# Geometrical Extension of Einstein's General Relativity

Name: Atreyi Dasgupta
Roll Number: 2021B5TS2797H

October 2023

# Contents

# 1 FLRW SPACE TIME

**Friedmann-Lemaître-Robertson-Walker (FLRW) Space-Time Metric**

The FLRW metric, a cornerstone of cosmology, represents a fundamental solution to Einstein's field equations within the framework of general relativity. It serves as a crucial tool for describing the large-scale structure and evolutionary dynamics of the universe. This metric is named after its notable contributors: Alexander Friedmann, Georges Lemaître, Howard Robertson, and Arthur Walker.

## 1.1 Key Features of FLRW Metric

The FLRW metric embodies key characteristics that underpin our understanding of the universe:

- **Homogeneity**: The universe is presumed to exhibit homogeneity, meaning it appears uniform and indistinguishable at every point when examined over significant scales.
- **Isotropy**: Isotropy denotes the universe's uniformity in all directions when observed over extensive scales. This characteristic forms a fundamental assumption of the FLRW metric.
- **Spatial Curvature**: The geometry of the universe is described by three distinct spatial curvatures: closed (positive curvature), flat (zero curvature), or open (negative curvature).
- **Scale Factor ($a$)**: The scale factor $a(t)$ acts as a pivotal parameter in the FLRW metric, delineating the universe's expansion over time. It quantifies the alteration in distances between cosmic objects as a function of cosmic time $t$.

## 1.2 The FLRW Metric Equation

The FLRW metric is mathematically expressed through the following line element:

$$ds^2 = -c^2 dt^2 + a(t)^2 \left[dr^2 + f(r) d\theta^2 + r^2 (d\phi^2 + \sin^2\theta d\theta^2)\right]$$

Where:

- $ds$ represents the infinitesimal proper time interval.
- $c$ signifies the speed of light.
- $a(t)$ characterizes the scale factor, a function reliant on time.
- $f(r)$ encapsulates the spatial curvature of the universe, denoted as $k = -1$ for an open universe, $k = 0$ for a flat universe, and $k = 1$ for a closed universe.

### 1.3 Interpreting the Metric

The metric is structured to encompass both time and spatial components. The first term is associated with time, with $t$ representing cosmic time. The subsequent term delineates spatial components, incorporating variables such as $r$, $\theta$, and $\phi$ that act as comoving coordinates.

### 1.4 Cosmic Evolution

The evolution of the universe hinges on the behavior of the scale factor $a(t)$. This parameter is fundamental for understanding the cosmos' expansion and its overall evolution. By studying $a(t)$, cosmologists gain valuable insights into the development of our universe.

## 2 Christoffel Symbols for FLRW Metric

The FLRW metric is represented by the metric tensor $G$, and we will calculate some of the associated Christoffel symbols.

### 2.1 Metric Components

## 3 Mathematical Expressions

Here are some mathematical expressions:

1. Expression for $g_{ij}$:
$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{a^2}{1-kr^2} & 0 & 0 \\ 0 & 0 & -a^2r^2 & 0 \\ 0 & 0 & 0 & -a^2r^2\sin^2\theta \end{pmatrix}$

2. Expression for $g^{ij}$
$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{1-kr^2}{a^2} & 0 & 0 \\ 0 & 0 & -\frac{1}{a^2r^2} & 0 \\ 0 & 0 & 0 & -\frac{1}{a^2r^2\sin^2\theta} \end{pmatrix}$

### 3.1 Christoffel Symbols

The formula for calculating the Christoffel Symbols is given as:

$$\Gamma^i_{jk} = \frac{1}{2}g^{il}\left(\frac{\partial g_{jl}}{\partial x^k} + \frac{\partial g_{lk}}{\partial x^j} - \frac{\partial g_{jk}}{\partial x^l}\right) \tag{3.1.1}$$

#### 3.1.1 Christoffel symbol $\Gamma^0_{11}$

$\frac{a\dot{a}}{1-kr^2}$

#### 3.1.2 Christoffel symbol $\Gamma^0_{22}$

$a\dot{a}r^2$

#### 3.1.3 Christoffel symbol $\Gamma^0_{33}$

$a\dot{a}r^2 sin^2\theta$

#### 3.1.4 Christoffel symbol $\Gamma^1_{01}= \Gamma^2_{02}= \Gamma^3_{03}$

$\frac{\dot{a}}{a}$

#### 3.1.5 Christoffel symbol $\Gamma^1_{11}$

$\frac{kr}{1-kr^2}$

#### 3.1.6 Christoffel symbol $\Gamma^2_{12} = \Gamma^3_{13}$

$\frac{1}{r}$

#### 3.1.7 Christoffel symbol $\Gamma^1_{22}$

$-r(1-kr^2)$

#### 3.1.8 Christoffel symbol $\Gamma^1_{33}$

$-r(1-kr^2)sin^2\theta$

#### 3.1.9 Christoffel symbol $\Gamma^2_{33}$

$-sin\theta cos\theta$

#### 3.1.10 Christoffel symbol $\Gamma^3_{23}$

$cot\theta$

### 3.2 Ricci Tensor

The Ricci tensor, denoted as $R_{ij}$, is a fundamental concept in General Relativity. It serves to represent the curvature of spacetime in gravitational fields. Specifically:

1. **Curvature of Spacetime**: The Ricci tensor reflects how the presence of mass and energy warps the fabric of spacetime. In the framework of General Relativity, massive objects cause the spacetime around them to curve, and the Ricci tensor encodes this curvature information.

2. **The Einstein Field Equations**: The Ricci tensor is a central component of Einstein's field equations, which establish the relationship between the distribution of matter and energy and the curvature of spacetime. These

equations form the basis of General Relativity and explain the origin of gravity and how objects move within gravitational fields.

3. **Gravitational Interactions**: The Ricci tensor plays a vital role in modeling gravitational interactions. It signifies how the presence of mass and energy induces spacetime curvature, and this curvature, as described by the Ricci tensor, governs the gravitational behavior of objects.

4. **Tidal Forces**: Different components of the Ricci tensor describe how tidal forces manifest in a gravitational field. Tidal forces denote variations in the gravitational forces experienced by various parts of an extended object, and these variations are characterized by the components of the Ricci tensor.

The formula for computing the Ricci tensor, denoted as $R_{ij}$, is given as:

$$R_{ij} = \frac{\partial \Gamma^l_{ij}}{\partial x^l} - \frac{\partial \Gamma^l_{il}}{\partial x^l} + \Gamma^m_{ij}\Gamma^l_{lm} - \Gamma^m_{il}\Gamma^l_{jm} \tag{3.2.1}$$

Here, Latin alphabet indices $i, j, l, m$ represent components and $x^l$ denotes the coordinate variable.

# 4 F$(T, T_G)$ Gravity

The Universe seems to have undergone two significant phases—early inflation and later acceleration. Researchers have explored two main paths to explain this phenomenon.

The first path involves changing what the Universe is made of. This means introducing new elements like scalar fields, vector fields, and more. These introduce concepts like the inflaton (related to early inflation) and dark energy (related to late acceleration). Many different models fall under this category, providing various explanations.

The second path involves tweaking the way gravity works. Instead of modifying the content of the Universe, we modify how gravity operates. It's interesting to note that you can switch between these paths because what matters is the number of additional factors beyond our standard understanding of particles and general relativity.

In modified gravitational theories, we often build upon the curvature-based Einstein-Hilbert action. But there's another intriguing approach using a different formulation, called the "teleparallel equivalent of general relativity" (TEGR). In TEGR, gravity is attributed to torsion instead of curvature. This involves using a connection known as the Weitzenböck connection instead of the more common Levi-Civita connection. The gravitational Lagrangian in this framework is determined by contractions of the torsion tensor, referred to as the "torsion scalar" T, akin to the "curvature scalar" R in general relativity.

Just like how we can extend general relativity with $f(R)$ modifications, we can also develop $f(T)$ extensions of TEGR. The interesting part is that $f(T)$ gravity is different from $f(R)$ gravity, even though TEGR itself aligns with general relativity. This novel class of gravitational modification has led to extensive studies on how it impacts the cosmos and solutions related to black holes.

But the modifications don't stop at simple changes to the curvature of gravity. More complex actions can be crafted, including higher-curvature corrections like the Gauss-Bonnet combination $G$ or arbitrary functions $f(G)$. This same approach can be applied to the teleparallel formulation of gravity, introducing higher-torsion corrections.

In recent work, a new quartic torsional scalar called $T_G$ was developed, which reduces to a topological invariant in four dimensions. Combining this with the torsion scalar $T$, a new gravity theory called $F(T, T_G)$ gravity emerged. This constitutes a fresh class of gravitational modifications distinct from both $f(T)$ and $f(R, G)$ gravity.

## 4.1 Torsion Scalar (T)

In the realm of understanding gravity, there's an alternative perspective that doesn't rely on spacetime curvature but on something called "torsion". To achieve this, we depart from the idea that the antisymmetric part of the connection is zero and instead use what's known as the Weitzenböck connection. In this framework, we create the torsion tensor, which contains all the information about the geometry and, consequently, about gravitational forces.

From this, we can build simple mathematical quantities that involve the first-order derivatives of something called the vierbein. These mathematical quantities are combined to create what's called the "torsion" scalar, denoted as T. This T scalar becomes the heart of the gravitational Lagrangian, which, when used to derive the equations of motion by varying it with respect to the vierbein, results in the same gravitational field equations as those in general relativity.

Einstein was so intrigued by this formulation that he named it the "teleparallel equivalent of general relativity" (TEGR) because, despite its different approach, it yields the same gravitational outcomes as the well-known theory of general relativity.

In the teleparallel formulation of gravity, we take a different perspective than traditional general relativity. Instead of focusing on spacetime curvature, we work with something called "torsion." This involves two key components: the vielbein field, denoted as $e_a^\mu(x^\mu)$, and the connection 1-forms, written as $\omega_a^b(x^\mu)$, which define parallel transportation.

We can express these components using coordinates as:

$$e_a = e_a^\mu \partial_\mu \quad \text{and} \quad \omega_a^b = \omega_{a\mu}^b dx^\mu = \omega_a^b ec.$$

We also introduce the dual vielbein as:

$$e^a = e_a^\mu dx^\mu.$$

The commutation relations for vielbein can be defined as:

$$\frac{1}{2}[e_a, e_b] = C_{ab}^c e_c,$$

where the structure coefficients $C^c_{ab}$ are given by:

$$C^c_{ab} = e^\mu_a e^\nu_b (e_{c\mu;\nu} - e_{c\nu;\mu}),$$

with a comma denoting differentiation.

We can also define the torsion tensor, expressed in tangent components as:

$$T_a^{bc} = \omega_a^{cb} - \omega_a^{bc} - C_a^{bc},$$

while the curvature tensor is defined as:

$$R_a^{bcd} = \omega_a^{bd;c} - \omega_a^{bc;d} + \omega_e^{bd}\omega_a^{ec} - \omega_e^{bc}\omega_a^{ed} - C^{cd}\omega_a^{be}.$$

To work with these elements effectively, we use the metric tensor $g$, which makes the vielbein orthonormal, given by:

$$g(e_a, e_b) = \eta_{ab},$$

where $\eta_{ab} = \mathrm{diag}(-1, 1, \ldots, 1)$.

This allows us to express the metric tensor $g^{\mu\nu}$ as:

$$g^{\mu\nu} = \eta^{ab} e^\mu_a e^\nu_b,$$

and indices $a, b, \ldots$ are raised/lowered using the Minkowski metric $\eta_{ab}$.

We further define the contorsion tensor as:

$$K^{abc} = \frac{1}{2}(T^{cab} - T^{bca} - T^{abc}) = -K^{bac}.$$

In this formulation, we impose the condition of teleparallelism, where $R_a^{bcd} = 0$ holds in all frames. This condition can be realized by assuming a specific connection known as the Weitzenböck connection.

The Ricci scalar $\bar{R}$ corresponding to the Levi-Civita connection can be expressed as:

$$e\bar{R} = -eT + 2(eT^\nu_{\nu;\mu}),$$

where we've defined the "torsion scalar" $T$ as:

$$T = \frac{1}{4}T^{\mu\nu\lambda}T_{\mu\nu\lambda} + \frac{1}{2}T^{\mu\nu\lambda}T_{\lambda\nu\mu} - T^{\nu\nu\mu}T^{\lambda\lambda\mu},$$

and $e = \det(e^\mu_a) = \sqrt{|g|}$.

## 4.2 Gauss-Bonnet Gravity

The construction of the Teleparallel Equivalent of General Relativity (TEGR) involved expressing the Ricci scalar $R$ for a general connection as the Ricci scalar $\bar{R}$ calculated with the Levi-Civita connection, plus additional terms arising from the torsion tensor. By imposing the condition of teleparallelism ($R^a_{bcd} = 0$), we found that $\bar{R}$ can be expressed as a torsion scalar plus a total derivative.

Now, we can follow a similar approach but use the Gauss-Bonnet combination, $G = R^2 - 4R_{\mu\nu}R^{\mu\nu} + R_{\mu\nu\kappa\lambda}R^{\mu\nu\kappa\lambda}$, instead of the Ricci scalar. In a previous work [28], we derived the teleparallel equivalent of Gauss-Bonnet gravity characterized by a new torsion scalar $T_G$ and the equations of motion for the modified gravity defined by the function $F(T, T_G)$.

When working with the Weitzenböck connection ($\omega_a^{bc} = 0$), we find that the Gauss-Bonnet term $\bar{G}$ calculated by the Levi-Civita connection can be expressed as:

$$e\bar{G} = eT_G + \text{total divergence}.$$

The torsion scalar $T_G$ in this context is given by:

$$T_G = K^{a_1}{}_{e_{a_1}} K^{e_{a_2}}{}_{e_{b_2}} K^{a_3}{}_{f_c} K^{f}{}_{a_4 d} - 2K^{a_1 a_2 a} K^{a_3 e_{b_2}}{}_{e_b} K^{e_{b_2}}{}_{f_c} K^{f}{}_{a_4 d}$$

$$+2K^{a_1 a_2 a} K^{a_3 e_{b_2}}{}_{e_b} K^{e_{b_2}}{}_{a_4 f} K^{f}_{cd} + 2K^{a_1 a_2 a} K^{a_3 e_{b_2}}{}_{e_b} K^{e_{b_2}}{}_{a_4 c; d}) \delta^{abcd}_{a_1 a_2 a_3 a_4},$$

where $\delta^{abcd}$ is the determinant of the Kronecker deltas.

In summary, $T_G$ serves as the teleparallel equivalent of the Gauss-Bonnet term $\bar{G}$.

# 5 Late-time constraints on Gauss-Bonnet cosmology

Here, we investigate a gravitational action that combines the Ricci scalar ($R$) and the topological Gauss-Bonnet term ($G$). Specifically, we focus on a specific class of modified gravity theories expressed as $f(R, G) = R^n G^{1-n}$. These theories are chosen based on symmetry considerations.

We concentrate on a scenario where the universe is spatially flat, homogeneous, and isotropic. In this context, we want to demonstrate that we can account for the observed acceleration of the universe using the underlying geometry itself, thus avoiding the issues associated with the cosmological constant.

Our methodology involves validating the Friedmann equations in the presence of pressureless matter to understand how the Hubble expansion rate varies with redshift. To validate our model, we impose constraints on the theory's free parameters using a Bayesian Monte Carlo method, which is applied to late-time cosmic observations.

## 5.1 Monte Carlo Markov Chain

MCMC is a powerful statistical technique used for sampling from probability distributions. It plays a crucial role in model exploration and parameter estimation. Here's an elaboration on what MCMC does:

- **Sampling and Parameter Adjustment:** MCMC involves the iterative sampling of model parameters from a probability distribution. The algorithm adjusts these parameters based on the likelihood of observed data, aiming to explore regions of the parameter space that contribute significantly to the model's fit.

- **Convergence to High-Likeness Regions:** Through successive iterations, MCMC aims to converge towards regions of the parameter space with high likelihood. This process refines the model, making it more consistent with the observed data and improving its overall accuracy.

- **Iterative Model Refinement:** MCMC operates in an iterative fashion, continually refining the model based on the information gained from the exploration of the parameter space. This iterative refinement enhances the model's predictive power and reliability.

- **Parallelized Walker Exploration:** To expedite the exploration process, MCMC often employs parallelized walker exploration. This means distributing the task among multiple walkers, allowing for simultaneous exploration of different regions within the parameter space.

- **Facilitating Complex Model Configurations:** MCMC is particularly useful when dealing with complex models with high-dimensional parameter spaces. It enables a more comprehensive exploration, ensuring that a diverse set of model configurations is considered.

## 5.2 Fitting $\Lambda_{CDM}$ model using MCMC Analysis.

The initial condition for the Hubble parameter, denoted as $H(0)$, is straightforward: $H(0) = H_0$. To determine the second initial condition, we aim to ensure that, at the present time, the first derivative of the Hubble parameter aligns with the predictions of the standard Cold Dark Matter (CDM) model. The CDM model is described by the expansion law:

$H_{\Lambda CDM} = 100h\sqrt{\Omega(1+\text{z})^3 + 1 - \Omega}$

By taking the first derivative of this equation with respect to $z$, we obtain:

$$H'_{\text{CDM}} = \frac{3H_0\Omega_{m0}(1+z)^2}{2\left[\Omega_{m0}(1+z)^3 + 1 - \Omega_{m0}\right]}$$

This expression determines the second initial condition for our model, specifically $H(0) = \frac{3H_0\Omega_(m0)}{2}$.

In our numerical analysis, we use the reduced Hubble constant $h$, defined as $h \equiv \frac{H_0}{100\,\text{km s}^{-1}\,\text{Mpc}^{-1}}$. This parameter, along with $m_0$, plays a crucial role in our model.

The code with MCMC Analysis is provided right here:

```
import numpy as np
from astropy.io import ascii
from scipy.optimize import curve_fit
import matplotlib.pyplot as plt
import emcee
```

```
import corner
%matplotlib inline

def model(parameters):
h, omega = parameters
return 100*h*np.sqrt(omega*(1+redshift)**3 + 1-omega)


def lnlike(parameters, x, y, yerr):
    return -0.5 * np.sum(((y - model(parameters))/yerr) ** 2)


def lnprior(parameters):
    h, omega = parameters
    if 0.5<h<0.8 and 0<omega<0.4:
        return 0.0
    return -np.inf


def lnprob(parameters, x, y, yerr):
    lp = lnprior(parameters)
    if not np.isfinite(lp):
        return -np.inf
    return lp + lnlike(parameters, x, y, yerr)


Yerr = data_original[:,2]
data = (hubble, redshift ,Yerr)
nwalkers = 700
niter = 400
initial = np.array([0.6, 0.3])
ndim = len(initial)
p0 = [np.array(initial) + 1e-7 * np.random.randn(ndim) for i in range(nwalke


def main(p0, nwalkers, niter, ndim, lnprob, data):
    sampler = emcee.EnsembleSampler(nwalkers, ndim, lnprob, args=data)

    print("Running burn-in...")
    p0, _, _ = sampler.run_mcmc(p0, 100)
    sampler.reset()

    print("Running production...")
    pos, prob, state = sampler.run_mcmc(p0, niter)

    return sampler, pos, prob, state
```

```
sampler, pos, prob, state = main(p0, nwalkers, niter, ndim, lnprob, data)

def plotter(sampler, redshift = redshift ,hubble = hubble):
    plt.ion()
    plt.scatter(redshift, hubble, color='b')
    samples = sampler.flatchain
    for parameters in samples[np.random.randint(len(samples), size=100)]:
        plt.plot(redshift, model(parameters), color="r", alpha=0.1)
    plt.ticklabel_format(style='sci', axis='x', scilimits=(0, 0))
    plt.xlabel('redshift')
    plt.ylabel('hubble')
    plt.legend()
    plt.show()


plotter(sampler)
```

ON running this code, we get the following output

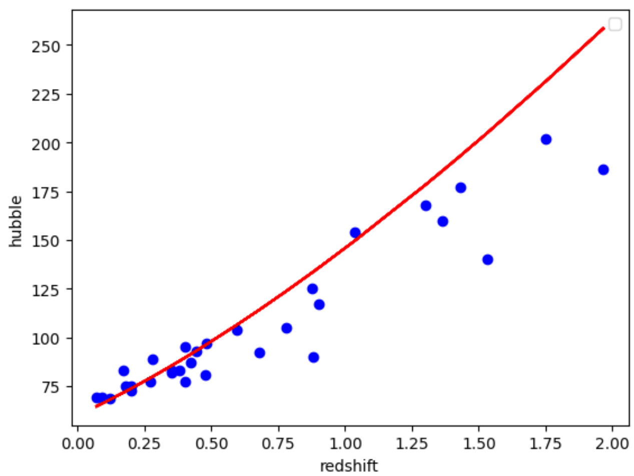


Figure 1: Output

This suggests that the MCMC function has run properly as the model is fitting properly on the observed data.

The optimised values of the parameters come out to be : $\Omega = 0.3$ and h = 0.6.

### 5.3 Numerical Differentiation of higher order ODE

Assuming matter behaves as a pressureless perfect fluid, we can express the matter density ($\rho_m$) as:

$$\rho_m = 3H_0^2\Omega_{m_0}(1+z)^3$$

where $\Omega_{m0}$ is the current value of the matter density parameter. Thus, for the specific model under consideration, the first Friedmann equation takes the form:

$$-\frac{H^2\cdot(n-1)\cdot(1+z)^2}{(H-(1+z)\cdot dHdz)^2\cdot(2H-(1+z)\cdot dHdz)}-\frac{dHdz^2\cdot 5\cdot H^3\cdot(n-1)\cdot(1+z)^2}{(H-(1+z)\cdot dHdz)^2\cdot(2H-(1+z)\cdot dHdz)}+\frac{dHdz\cdot 3\cdot H^4\cdot(n-1)\cdot(1+z)}{(H-(1+z)\cdot dHdz)^2\cdot(2H-(1+z)\cdot dHdz)}+\frac{dHdz^3\cdot 2\cdot H^2\cdot(n-1)\cdot(1+z)^3}{(H-(1+z)\cdot dHdz)^2\cdot(2H-(1+z)\cdot dHdz)}+\frac{H0^2\cdot 4^{(n-1)}\cdot(1+z)^3\cdot\Omega_{m0}}{n}\cdot\left(\frac{H^2\cdot(H-(1+z)\cdot dHdz)}{2H-(1+z)\cdot dHdz}\right)^{(n-1)}=H^2$$

Used the odeint method to find H' values and H" Values and also plotted H' vs z plots and H" vs z plots.

- `odeint` is commonly used to solve systems of first-order ordinary differential equations (ODEs).
- For higher-order differential equations, we can convert them to first-order form by introducing new variables.
- A function is then defined to represent the system of first-order differential equations, allowing `odeint` to be applied.
- The corresponding code for the following function is given in the next slide.

```
    # Define your ODE function 'f' here
def f(u, z, H0, Omega_m0, n):
    H, dHdz = u
    # Define the ODEs here
    dHdt = dHdz
    dHdz = -((H**2) * (n - 1) * (1 + z)**2) / ((H - (1 + z) * dHdz)**2 *
    (2 * H - (1 + z) * dHdz)) - dHdz**2) * (5 * (H**3) * (n - 1) * (1 + z)**2)
    / ((H - (1 + z) * dHdz)**2 * (2 * H - (1 + z) * dHdz)) + \
        dHdz * (3 * (H**4) * (n - 1) * (1 + z)) / ((H - (1 + z) * dHdz)**2 *
        (2 * H - (1 + z) * dHdz)) + dHdz**3) * (2 * (H**2) * (n - 1) *
        (1 + z)**3) ((H - (1 + z) * dHdz)**2 * (2 * H - (1 + z) * dHdz)) + \
        ((H0**2 * 4**(n - 1) * (1 + z)**3 * Omega_m0) / n) * (((H**2) *
        (H - (1 + z) * dHdz)) / (2 * H - (1 + z) * dHdz))**(n - 1) - (H**2)

    return [dHdt, dHdz]

# Define H0, Omega_m0, n, and other parameters
H0 = 70
Omega_m0 = 0.3
n = 1.2

# Assuming data is loaded and has at least two columns
```

```
zs = np.sort(data[:, 0])  # Sort the time values
# zs = np.unique(zs)  # Remove duplicates, if any

# Define initial conditions
y0 = [H0, (3 * H0 * Omega_m0) / 2]

# Solve the ODE
os = odeint(f, y0, zs, args=(H0, Omega_m0, n))
```

Using the following odient function, we plot H vs z , H' vs z and H" vs z plots given below.

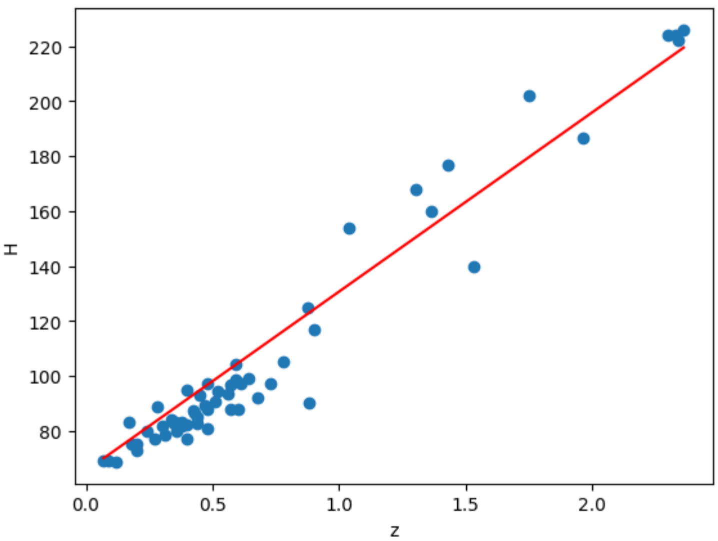


Figure 2: H vs z

Looking at H vs Z plot, we can confirm the validity of the values as it is giving a good fit with the observed data.

Moreover, the plot seems to be a straight line which implies that the slope is constant throughout. We will see that in the next plot.

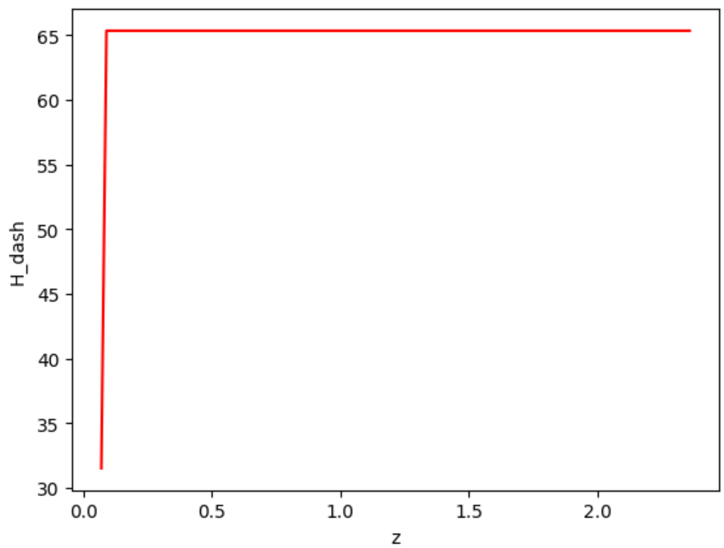


Figure 3: H' vs z

As we had predicted, we see that the slope is indeed a straight line and H' values are around 65.09.

Now we see the H" vs z plot.

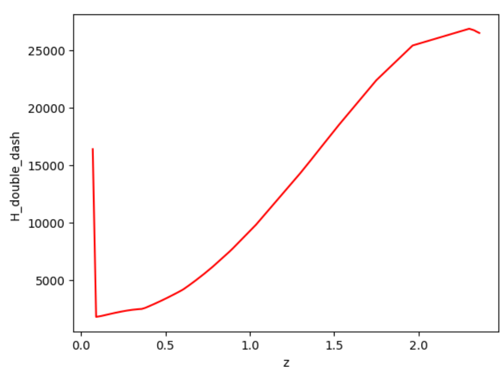


Figure 4: H" vs z

# 6 Conclusion

Now all that is left is to substitute the values of H' and H" in the Friedmann equation given above and fit for H. MCMC function has to be used to determine the best fit parameter values of $\Omega$ , n and h.

In conclusion, the framework for model exploration discussed involves key components such as the model-generating function, ensemble of walkers, systematic parameter space exploration, and the implementation of Markov Chain Monte Carlo (MCMC) steps for iterative model refinement. The MCMC technique plays a crucial role in converging towards high-likelihood regions of the parameter space, allowing for the adjustment of model parameters to better align with observed data. Furthermore, the numerical analysis considers the reduced Hubble constant $h$ as a free parameter, along with $m_0$, providing flexibility in exploring a wide range of model configurations. Additionally, the derivation of initial conditions for the Hubble parameter in the context of a specific model showcases the methodological considerations involved. The utilization of `odeint` for solving systems of first-order ordinary differential equations is highlighted, emphasizing the importance of converting higher-order equations to first-order form for efficient numerical solutions. Overall, this comprehensive approach provides a robust foundation for model exploration and refinement in cosmological studies.