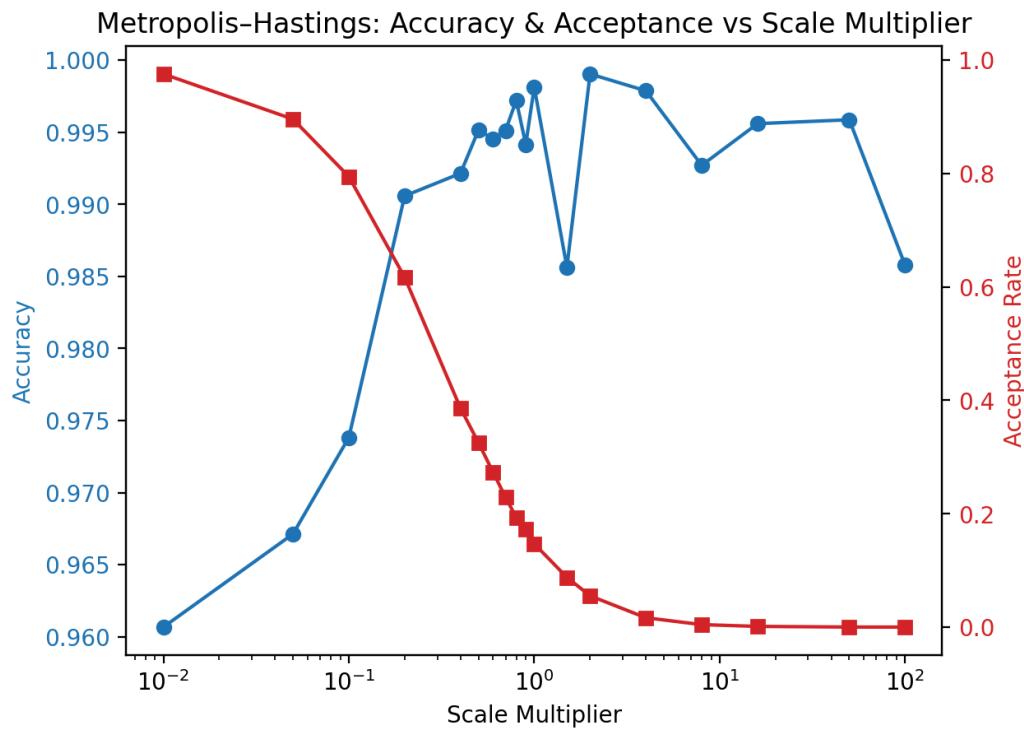
# Histograms and Trace Plots



# Optimization

---

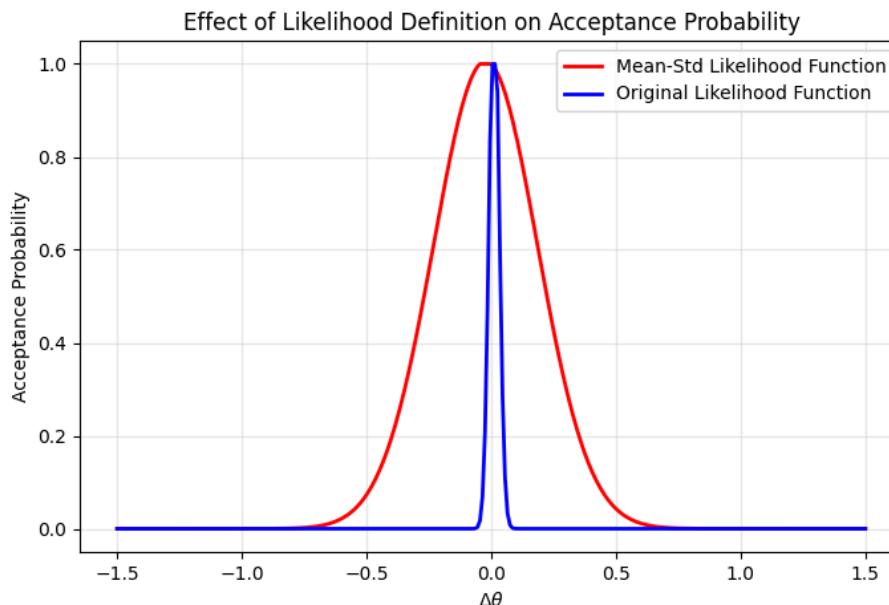
# Scale Selection



- **Small Scales ( $< 10^{-1}$ )**
  - Chain barely moves → strong autocorrelation → low accuracy.
- **Chosen Scales ( $\alpha: 0.005, \beta: 0.1, \gamma: 0.2$ )**
  - ([Roberts & Rosenthal, 2004](#)) predicts optimal acceptance  $\approx 0.234$  for high-dimensional targets.
  - Accuracy peaks – this is the optimal region for efficient sampling.
- **Large Scales ( $> 5$ )**
  - Acceptance rate falls → chain stagnates → accuracy degrades.

# Likelihood Function

-> The original likelihood makes even tiny parameter changes look catastrophically unlikely, driving acceptance to near zero. The new version fixes this by taking a mean instead of summation, keeping acceptance stable.



```
def likelihood_new(y_data, y_prior):  
    y_err = 0.1 * np.std(y_data)  
    Y = np.mean((y_data-y_prior)**2)/y_err**2  
    return -0.5 * Y  
  
def likelihood_original(y_datay, y_priory):  
    y_err = 0.1 * (y_data + y_prior) + 1e-6  
    Y = np.sum(((y_data - y_prior)/y_err) ** 2 )  
    return -0.5 * Y
```

# Data Generation for Better Inference

## System Overview

The data generation pipeline accepts an analytical gravitational wave model function and produces a CSV file containing noisy observations at discrete time points. This system serves two primary purposes:

- **Algorithm Validation:** Generate data with known ground truth parameters to verify MCMC recovery accuracy
- **Sensitivity Analysis:** Test algorithm performance under varying noise conditions and proposal scales
- **Robustness Check :** Validate algorithm stability under heteroscedastic noise to assess reliability across signal regimes.

## Noise Formula

The noise standard deviation for each data point is calculated as:

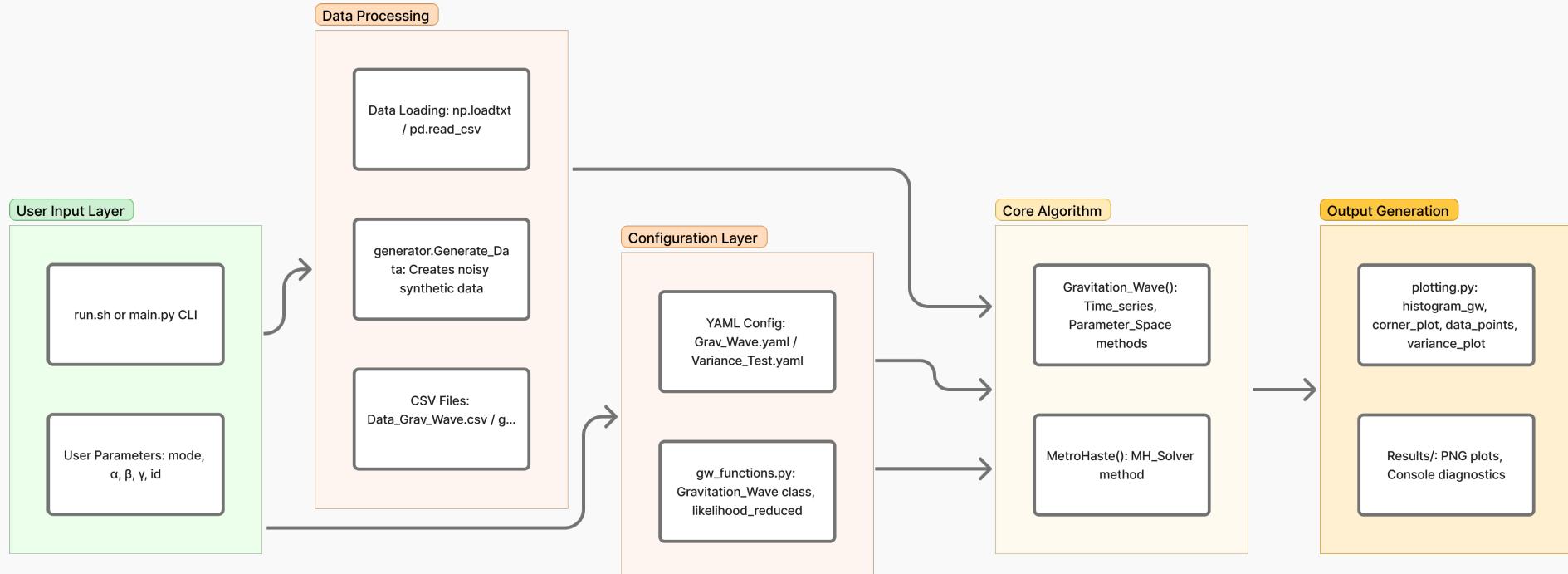
```
f_error = noise * f_points + 0.1 * std(f_points)  
f_noisy = f_points +  
          np.random.normal(0, np.abs(f_error), num)
```

This creates heteroscedastic noise where measurement error grows with signal amplitude.

## Noise Characteristics

Component	Purpose	Typical Magnitude
Proportional	Scales with signal amplitude	Dominant near peaks
Baseline	Constant floor noise	4–8 for typical signals

# Code Structure



# Thank You

---