Identifying Gravitational Wave Signatures

A Metropolis-Hastings MCMC Approach

Group no : 8 Sameer Choudhary Arpit Naman Verma

IIT Hyderabad

November 13, 2025

Outline

Introduction: What are Gravitational Waves?

Gravitational Waves (GW) are fundamental to our understanding of the universe.

- Predicted by Albert Einstein's Theory of General Relativity in 1916.

Introduction: What are Gravitational Waves?

Gravitational Waves (GW) are fundamental to our understanding of the universe.

- Predicted by Albert Einstein's Theory of General Relativity in 1916.
- They are "ripples" in the fabric of spacetime, traveling at the speed of light.

Introduction: What are Gravitational Waves?

Gravitational Waves (GW) are fundamental to our understanding of the universe.

- Predicted by Albert Einstein's Theory of General Relativity in 1916.
- They are "ripples" in the fabric of spacetime, traveling at the speed of light.
- Caused by the most cataclysmic events in the cosmos, such as:
  - Merging black holes
  - Colliding neutron stars
  - Supernovae

Introduction: What are Gravitational Waves?

Gravitational Waves (GW) are fundamental to our understanding of the universe.

- Predicted by Albert Einstein's Theory of General Relativity in 1916.
- They are "ripples" in the fabric of spacetime, traveling at the speed of light.
- Caused by the most cataclysmic events in the cosmos, such as:
  - Merging black holes
  - Colliding neutron stars
  - Supernovae
- First directly detected in 2015 by LIGO and Virgo, opening a new era of "multi-messenger astronomy."

The Project: Defining the Problem

We are given a mock time-series strain data file, ‘gw_data.csv‘.

- This file contains a simulated gravitational wave burst signal.

The Project: Defining the Problem

We are given a mock time-series strain data file, ‘gw_data.csv‘.

- This file contains a simulated gravitational wave burst signal.
- The "true" signal is hidden within a significant amount of background noise.

The Project: Defining the Problem

We are given a mock time-series strain data file, ‘gw_data.csv‘.

- This file contains a simulated gravitational wave burst signal.
- The "true" signal is hidden within a significant amount of background noise.
- **Our Goal:** To perform parameter estimation. We must extract the *true physical parameters* of the signal that are buried in the noise.

The Project: Defining the Problem

We are given a mock time-series strain data file, 'gw_data.csv'.

- This file contains a simulated gravitational wave burst signal.
- The "true" signal is hidden within a significant amount of background noise.
- **Our Goal:** To perform parameter estimation. We must extract the *true physical parameters* of the signal that are buried in the noise.
- **Our Tool:** We will use a Bayesian statistical method to find the most probable set of parameters that describe the data.

The Project: The Model Template

The analytical template for the signal, $h(t)$, is given by:

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

- This phenomenological equation models a burst signal:

The Project: The Model Template

The analytical template for the signal, $h(t)$, is given by:

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

- This phenomenological equation models a burst signal:
- $\sin(\gamma t)$: Represents the core oscillation of the wave.

The Project: The Model Template

The analytical template for the signal, $h(t)$, is given by:

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

- This phenomenological equation models a burst signal:
- $\sin(\gamma t)$: Represents the core oscillation of the wave.
- $\alpha \exp(t)$: Models the rapid, exponential growth in the signal's amplitude.

The Project: The Model Template

The analytical template for the signal, $h(t)$, is given by:

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

- This phenomenological equation models a burst signal:
- $\sin(\gamma t)$: Represents the core oscillation of the wave.
- $\alpha \exp(t)$: Models the rapid, exponential growth in the signal's amplitude.
- $\{1 - \tanh[...]\}$: Acts as a smooth "turn-off" switch, which rapidly cuts the signal off after $t \approx \beta$.

The Project: Defining the Parameters

Our goal is to find the posterior probability distribution for $\theta \equiv (\alpha, \beta, \gamma)$.

1. α (Alpha): Amplitude

- Describes the strength or "loudness" of the signal.
- Prior (our initial boundary): $0 < \alpha < 2$

The Project: Defining the Parameters

Our goal is to find the posterior probability distribution for $\theta \equiv (\alpha, \beta, \gamma)$.

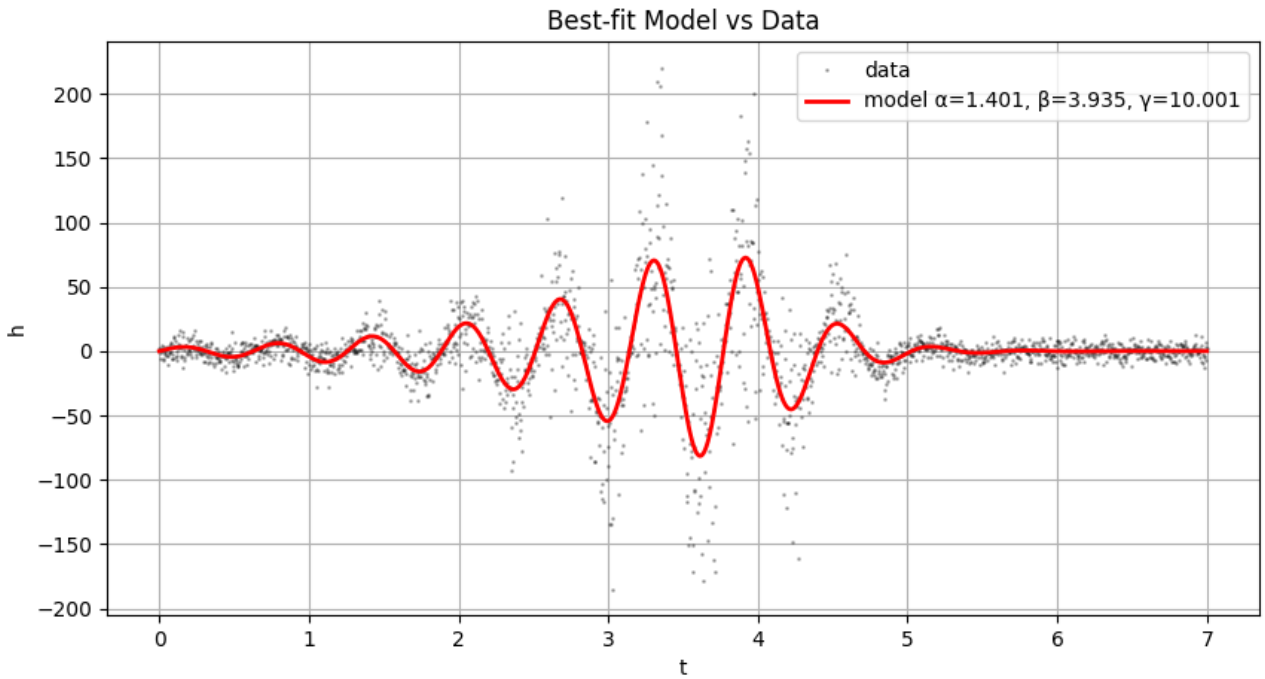
1. α **(Alpha): Amplitude**
   - Describes the strength or "loudness" of the signal.
   - Prior (our initial boundary): $0 < \alpha < 2$
2. β **(Beta): Turn-off Time**
   - Describes the time at which the signal burst ends.
   - Prior: $1 < \beta < 10$

The Project: Defining the Parameters

Our goal is to find the posterior probability distribution for $\theta \equiv (\alpha, \beta, \gamma)$.

1. α **(Alpha): Amplitude**
   - Describes the strength or "loudness" of the signal.
   - Prior (our initial boundary): $0 < \alpha < 2$
2. β **(Beta): Turn-off Time**
   - Describes the time at which the signal burst ends.
   - Prior: $1 < \beta < 10$
3. γ **(Gamma): Frequency**
   - Describes the frequency of the wave's oscillation.
   - Prior: $1 < \gamma < 20$

The Project: Best-Fit Model vs. Data



The Method: Bayesian Inference

We use Bayes' Theorem to find the probability of our parameters *given* the data.

$$\underbrace{P(\theta|\text{data})}_{\text{Posterior}} \propto \underbrace{P(\text{data}|\theta)}_{\text{Likelihood}} \times \underbrace{P(\theta)}_{\text{Prior}}$$

- **Posterior:** What we want. The probability of a parameter set (α, β, γ) after seeing the data.

The Method: Bayesian Inference

We use Bayes' Theorem to find the probability of our parameters *given* the data.

$$\underbrace{P(\theta|\text{data})}_{\text{Posterior}} \propto \underbrace{P(\text{data}|\theta)}_{\text{Likelihood}} \times \underbrace{P(\theta)}_{\text{Prior}}$$

- **Posterior:** What we want. The probability of a parameter set (α, β, γ) after seeing the data.
- **Likelihood:** The probability of seeing our data, *if* our parameters were the true values. This is our "goodness-of-fit" metric.

The Method: Bayesian Inference

We use Bayes' Theorem to find the probability of our parameters *given* the data.

$$\underbrace{P(\theta|\text{data})}_{\text{Posterior}} \propto \underbrace{P(\text{data}|\theta)}_{\text{Likelihood}} \times \underbrace{P(\theta)}_{\text{Prior}}$$

- **Posterior:** What we want. The probability of a parameter set (α, β, γ) after seeing the data.
- **Likelihood:** The probability of seeing our data, *if* our parameters were the true values. This is our "goodness-of-fit" metric.
- **Prior:** Our initial belief about the parameters (the ranges $0 < \alpha < 2$, etc.). This is a "box" that is 1 inside and 0 outside.

The Method: The Log-Likelihood

The project defines our Likelihood as $P(\text{data}|\theta) \propto \exp(\mathcal{Y})$.

Where $\mathcal{Y}$ is the (negative) Chi-Squared, a sum of the squared differences between the data and the model:

$$\mathcal{Y} = -\sum_i \frac{(y_{\text{data},i} - y_{\text{model},i})^2}{y_{\text{err},i}^2}$$

- We assume an error of 20% at each data point ($y_{\text{err},i} = 0.2 \times y_{\text{data},i}$).

The Method: The Log-Likelihood

The project defines our Likelihood as $P(\text{data}|\theta) \propto \exp(\mathcal{Y})$.

Where $\mathcal{Y}$ is the (negative) Chi-Squared, a sum of the squared differences between the data and the model:

$$\mathcal{Y} = -\sum_i \frac{(y_{\text{data},i} - y_{\text{model},i})^2}{y_{\text{err},i}^2}$$

- We assume an error of 20% at each data point ($y_{\text{err},i} = 0.2 \times y_{\text{data},i}$).
- In log-space, our scoring function becomes:
- $\log(\text{Posterior}) = \log(\text{Likelihood}) + \log(\text{Prior})$
- A "good fit" is one that minimizes the difference, making $\mathcal{Y}$ a large negative number, but as close to 0 as possible.

The Method: Metropolis-Hastings (MCMC)

How do we "search" the 3D parameter space for the best-fit?
We use a Markov Chain Monte Carlo, a "smart" random walk.

1. Start with an initial guess for (α, β, γ).

The Method: Metropolis-Hastings (MCMC)

How do we "search" the 3D parameter space for the best-fit?
We use a Markov Chain Monte Carlo, a "smart" random walk.

1. Start with an initial guess for (α, β, γ).
2. "Jiggle" the parameters randomly to propose a new, nearby step.

The Method: Metropolis-Hastings (MCMC)

How do we "search" the 3D parameter space for the best-fit?
We use a Markov Chain Monte Carlo, a "smart" random walk.

1. Start with an initial guess for (α, β, γ).
2. "Jiggle" the parameters randomly to propose a new, nearby step.
3. Calculate the *Posterior* "score" for this new step.

The Method: Metropolis-Hastings (MCMC)

How do we "search" the 3D parameter space for the best-fit?
We use a Markov Chain Monte Carlo, a "smart" random walk.

1. Start with an initial guess for (α, β, γ).
2. "Jiggle" the parameters randomly to propose a new, nearby step.
3. Calculate the *Posterior* "score" for this new step.
4. **The Metropolis Rule:**
   - If the new step has a **better** score, we **always accept it**.
   - If the new step has a **worse** score, we **sometimes accept it** (based on a random probability).

The Method: Metropolis-Hastings (MCMC)

How do we "search" the 3D parameter space for the best-fit?
We use a Markov Chain Monte Carlo, a "smart" random walk.

1. Start with an initial guess for (α, β, γ).
2. "Jiggle" the parameters randomly to propose a new, nearby step.
3. Calculate the *Posterior* "score" for this new step.
4. **The Metropolis Rule:**
   - If the new step has a **better** score, we **always accept it**.
   - If the new step has a **worse** score, we **sometimes accept it** (based on a random probability).
5. Repeat this process 80,000 times.

This creates a "chain" of steps that preferentially explores the most probable regions of the parameter space.

The Algorithm: Metropolis-Hastings MCMC

1. Initialize: Start the "walker" at an initial position $\theta_{\text{current}} = (\alpha_0, \beta_0, \gamma_0)$.

2. Score: Calculate the Log-Posterior "score" for this position:

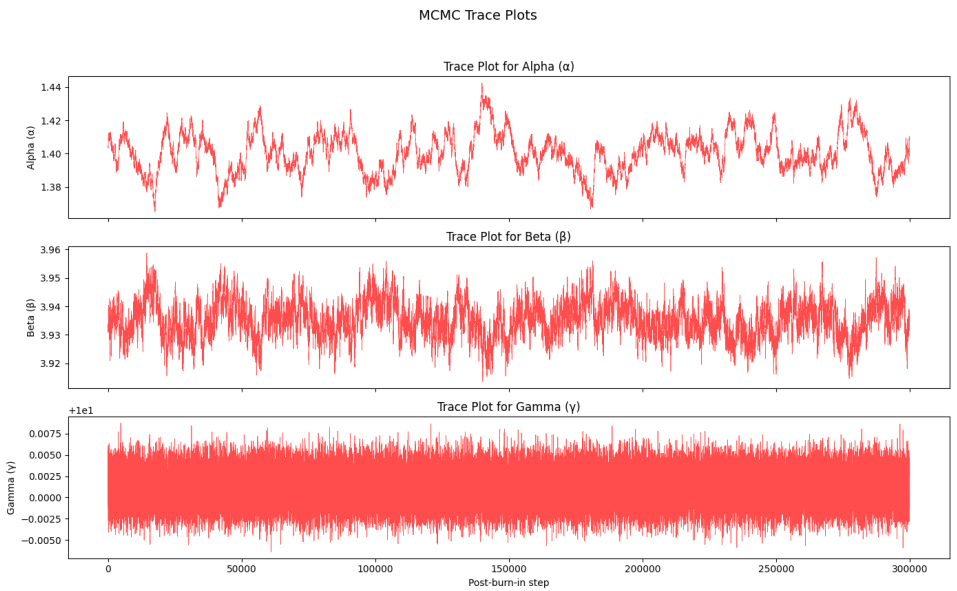
$$S_{\text{current}} = \ln(\text{Likelihood}(\theta_{\text{current}})) + \ln(\text{Prior}(\theta_{\text{current}}))$$

3. Iterate (Loop N times):
   - a. Propose: Generate a new position θ_{prop} by adding a small random number (from a Gaussian) to θ_{current}.
   - b. Score: Calculate the new score, S_{prop}, for the proposed position.
   - c. Accept/Reject (Metropolis Rule):
     - If $S_{\text{prop}} > S_{\text{current}}$ (the new spot is better), accept it.
     - If $S_{\text{prop}} \leq S_{\text{current}}$ (the new spot is worse),accept it with probability $A = \exp(S_{\text{prop}} - S_{\text{current}})$
   - d. Update: If accepted, set $\theta_{\text{current}} = \theta_{\text{prop}}$. If rejected, θ_{current} stays the same.
   - e. Record: Store the position θ_{current}.

4. Analyze: Discard the initial "burn-in" steps (e.g., first 100,000) and analyze the remaining chain.

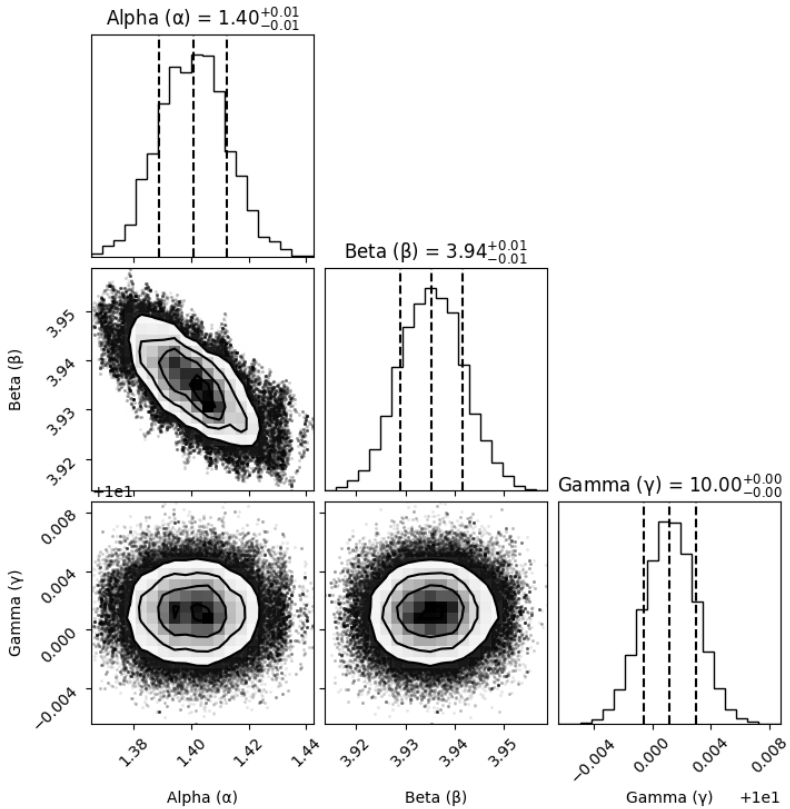
Results: MCMC Convergence (Trace Plots)

First, we check if the simulation converged. These plots show the value of each parameter at every step (after burn-in).



Results: Posterior Probability Distributions

These histograms show the final probability distribution for each parameter.



Results: Final Parameter Values

The peaks of the histograms (and the plot titles) show the most probable value and uncertainties.

Parameter	**Recovered Value (from plot)**
α (Amplitude)	1.40 ± 0.01
β (Turn-off)	3.94 ± 0.01
γ (Frequency)	10.000 ± 0.002

Results: Final Parameter Values

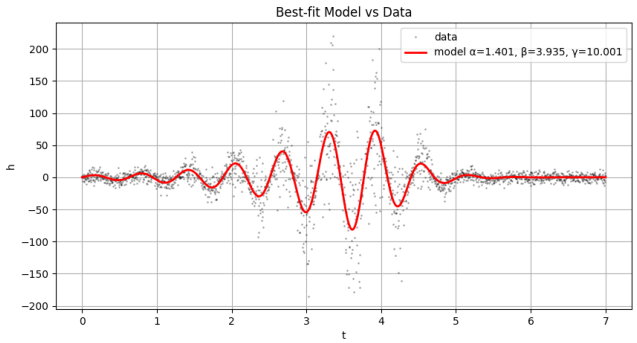
The peaks of the histograms (and the plot titles) show the most probable value and uncertainties.

Parameter	**Recovered Value (from plot)**
α (Amplitude)	1.40 ± 0.01
β (Turn-off)	3.94 ± 0.01
γ (Frequency)	10.000 ± 0.002

- The narrow, sharp peaks for all three parameters indicate a high degree of confidence in these recovered values.

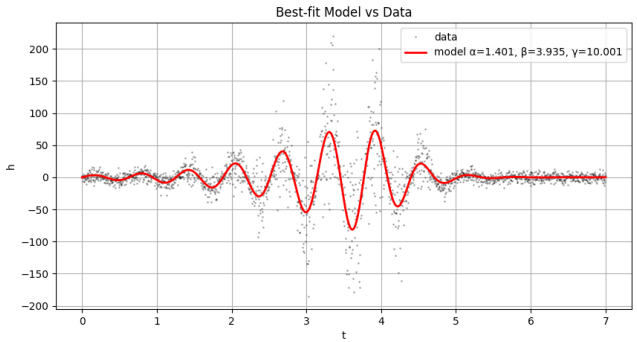
Results: Best-Fit Model vs. Data

We plot the model using the recovered parameters against the original noisy data.



Results: Best-Fit Model vs. Data

We plot the model using the recovered parameters against the original noisy data.



Analysis:

- The red line (model) passes perfectly through the dense regions of the grey data points (noise).
- The recovered parameters (legend values) are:
- $\alpha \approx 1.401, \beta \approx 3.935, \gamma \approx 10.001$
- This visually confirms a high-quality fit.

Conclusion

- We successfully implemented a Metropolis-Hastings MCMC algorithm to analyze a mock gravitational wave signal.

Conclusion

- We successfully implemented a Metropolis-Hastings MCMC algorithm to analyze a mock gravitational wave signal.
- The provided analytical template $h(t)$ proved to be an excellent model for the underlying data.

Conclusion

- We successfully implemented a Metropolis-Hastings MCMC algorithm to analyze a mock gravitational wave signal.
- The provided analytical template $h(t)$ proved to be an excellent model for the underlying data.
- Our algorithm successfully converged and recovered the hidden parameters (α, β, γ) from the noisy time-series data with high confidence.

Conclusion

- We successfully implemented a Metropolis-Hastings MCMC algorithm to analyze a mock gravitational wave signal.
- The provided analytical template $h(t)$ proved to be an excellent model for the underlying data.
- Our algorithm successfully converged and recovered the hidden parameters (α, β, γ) from the noisy time-series data with high confidence.
- This project demonstrates a core technique used in real gravitational wave data analysis for parameter estimation.

Questions?

Thank you.