Gamma-Ray Burst Parameter Estimation Using VegasAfterglow and MCMC Methods

Cheremnov Roman
Zhienbayev Adil

Research Proposal
Under the direction of

Dr. Ernazar Abdikamalov
Nazarbayev University

8 August 2025

Abstract

Gamma-Ray Bursts (GRBs) and their multi-wavelength afterglows offer critical insights into extreme astrophysical phenomena. The VegasAfterglow framework provides a powerful tool for modeling these afterglows and inferring physical parameters using Markov Chain Monte Carlo (MCMC) algorithms. Before applying such sophisticated techniques to real observations, rigorous validation is essential. This proposal outlines a systematic study to assess the reliability and accuracy of VegasAfterglow's MCMC parameter estimation pipeline. We will generate synthetic GRB afterglow light curves using a TophatJet model with precisely known input parameters and realistic observational noise. The MCMC algorithm will then be applied to these synthetic datasets to recover the parameters. Our methodology involves a direct comparison of the MCMC-derived posterior distributions with the true input values to quantify the recovery's consistency and precision. This validation forms a crucial foundation for future research, enabling the robust application of VegasAfterglow to real GRB observations and facilitating systematic population studies to uncover new astrophysical insights.

1 Introduction

Gamma-Ray Bursts (GRBs) are short but intensely luminous flashes of gamma-rays from distant galaxies, typically caused by the core collapse of massive stars or mergers of neutron star binaries. These bursts are followed by multi-wavelength afterglow emission, which occurs when relativistic jets decelerate as they interact with the surrounding medium [1]. The afterglow radiates energy primarily through synchrotron radiation and, when observed and analyzed across multiple wavelengths, can provide crucial information about jet structure, energy, and the circumburst environment [2].

VegasAfterglow is a high-performance modeling framework that generates synthetic afterglow data from initial conditions and observer viewing angles [3]. The framework includes functionality to determine the initial characteristics of bursts from observational data using **Markov Chain Monte Carlo (MCMC)** algorithms, which allow extraction of initial burst conditions from light curve data at single or multiple frequencies.

However, before applying such sophisticated parameter estimation techniques to real GRB observations, it is essential to validate the reliability and accuracy of the recovery process. In this work, we present a systematic validation study of MCMC parameter estimation for GRB afterglows using VegasAfterglow. Our primary objective is to assess how well the MCMC pipeline can recover known input parameters from synthetic afterglow data under realistic observational conditions. This validation forms the foundation for a broader research aimed at robust parameter estimation for real GRB observations.

2 Methodology

This section outlines the systematic approach undertaken to validate the Markov Chain Monte Carlo (MCMC) parameter estimation capabilities within the **VegasAfterglow** framework. The methodology involved three primary phases: the **generation of synthetic Gamma-Ray Burst (GRB) afterglow data**, the **application of MCMC algorithms for parameter recovery**, and a **rigorous comparison** of the recovered parameters against the known input values.

1. Synthetic Data Generation

The initial phase involved creating **synthetic GRB afterglow light curves** using the `VegasAfterglow` high-performance modeling framework [3]. For this study, a specific **TophatJet model** was employed to represent the relativistic jet structure. A set of **known initial physical parameters** for a hypothetical GRB afterglow was precisely defined. These parameters, consistent with the simulation code, included:

- **Jet Opening Angle** (θ_c): Set to 0.1 radians.
- **Isotropic Equivalent Energy** (E_{iso}): Set to 10^{51} ergs.
- **Initial Lorentz Factor** (Γ_0): Set to 300.
- **Circumburst Medium Density** (n_{ism}): Set to 0.1 particles/cm^3.

- **Radiation Parameters**: Electron energy fraction ($\epsilon_e = 10^{-1}$), magnetic field energy fraction ($\epsilon_B = 10^{-3}$), electron power-law index ($p = 2.3$), and electron injection efficiency ($\xi_e = 1$).

These predefined parameters served as the "ground truth" for our validation. The `VegasAfterglow` code was then utilized to simulate the expected multi-wavelength light curve emission based on these inputs over a specified time grid (from 10^4 to 10^6 seconds, with 20 points) at a single observation band (4.84×10^{14} Hz). To mimic realistic observational conditions, a **relative uniform random noise** ranging from -10% to +10% of the calculated flux was added to each synthetic data point. The generated synthetic data, comprising time, observed flux, and associated error, was then saved to a CSV file for subsequent analysis.

2. MCMC Parameter Recovery

Following the generation of synthetic data, the next critical step was to apply `VegasAfterglow`'s integrated **MCMC parameter estimation algorithm** to these simulated light curves. The objective here was to determine how effectively the algorithm could recover the original, known input parameters without prior knowledge of their values. The synthetic light curve data was fed into the MCMC sampler, which then explored the multi-dimensional parameter space. The MCMC algorithm generates **posterior probability distributions** for each parameter, providing not just a single best-fit value, but a range of probable values and their associated uncertainties. This process allowed us to simulate the real-world scenario where only observational data is available, and the underlying physical parameters are unknown.

3. Validation and Comparison

The final phase of the methodology involved a direct **comparison of the MCMC-recovered parameters with the initially defined true parameters**. For each parameter, the peak and credible intervals of the MCMC-derived posterior distributions were assessed against the corresponding input values used during synthetic data generation. The primary metric for validation was the **consistency** between the known input parameters and the central tendency and spread of their respective posterior distributions. This systematic comparison allowed us to quantify the **accuracy and reliability** of the `VegasAfterglow` MCMC pipeline in recovering physical properties under controlled conditions.

3 Future Work and Conclusion

This validation study serves as a crucial foundational step, demonstrating the reliability of the `VegasAfterglow` MCMC parameter estimation pipeline under controlled, synthetic conditions. The successful recovery of known input parameters from simulated afterglow light curves provides confidence in the framework's ability to accurately infer physical properties.

Building upon this validated methodology, our future research will focus on several key areas:

- **Application to Real Observational Data**: The immediate next step involves applying the validated MCMC pipeline to analyze real GRB afterglow observational data. This will allow us to extract physical parameters from actual cosmic events and contribute to the understanding of GRB diversity and evolution.

- **Exploration of Complex Jet Models**: Beyond the TophatJet model, future work will extend the validation to more complex jet geometries (e.g., structured jets, Gaussian jets) and their impact on parameter recovery.

- **Population Studies and New Physics**: Ultimately, this work forms the basis for systematic population studies of GRBs, which could reveal correlations between physical parameters and potentially uncover new insights into the underlying physics of these extreme astrophysical phenomena.

This systematic validation ensures that future scientific inferences drawn using `VegasAfterglow` are robust and reliable, paving the way for significant contributions to GRB afterglow research.

References

[1] H. Van Eerten. Gamma-ray burst afterglow theory. *arXiv.org*, 2013. URL `https://arxiv.org/abs/1309.3869`.

[2] D. Miceli and L. Nava. Gamma-ray bursts afterglow physics and the vhe domain. *Galaxies*, 10(3):p. 66, 2022. doi:10.3390/galaxies10030066.

[3] Y. Wang and et al. Vegasafterglow: A high-performance framework for gamma-ray burst afterglows. *arXiv.org*, 2025. URL `https://arxiv.org/abs/2507.10829`.

A Data generation code with multiple lines on a graph

```
import matplotlib.pyplot as plt
import numpy as np
from VegasAfterglow import ISM, TophatJet, Observer, Radiation, Model
import os
import matplotlib.cm as cm
import matplotlib.colors as colors

angles = [0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3]
energies = [1e50, 5 * 1e50, 1e51, 5 * 1e51, 1e52, 5 * 1e52, 1e53]
gammas = [10, 50, 100, 150, 300, 500, 750, 1000]
freq = 1e14


def plot_energy(save=False):
    medium = ISM(n_ism=0.1)
    obs = Observer(lumi_dist=1e26, z=0.1, theta_obs=0)
    rad = Radiation(eps_e=1e-1, eps_B=1e-3, p=2.3, xi_e=1)

    theta_c = 0.1
    gamma0 = 200

    fig, ax = plt.subplots(figsize=(10, 6))

    cmap = cm.viridis
    norm = colors.LogNorm(vmin=min(energies), vmax=max(energies))

    lines = []

    ax.set_title(f"Different energies(freq={freq:.0e})")
    ax.set_xlabel("Time(s)")
    ax.set_ylabel("Flux")

    for i, energy in enumerate(energies):
        jet = TophatJet(theta_c=theta_c, E_iso=energy, Gamma0=gamma0, duration=10)
        model = Model(jet=jet, medium=medium, observer=obs, forward_rad=rad, resolutions=(0.3,3,5))
        times = np.logspace(2, 8, 200)
        bands = np.array([freq])
        results = model.specific_flux(times, bands)

        color = cmap(norm(energy))


        line = ax.loglog(times, results['syn'][0,:], color=color, linewidth=2)
        lines.extend(line)


    sm = plt.cm.ScalarMappable(cmap=cmap, norm=norm)
    sm.set_array([])
    cbar = plt.colorbar(sm, ax=ax, shrink=0.8, pad=0.02)
    cbar.set_label('Energy (erg)', rotation=270, labelpad=20)

    cbar_ticks = energies
```

```
    cbar.set_ticks(cbar_ticks)
    cbar.set_ticklabels([f'{e:.0e}' for e in cbar_ticks])

    ax.grid(True, alpha=0.3)
    plt.tight_layout()
    if save:
        output_dir = "new_plots"
        os.makedirs(output_dir, exist_ok=True)
        filepath = os.path.join(output_dir, f"Different energies(freq={freq:.0e}).png")
        fig.savefig(filepath, dpi=150, bbox_inches='tight')
    plt.show()



def plot_gamma(save=False):
    medium = ISM(n_ism=0.1)
    obs = Observer(lumi_dist=1e26, z=0.1, theta_obs=0)
    rad = Radiation(eps_e=1e-1, eps_B=1e-3, p=2.3, xi_e=1)

    E_iso = 1e52
    theta_c = 0.1

    fig, ax = plt.subplots(figsize=(10, 6))

    cmap = cm.viridis
    norm = colors.LogNorm(vmin=min(gammas), vmax=max(gammas))


    lines = []

    ax.set_title(f"Different gammas(freq={freq:.0e})")
    ax.set_xlabel("Time(s)")
    ax.set_ylabel("Flux")

    for i, gamma in enumerate(gammas):
        jet = TophatJet(theta_c=theta_c, E_iso=E_iso, Gamma0=gamma, duration=10)
        model = Model(jet=jet, medium=medium, observer=obs, forward_rad=rad, resolutions=(0.3,3,5))
        times = np.logspace(0, 8, 200)
        bands = np.array([freq])
        results = model.specific_flux(times, bands)
        color = cmap(norm(gamma))

        line = ax.loglog(times, results['syn'][0,:], color=color, linewidth=2)
        lines.extend(line)

    sm = plt.cm.ScalarMappable(cmap=cmap, norm=norm)
    sm.set_array([])
    cbar = plt.colorbar(sm, ax=ax, shrink=0.8, pad=0.02)
    cbar.set_label('gamma', rotation=270, labelpad=20)

    cbar_ticks = gammas
    cbar.set_ticks(cbar_ticks)
    cbar.set_ticklabels([f'{e:.2f}' for e in cbar_ticks])
```

```
    ax.grid(True, alpha=0.3)
    plt.tight_layout()
    if save:
        output_dir = "new_plots"
        os.makedirs(output_dir, exist_ok=True)
        filepath = os.path.join(output_dir, f"Different gammas(freq={freq:.0e}).png")
        fig.savefig(filepath, dpi=150, bbox_inches='tight')
    plt.show()


def plot_theta_c(save=False):
    medium = ISM(n_ism=0.1)
    obs = Observer(lumi_dist=1e26, z=0.1, theta_obs=0)
    rad = Radiation(eps_e=1e-1, eps_B=1e-3, p=2.3, xi_e=1)

    E_iso = 1e52
    gamma0 = 200

    fig, ax = plt.subplots(figsize=(10, 6))

    cmap = cm.viridis
    norm = colors.LogNorm(vmin=min(angles), vmax=max(angles))

    lines = []

    ax.set_title(f"Different theta_c(freq={freq:.0e})")
    ax.set_xlabel("Time(s)")
    ax.set_ylabel("Flux")

    for i, theta in enumerate(angles):
        jet = TophatJet(theta_c=theta, E_iso=E_iso, Gamma0=gamma0, duration=10)
        model = Model(jet=jet, medium=medium, observer=obs, forward_rad=rad, resolutions=(0.3,3,5))
        times = np.logspace(2, 8, 200)
        bands = np.array([freq])
        results = model.specific_flux(times, bands)

        color = cmap(norm(theta))

        line = ax.loglog(times, results['syn'][0,:], color=color, linewidth=2)
        lines.extend(line)

    sm = plt.cm.ScalarMappable(cmap=cmap, norm=norm)
    sm.set_array([])
    cbar = plt.colorbar(sm, ax=ax, shrink=0.8, pad=0.02)
    cbar.set_label('theta_c (rad)', rotation=270, labelpad=20)

    cbar_ticks = angles
    cbar.set_ticks(cbar_ticks)
    cbar.set_ticklabels([f'{e:.3f}' for e in cbar_ticks])

    ax.grid(True, alpha=0.3)
    plt.tight_layout()
    if save:
        output_dir = "new_plots"
```

```
        os.makedirs(output_dir, exist_ok=True)
        filepath = os.path.join(output_dir, f"Different theta_c(freq={freq:.0e}).png")
        fig.savefig(filepath, dpi=150, bbox_inches='tight')
    plt.show()
```

B Data generation code for MCMC

```
import matplotlib.pyplot as plt
import numpy as np
from VegasAfterglow import ISM, TophatJet, Observer, Radiation, Model
import os
import random

angle = 0.1
energy = 1e51
gamma0 = 300

def gen_data():
    medium = ISM(n_ism=0.1)
    obs = Observer(lumi_dist=3.364e28, z=1.58, theta_obs=0)
    rad = Radiation(eps_e=1e-1, eps_B=1e-3, p=2.3, xi_e=1)

    jet = TophatJet(theta_c=angle, E_iso=energy, Gamma0=gamma0, duration=10)
    model = Model(jet=jet, medium=medium, observer=obs, forward_rad=rad, resolutions=(0.3,3,5))
    times = np.logspace(4, 6, 20)
    bands = np.array([4.84e14])
    fluxes = []
    for t in times:
        flux = model.specific_flux(np.array([t]), bands)
        fluxes.append([float(t),float(flux['syn'][0]),float(flux['syn'][0]*random.uniform(-0.1,0.1))])
    print(fluxes)
    results = np.array(fluxes)
    output_dir = "data"
    os.makedirs(output_dir, exist_ok=True)
    np.savetxt('data/r.csv', results, delimiter=',', header='t,Fv_obs,Fv_err', comments='')
```

C MCMC code

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import corner
from VegasAfterglow import ObsData, Setups, Fitter, ParamDef, Scale

data = ObsData()

band = [4.84e14]
lc_files = ["data/r.csv"]

for nu, fname in zip(band, lc_files):

    df = pd.read_csv(fname)
```

```
    data.add_light_curve(nu_cgs=nu, t_cgs=df["t"], Fnu_cgs=df["Fv_obs"], Fnu_err=df["Fv_err"])


cfg = Setups()
cfg.lumi_dist = 3.364e28
cfg.z         = 1.58
cfg.medium    = "ism"
cfg.jet       = "tophat"

mc_params = [
    ParamDef("E_iso",    1e50,  1e54,  Scale.LOG),
    ParamDef("Gamma0",      5,  1000,  Scale.LOG),
    ParamDef("theta_c",   0.0,   0.5,  Scale.LINEAR),
    ParamDef("theta_v",   0.0,   0.0,  Scale.FIXED),
    ParamDef("eps_e",    1e-2,   0.5,  Scale.LOG),
    ParamDef("eps_B",    1e-4,   0.5,  Scale.LOG),
    ParamDef("A_star",   1e-3,     1,  Scale.LOG),
    ParamDef("n_ism",    1e-3,    10,  Scale.LOG),
    ParamDef("xi",       1e-3,     1,  Scale.LOG),
]

fitter = Fitter(data, cfg)

result = fitter.fit(
    mc_params,
    resolution=(0.4, 1, 5),
    total_steps=10000,
    burn_frac=0.3,
    top_k=10
)

top_k_data = []
for i in range(result.top_k_params.shape[0]):
    row = {'Rank': i+1, 'chi^2': f"{-2*result.top_k_log_probs[i]:.2f}"}
    for name, val in zip(result.labels, result.top_k_params[i]):
        row[name] = f"{val:.4f}"
    top_k_data.append(row)

top_k_df = pd.DataFrame(top_k_data)
print("Top-k parameters:")
print(top_k_df.to_string(index=False))
```