Here I provide some description of notebooks I created over the years. The Minimal Version of the Hypergeometrical Universe Theory was provided in the main paper. Certain aspects of the theory were not explained because people has to be able to understand the basics before learning the more advanced parts of HU.

Some notebooks are deadends. If you have any questions reach out and I will try to remember what I want to do with an specific notebook or library.

Summary and Initial Documentation for `AAA_Final_4D_Cross_Product.ipynb`
Overview
This notebook appears to involve computations related to rotations and cross products in a 4-dimensional space. The content includes symbolic representations and matrices, aligning with the Hypergeometrical Universe Theory's approach to 4D spatial dynamics.

Notebook Purpose
The purpose of `AAA_Final_4D_Cross_Product.ipynb` seems to be the implementation of a method for computing cross products in 4-dimensional space, along with a rotation matrix for 4D vectors. This is consistent with 4D spatial modeling, which is central to your HU framework.

Cells and Their Purpose

1. **Imports and Variable Definitions**:

    - Imports essential libraries, `numpy` and `sympy`, for numerical and symbolic computation.
    - Defines symbolic variables for coefficients and angles used in 4D vector rotations and cross products.

2. **4D Cross Product Placeholder**:

    - Defines a placeholder function `cross_product_4d`, which takes three vectors (`u`, `v`, `w`) and computes a cross product for a 4D subspace.
    - Uses a symbolic skew-symmetric matrix (`A`) setup, where the entries are based on standard 4D vector rules. However, the implementation hints that it's a placeholder rather than a complete solution.

3. **Skew-Symmetric Matrix and Rotation Matrix Setup**:

    - Defines `skew_symmetric` to create a skew-symmetric matrix from a 4D vector, essential for rotation operations in 4D.
    - Constructs a 4D rotation matrix `R` using symbolic sine and cosine terms, following principles of 4D spatial rotation around a subspace spanned by given vectors.

4. **Rotation of a 4D Vector**:

    - A symbolic 4D vector `p` is defined and then rotated using the matrix `R`.
    - The rotated vector `p_rotated` is calculated, showing the application of the 4D rotation matrix to a vector.

Next Steps for Documentation
Based on this summary, I'll write up detailed descriptions for each function and matrix transformation in the notebook, using references to 4D cross products and rotation concepts relevant to HU Theory. Let me proceed with generating specific documentation for these steps.

Here's the initial documentation for `AAA_Final_4D_Cross_Product.ipynb`. This documentation is organized based on each cell's function and is integrated with relevant theoretical context.

Notebook: AAA_Final_4D_Cross_Product.ipynb
Purpose: This notebook explores cross products and rotations in 4-dimensional space, in line with the Hypergeometrical Universe Theory (HU). These calculations support the HU model, where particles are described as wave generators in a 4D spatial manifold.

Cell Explanations and Documentation

1. **Imports and Symbolic Variable Definitions**:

```
import numpy as np

import sympy as sp

# Symbolic variables for coefficients and angles

a, b, c, theta = sp.symbols('a b c theta')

# Define symbolic 4D vectors

v1 = np.array([sp.symbols('v1_x'), sp.symbols('v1_y'), sp.symbols('v1_z'),
sp.symbols('v1_w')])
```

```
v2 = np.array([sp.symbols('v2_x'), sp.symbols('v2_y'), sp.symbols('v2_z'),
sp.symbols('v2_w')])

v3 = np.array([sp.symbols('v3_x'), sp.symbols('v3_y'), sp.symbols('v3_z'),
sp.symbols('v3_w')])

# Linear combination of basis vectors defining a rotation axis

v = a * v1 + b * v2 + c * v3
```

- **Purpose**: Sets up symbolic variables and 4D vectors (`v1`, `v2`, `v3`) for use in cross-product and rotation computations. The vector `v` is a symbolic linear combination that serves as the rotation axis.
- **HU Context**: This setup supports HU's 4D vector space, where particles exist as shapeshifting deformations along a hyperspherical locus.

2. **4D Cross Product (Placeholder)**:

```
def cross_product_4d(u, v, w):

   # Placeholder function for 4D cross product computation

   A = np.array([

      [0, -u[2], u[1], u[3]],

      [u[2], 0, -u[0], -u[3]],

      [-u[1], u[0], 0, u[3]],

      [-u[3], u[3], -u[3], 0]

   ])

   return np.dot(A, v) + np.dot(np.dot(A, w), v)  # This is not the actual math

# Orthogonal vector computation

u = cross_product_4d(v1, v2, v3)

sp.pprint(u)
```

- **Purpose**: A placeholder function that approximates a 4D cross product. It uses a skew-symmetric matrix `A` but isn't fully formulated for general 4D cross products.

- **HU Context**: The 4D cross product aligns with the concept of particles as 4D wave generators in HU, where orthogonality and vector projections are key for defining particle behavior in the 4D hyperspherical framework.

3. **Skew-Symmetric Matrix and Rotation Matrix Setup**:

```
def skew_symmetric(v):
    return np.array([
        [    0, -v[2],  v[1], -v[3]],
        [ v[2],     0, -v[0],  v[3]],
        [-v[1],  v[0],     0, -v[3]],
        [ v[3], -v[3],  v[3],     0]
    ])

# Rotation matrix for 4D vector space
A = skew_symmetric(u)
I = np.eye(4)
R = I + sp.sin(theta) * A + (1 - sp.cos(theta)) * A @ A
R
```

   - **Purpose**: Defines a `skew_symmetric` matrix, used in constructing the 4D rotation matrix `R`. This matrix represents a rotation transformation in the 4D vector space based on symbolic angle `theta`.
   - **HU Context**: The rotation matrix `R` is used to apply transformations in the HU model, where particles "surf" 4D metric waves. This transformation is significant for capturing rotation effects across 4D.

4. **Applying Rotation to a 4D Vector**:

```
p = np.array([sp.symbols('p_x'), sp.symbols('p_y'), sp.symbols('p_z'), sp.symbols('p_w')])
p_rotated = R @ p
p_rotated
```

- **Purpose**: Defines a 4D vector `p` and applies the rotation matrix `R` to it, yielding the rotated vector `p_rotated`.
- **HU Context**: This rotated vector represents how a particle's position or orientation would change within the 4D spatial framework as per HU Theory. It reflects the dynamic positioning of particles in the 4D space.

Summary and Initial Documentation for `AAA_Final_Alpha_Analysis.ipynb`
Overview
This notebook appears to focus on calculating fundamental constants and evaluating parameters central to HU Theory, particularly focusing on the fine-structure constant (`alpha`). It also explores fundamental interactions, including force derivations between Fundamental Dilators (FDs) and concepts such as Compton wavelength, Universe thickness, and other cosmological constants.

Notebook Purpose
The purpose of `AAA_Final_Alpha_Analysis.ipynb` is to perform symbolic and numeric calculations relevant to the fine-structure constant and related parameters within the HU framework. It includes detailed derivations to approximate values for the Universe's physical characteristics, such as its thickness and the force between Fundamental Dilators.

Cells and Their Purpose

1. **Imports and Constant Definitions**:

   import numpy as np

   from astropy import constants as cc, units as uu

   - **Purpose**: Imports `numpy` for numerical calculations and `astropy` for physical constants and units, setting up a foundation for precise calculations based on established physical constants.

2. **Fine-Structure Constant (`alpha`) Calculation**:

   # fine constant

   alpha = (1/4/np.pi/cc.eps0*cc.e.si**2/cc.hbar/cc.c).si

   - **Purpose**: Calculates the fine-structure constant, a dimensionless constant representing the strength of electromagnetic interactions, foundational to HU's electromagnetic modeling.
   - **HU Context**: `alpha` is integral to HU's modeling of the electromagnetic field as it interacts with 4D Fundamental Dilators (FDs).

3. **Mass and Wavelength Calculations**:

   m_H = (cc.m_p + cc.m_e) / 2

   N = (uu.kg / m_H).si

   lambda_1_value = cc.h / (m_H * cc.c)

   P = (cc.eps0 * cc.c**2 * lambda_1_value / (2 * np.pi**2) / N / cc.e.si**2 * uu.kg).si * (2 * np.pi)

   B = 1 / (4 * np.pi**2 * alpha)

   lambda_1_value, N, P * (2 * np.pi * alpha)

   - **Purpose**: Calculates average mass `m_H`, the fundamental Compton wavelength (`lambda_1_value`), and terms related to particle density (`N`) and pressure (`P`) based on HU's fundamental units.
   - **HU Context**: These parameters are essential for approximating the interaction scales and distances relevant to HU, providing a basis for further cosmological calculations, including Universe thickness.

4. **Further Symbolic Derivations Using SymPy**:

   from sympy import symbols, cos, diff, pi, simplify, sin, init_printing, solve

   init_printing()

   # Define symbolic constants and expressions

   r, R, lambda_1, lambda_2, alpha, N, P, e, epsilon_0, h, c = symbols('r R lambda_1 lambda_2 alpha N P e epsilon_0 h c')

   k1 = 2 * pi / lambda_1

   k2 = 2 * pi * N / lambda_1

   lambda_2 = lambda_1 / N

   P = 1 / (2 * pi * alpha)

   # Define and simplify Phi functions and their derivatives

   ...

   solution_r = solve(dPhi_total_dr_simplified, r)

- **Purpose**: Uses symbolic calculations to define and manipulate expressions for potential fields and wavefunctions (`Phi_1`, `Phi_2`) and their derivatives.
- **HU Context**: This symbolic approach supports HU's quantum mechanical interpretation of Fundamental Dilators, allowing precise calculations of field potentials in a 4D hypersphere. The derivations here approximate the interaction effects based on small perturbations in fields around the Fundamental Dilators.

5. **Force Between Fundamental Dilators**:

   # Define and substitute expressions to solve for force F between Fundamental Dilators

   F = (m_0 * c**2 * x / lambda_1**2)

   F_substituted = F.subs(kg, N * h / c / lambda_1)

   F_simplified = simplify(F_substituted)

   - **Purpose**: Derives an expression for the force between Fundamental Dilators in HU, involving constants such as `m_0` (dilator mass), `c` (speed of light), and `lambda_1` (Compton wavelength).
   - **HU Context**: This force equation encapsulates HU's model of gravitational and electromagnetic interactions as Van der Waals-like forces between metric wave distortions in the 4D hyperspace.

6. **Cosmological Calculation - Universe Thickness**:

   t1 = lambda_1_value / cc.c

   alpha_value = cc.e.si**2 / (2 * cc.eps0 * cc.h * cc.c)

   A = 1 / (2 * np.pi * alpha_value)

   B = 2 * np.pi**2

   TotalRadius = (A / (2 * np.pi**2))**(1/3)

   UniverseThickness = (TotalRadius - 1) / 4

   - **Purpose**: Calculates the thickness of the Universe based on HU's model parameters.
   - **HU Context**: This approximation of Universe thickness is aligned with HU's cosmological interpretation, where the hyperspherical Universe expands in a structured way, yielding distinct layers.

Summary and Initial Documentation for
`AAA_Final_Approximations4UniverseMap.ipynb`

Overview

This notebook focuses on calculating the number of basis functions required for mapping the Universe. It uses the `healpy` library, commonly applied in astrophysics for spherical data representation, and `numba` for efficient computation. These approximations are essential for modeling the Universe in the HU framework.

Notebook Purpose

The primary goal of `AAA_Final_Approximations4UniverseMap.ipynb` is to calculate the number of basis functions needed for a Universe map representation at different resolutions. This mapping process aligns with HU's 4D cosmology, where understanding the distribution of spherical harmonics over the cosmic hypersurface is key.

Cells and Their Purpose

1. **Imports and Function Definition for Basis Functions**:

```
import healpy as hp

import numpy as np

from numba import jit

@jit

def numberOfBaseFunctions(kk):

    count = 0

    for k in np.arange(kk+1):

        for l in np.arange(k+1):

            for m in np.arange(-l-1, l+1):

                count += 1

    return count
```

   - **Purpose**: Imports necessary libraries (`healpy` for spherical harmonics, `numba` for optimization) and defines a function `numberOfBaseFunctions`. This function counts the number of basis functions (spherical harmonics) up to a given order `kk`.

- **HU Context**: The basis function count is critical for modeling spherical harmonic structures within HU, representing large-scale cosmic structures as projected onto a hyperspherical universe.

2. **High-Resolution Problem Dimensions**:

   numberOfBaseFunctions(48), hp.nside2npix(nside=1024)

   - **Purpose**: Calculates the number of basis functions for `kk = 48` and uses `healpy` to get the number of pixels for `nside = 1024`, a high-resolution setup.
   - **HU Context**: This resolution aligns with HU's approach to capturing fine-grained details on the cosmic hypersurface. High-resolution mapping provides more precise data for structures like galaxy distributions or cosmic microwave background patterns.

3. **Approximate Solution for Lower Resolution**:

   numberOfBaseFunctions(49), hp.nside2npix(nside=64)

   - **Purpose**: Calculates the number of basis functions for `kk = 49` and the pixel count for `nside = 64`, which is a lower resolution.
   - **HU Context**: Lower resolutions are used for approximate or preliminary mappings in HU. Such approximations are useful for scaling down calculations when exploring large-scale structures without focusing on small details.

Summary and Initial Documentation for `AAA_Final_Astroquery.ipynb`
Overview
This notebook uses the `astroquery` library to retrieve astronomical data, specifically focusing on querying objects like the Crab Nebula (`m1`) and Andromeda Galaxy (`m31`). It also includes calculations involving Sagittarius A* (the Milky Way's central black hole), focusing on parameters like mass and radius.

Notebook Purpose
The main objective of `AAA_Final_Astroquery.ipynb` is to demonstrate and document astronomical data retrieval using `astroquery`, with additional astrophysical calculations involving black hole properties. These data retrievals and calculations are useful for validating HU-based cosmological models against observed astrophysical data.

Cells and Their Purpose

1. **Notebook Content and Introduction**:

   # CONTENT

Astroquery Example

- **Purpose**: Provides an introductory markdown cell, indicating that the notebook will showcase examples of using `astroquery` for data retrieval.
- **HU Context**: This introduction sets up the purpose of querying astronomical databases to gather empirical data supporting HU Theory, particularly regarding cosmic structures.

2. **Imports and Setup**:

```
import astroquery

from astroquery.simbad import Simbad

from astropy import units as uu, constants as cc

import numpy as np

from parameters import *
```

- **Purpose**: Imports essential libraries, including `astroquery` for accessing astronomical databases and `astropy` for constants and unit handling.
- **HU Context**: These tools enable querying for astrophysical objects, which is central to obtaining observable parameters (e.g., mass, distance) that can be cross-referenced with HU-based predictions.

3. **Query Example - Crab Nebula (`m1`)**:

```
result_table = Simbad.query_object("m1")

result_table
```

- **Purpose**: Queries the SIMBAD database for the Crab Nebula (`m1`) and displays the resulting data table.
- **HU Context**: By examining objects like the Crab Nebula, the notebook supports HU-based studies, potentially linking emission or spatial properties of nebulae to specific predictions of the HU model.

4. **Extended Query - Andromeda Galaxy (`m31`)**:

```
s = Simbad()

s.add_votable_fields('bibcodelist(2003-2013)', "z_value")

r = s.query_object('m31')
```

r.pprint()

- **Purpose**: Performs an extended query on `m31`, including bibliographic data and redshift (`z_value`).
- **HU Context**: Data on the Andromeda Galaxy's redshift and bibliographic history may provide observational support to HU's approach to cosmic expansion and redshift interpretation.

5. **Object Query - UGC 7831**:

   s.query_object('ugc7831')

   - **Purpose**: Queries `ugc7831`, which is another galaxy, providing positional and structural information for further analysis.
   - **HU Context**: Information on this galaxy adds another comparative data point for HU's model, potentially validating HU's explanation of galaxy distribution and dynamics.

6. **NED Database Query for Galaxy Positions**:

   from astroquery.ipac.ned import Ned

   result_table = Ned.get_table("ugc 7831", table='positions')

   print(result_table)

   result_table['No.','RA','DEC','Frequency']

   - **Purpose**: Uses the NED database to obtain positional data for `ugc 7831`, focusing on right ascension (RA), declination (DEC), and frequency.
   - **HU Context**: Positional data can aid in generating Universe maps or examining alignments within HU's hyperspherical model, helping to visualize the cosmic structure.

7. *Sagittarius A Mass and Radius Calculation**:

   SaggitariusAMass = 4.1E6 * cc.M_sun

   SaggitariusARadius = (3 / 4 / np.pi * (SaggitariusAMass / dbh)**(1/3))

   SaggitariusARadius.to(uu.km)

   # Black Hole Density = 5.8E18 kg/m3

   # SaggitariusARadius = 267 km

- **Purpose**: Calculates the radius of Sagittarius A*, the Milky Way's central black hole, based on an assumed density (`dbh`) and known mass.
- **HU Context**: Calculating Sagittarius A*'s physical parameters aligns with HU's exploration of black hole dynamics and gravitational effects within a 4D spatial model.

Summary and Initial Documentation for `AAA_Final_BigPopPaperFigures.ipynb`
Overview
This notebook is dedicated to generating figures for the *Big Pop Cosmogenesis* paper. It includes calculations and plots related to the HU cosmological model, utilizing functions and data from various scientific libraries. Key topics appear to be cosmological parameters, phase transitions, and temperature relationships over cosmic time.

Notebook Purpose
The notebook's primary purpose is to produce visualizations that represent theoretical aspects of HU's Big Pop Cosmogenesis, such as temperature progression, energy density, and cosmological phase transitions, for inclusion in the associated research paper.

Cells and Their Purpose
1. **Content Header**:

   # CONTENT

   ## Creation of all figures for the Big Pop Cosmogenesis Paper

   - **Purpose**: A markdown cell indicating the notebook's goal to create all figures required for the *Big Pop Cosmogenesis* paper.
   - **HU Context**: Sets up the objective to visualize HU cosmogenesis, illustrating critical parameters and transitions in the universe's evolution as described by HU.

2. **Mathematical Equations for Cosmogenesis**:

   $$ \\frac{{P\\left({n,{\\text{ }}x} \\right)}}{{{n_0}{\\text{ }}{T_0}{\\text{ }}}}{\\text{ }} = {\\text{ }}\\frac{x}{{{T_0}}}\\left({Mp + Me - Mn} \\right){\\left({\\frac{n}{{{n_0}}}} \\right)^2} + \\frac{2}{5}[{x^{5/3}} + {\\text{ }}{(1 - x)^{5/3}}]{(\\frac{{2n}}{{{n_0}}})^{2/3}} - [(2\\alpha - 4{\\alpha _L})x(1 - x){\\text{ }} + {\\alpha _L}]{\\left({\\frac{n}{{{n_0}}}} \\right)^2} + {\\text{ }}\\gamma [(2\\eta - 4{\\eta _L})x(1 - x){\\text{ }} + {\\eta _L}]{(\\frac{n}{{{n_0}}})^{\\gamma + 1}} $$

   ...

- **Purpose**: Presents complex equations related to pressure, energy density, and temperature (`T_0`) as functions of density and other HU parameters.
- **HU Context**: These equations encapsulate HU's approach to cosmological evolution, with parameters like `alpha`, `gamma`, and `eta`, specific to HU's model of particle interactions and cosmic phase transitions.

3. **Imports and Plotting Setup**:

   %matplotlib inline

   import matplotlib.pyplot as plt

   import numpy as np

   import scipy as sp

   from parameters import *

   ...

   - **Purpose**: Sets up essential libraries for plotting (`matplotlib`, `scipy`, and `numpy`), imports parameters, and adjusts data display formats.
   - **HU Context**: These setups facilitate the generation of detailed plots depicting HU's theoretical values and trends, such as the speed of sound in neutronium and density variations over time.

4. **Initializing Universe Model for Calculations**:

   myU = Universe(eta, alpha, alpha_L, eta_L, T0, gamma, n0, vssquaredpd)

   - **Purpose**: Initializes an instance of the `Universe` class with specific HU parameters, providing a foundation for generating values and figures representing the cosmological model.
   - **HU Context**: The initialized `Universe` object encapsulates HU-specific cosmological constants, allowing for precise simulations of cosmic evolution based on HU Theory.

5. **Key Parameter Extraction and Optimization**:

   myU.k0

   myU.find_k0([1.22146774e+00, 1.33334958e+00, 2.77144922e-08])

- **Purpose**: Uses functions within the `Universe` class to extract or optimize key parameters, such as `k0`, which may represent a characteristic wave number or frequency within HU cosmogenesis.
- **HU Context**: Parameter optimization is likely aimed at fine-tuning the model to reflect accurate cosmic data, aiding in the visual representation of HU predictions for Big Pop Cosmogenesis.

6. **Temperature and Redshift Calculations**:

   myU.z_transparency, 3443/2.734, 3443/2.725

   - **Purpose**: Calculates temperature and redshift values, linking current temperature to that during transparency (e.g., Cosmic Microwave Background).
   - **HU Context**: These values are critical for mapping HU's cosmological timeline, showing temperature progression and phase transitions.

Detailed Documentation for AAA_Final_BigPopPaperFigures.ipynb
Overview
This notebook serves as a figure-generation pipeline for the *Big Pop Cosmogenesis* paper. It computes and visualizes key cosmological parameters and relationships central to the Hypergeometrical Universe (HU) Theory, particularly those describing the Big Pop model of cosmogenesis. This model differs from traditional Big Bang cosmology by describing the universe's expansion within a higher-dimensional, lightspeed-expanding hyperspherical topology. Figures produced in this notebook visually demonstrate the progression of the universe's physical characteristics, including temperature, density, and pressure, as functions of cosmological parameters.

Notebook Purpose
The notebook's primary goal is to support the *Big Pop Cosmogenesis* paper by creating comprehensive visualizations that elucidate how specific HU parameters influence cosmic evolution. This involves calculating and plotting quantities like phase transition points, energy densities, and the expansion rate, thus providing an empirical basis for the HU cosmological model.

Cells and Their Purpose

1. **Content Header and Objective Setting**

   # CONTENT

   ## Creation of all figures for the Big Pop Cosmogenesis Paper

- **Purpose**: Introduces the notebook's goal to generate figures that illustrate the core components of HU's Big Pop Cosmogenesis model.
- **HU Context**: Establishes the intent to visualize the HU universe's evolution, capturing its transitions and physical state changes over cosmic time.

2. **Mathematical Equations for Cosmogenesis in HU Theory**

$$ \\frac{{P\\left({n,{\\text{ }}x} \\right)}}{{{n_0}{\\text{ }}{T_0}{\\text{ }}}}{\\text{ }} = ... $$

$$ \\frac{{\\varepsilon (n,x)}}{{{n T_0}}} = ... $$

$$ {T_0}{\\text{ }} = {\\text{ }}{\\left({\\frac{{3{\\pi ^{\\text{2}}}}{n_0}}}{2}} \\right)^{\\frac{2}{3}}}\\frac{{{{\\hbar ^2}}}{{2m}} $$

- **Purpose**: Displays key equations for pressure (`P(n,x)`), energy density (`\\varepsilon(n,x)`), and temperature (`T_0`) as functions of density (`n`), mixing ratio (`x`), and other parameters.
- **HU Context**: These equations model the universe's macroscopic properties within the HU framework. The parameters represent energy and density contributions at each cosmological phase, allowing the Big Pop model to capture dynamic shifts in the universe's state, such as phase transitions at critical densities.

3. **Libraries, Constants, and Plotting Configuration**

```
%matplotlib inline

import matplotlib.pyplot as plt

import numpy as np

import scipy as sp

from parameters import *

...
```

- **Purpose**: Imports essential libraries (`matplotlib` for plotting, `scipy` for scientific computations, and `astropy` for cosmological constants) and configures display settings.
- **HU Context**: These libraries support computations of HU model parameters and produce high-quality plots. The imported `parameters` module likely includes specific HU constants and equations necessary for accurate model predictions.

4. **Defining and Initializing the HU `Universe` Model for Cosmogenesis**

   myU = Universe(eta, alpha, alpha_L, eta_L, T0, gamma, n0, vssquaredpd)

   - **Purpose**: Creates an instance of the `Universe` class, which encapsulates HU parameters. This `myU` object represents the HU universe in a computational format, allowing for simulations and calculations based on HU-defined cosmological parameters.
   - **HU Context**: The `Universe` class uses HU-specific constants (e.g., `eta`, `alpha`, `gamma`) and physical attributes like `n0` (reference density) and `T0` (reference temperature). This setup provides the foundation for deriving various cosmological attributes within the HU context, supporting HU's 4D spatial framework for universe expansion.

5. **Extraction and Optimization of Key HU Parameters**

   myU.k0

   myU.find_k0([1.22146774e+00, 1.33334958e+00, 2.77144922e-08])

   - **Purpose**: Retrieves `k0`, a critical wave vector within HU cosmology, and performs optimization to obtain parameter values that best fit observational data.
   - **HU Context**: In HU Theory, `k0` could represent a fundamental frequency or wave vector related to metric fluctuations or dilation field propagation. Optimizing this parameter aligns the model's predictions with current cosmic observations, refining HU's mapping of the universe's spatial dynamics.

6. **Temperature and Redshift Calculations for Transparency Epoch**

   myU.z_transparency, 3443/2.734, 3443/2.725

   - **Purpose**: Computes the universe's temperature and redshift at the transparency epoch, a crucial point in cosmic evolution marked by the decoupling of matter and radiation (linked to CMB formation).
   - **HU Context**: Transparency epoch calculations are essential for situating HU's cosmological timeline alongside observable data (e.g., CMB temperature and redshift). By calculating these values, the HU model can align its theoretical universe age, density, and temperature predictions with known cosmological events, offering empirical grounding for the Big Pop cosmogenesis model.

Summary and Initial Documentation for
`AAA_Final_CMB_Modeling_UniverseMap.ipynb`
Overview

This notebook appears to handle the creation and manipulation of a cosmic microwave background (CMB) model and a 3D map of the universe. Using a 4D hyperspherical model, the notebook uses simulated data from the Planck satellite (specifically the SMICA map) to test and refine HU's interpretation of the CMB and the overall universe topology.

Notebook Purpose
The primary objective of `AAA_Final_CMB_Modeling_UniverseMap.ipynb` is to generate maps and visual representations of the universe based on the HU model, specifically focusing on its 4D hyperspherical hypersurface structure. These visualizations compare HU's universe map with empirical CMB data to optimize fit and refine the representation of cosmological features.

Cells and Their Purpose

1. **Disabling Swap Space (Initialization)**:

   !pkexec sudo swapoff -a

   - **Purpose**: Disables swap space on the system, likely to improve performance during heavy computation or manage memory usage.
   - **HU Context**: Ensures optimal performance for processing large datasets or running complex algorithms, such as 3D and 4D mapping of cosmic structures.

2. **Imports and Initial Configuration**:

   from lib4 import *

   import numpy as np

   from scipy.optimize import curve_fit

   mypath = "./PG_data"

   x0 = [4.93231069, 4.97130803, 0.85524497] # interrupted optimized value 03/27/2024

   - **Purpose**: Imports libraries and configuration settings, including initial guesses (`x0`) for parameters related to HU's cosmic mapping.
   - **HU Context**: The imported library `lib4` likely contains functions specific to HU's modeling requirements, such as metric wave calculations. The initial configuration parameters are used in fitting or mapping functions within the 4D hypersphere.

3. **Task Configuration List**:

   todo = [

   # List of tasks to perform, including:

Color.EVALUATE_DF_AT_POSITION, Color.OPTIMIZE_SPECTRUM, etc.

]

- **Purpose**: Defines tasks to control specific operations, like creating histograms, optimizing spectrum fit, and generating universe maps.
- **HU Context**: Each task corresponds to an action relevant to HU's model refinement and visual output, where `Color` attributes facilitate customizing representations of CMB and 3D universe maps.

4. **Neighborhood Finding and Mapping**:

   if Color.FINDNEIGHBORHOOD in todo:

   # Searches optimal neighborhood parameters and creates maps

   - **Purpose**: Searches for optimal parameters within a local neighborhood in the 4D space, adjusting resolution and coordinates to achieve the best map fit.
   - **HU Context**: Locates the best-fitting 3D projections within the 4D hyperspherical framework, refining HU's CMB-based universe map representation.

5. **Position Minimization**:

   if Color.MINIMIZEPOSITION in todo:

   # Minimizes positional parameters to optimize map fit

   - **Purpose**: Runs optimization (e.g., Nelder-Mead) to minimize error in the positional coordinates (`lambda_k`, `lambda_l`, `lambda_m`).
   - **HU Context**: Fine-tunes HU parameters to improve alignment with CMB data, providing a closer match to observational features within the hyperspherical model.

6. **Evaluation of Density Functions at a Given Position**:

   if Color.EVALUATE_DF_AT_POSITION in todo:

   # Evaluates density functions at a specified hyperspherical position

   - **Purpose**: Calculates density functions based on optimized positional data to test spatial structure consistency within the hyperspherical model.
   - **HU Context**: Evaluates how well HU's theoretical density functions align with observed CMB data at various positions, validating spatial assumptions in HU.

7. **High-Resolution Background Plot Generation**:

if Color.CREATE_HIGH_RESOL_BACKGROUNDPLOT in todo:

Generates high-resolution background plots and difference maps

- **Purpose**: Creates high-resolution CMB background maps for detailed analysis, potentially contrasting SMICA data with HU projections.
- **HU Context**: Produces comparative visuals to assess HU model predictions at a fine resolution, essential for validating model accuracy against observed CMB data.

8. **Histogram Creation**:

if Color.CREATE_HISTOGRAM in todo:

Plots histograms for amplitude distributions in the model

- **Purpose**: Generates histograms for amplitude data, analyzing frequency distribution within HU model projections.
- **HU Context**: Histogram data supports analysis of amplitude distributions predicted by HU, illustrating how mode energies are distributed across the 4D hypersphere.

9. **Spectrum Characterization**:

if Color.CHARACTERIZE_SPECTRUM in todo:

Characterizes spectral data based on HU's model parameters

- **Purpose**: Analyzes amplitude, energy, and standard deviation for different modes (k) in the HU projection, comparing predicted and observed values.
- **HU Context**: Characterizing spectrum data provides insights into mode-specific behaviors and supports energy distribution analysis under HU's hypothesis of hyperspherical mode amplitude scaling.

10. **Universe Map Creation**:

if Color.CREATEMAPOFUNIVERSE in todo:

Generates a 3D universe map for various radius values

- **Purpose**: Constructs a 3D map of the universe at different radius intervals within the hypersphere, creating visualization snapshots for different universe layers.
- **HU Context**: These maps showcase HU's hyperspherical universe, enabling visualization of cosmic structures as projected from different radii along the hyperspherical surface.

Summary and Initial Documentation for `AAA_Final_Creating a circular wave for articles.ipynb`

Overview

This notebook appears to generate visualizations of circular waves for potential inclusion in articles. The images created are focused on illustrating wave propagation and expansion in two or three dimensions, using color maps and various mathematical functions.

Notebook Purpose

The purpose of `AAA_Final_Creating a circular wave for articles.ipynb` is to create aesthetically and scientifically meaningful images of circular waves. These images are likely intended to represent aspects of HU theory, particularly how waves might propagate or form within a hyperspherical model. The notebook includes configurations for both 2D and 3D wave plots.

Cells and Their Purpose

1. **Imports and Plot Setup**:

   import matplotlib.pyplot as plt

   import numpy as np

   - **Purpose**: Imports `matplotlib` and `numpy` for plotting and data manipulation.
   - **HU Context**: These libraries enable precise control over wave shapes and colors, aligning with HU's conceptualization of wave dynamics within its hyperspherical model.

2. **2D Circular Wave Creation**:

   edge = 6

   uni = edge + 1

   n = 500

   r = np.linspace(0, (edge + 2) * np.pi, n)

   theta = np.linspace(0, 2 * np.pi, n)

   # Initializes arrays for X, Y, and Z coordinates and wave values

   X = np.zeros((n, n))

   Y = np.zeros((n, n))

```
Z = np.zeros((n, n))

randomthetas = -np.random.rand(n)

for i in range(n):

    for j in range(n):

        X[i, j] = r[i] * np.cos(theta[j])

        Y[i, j] = r[i] * np.sin(theta[j])

        darkcenter = (r[i] > edge * np.pi) * (r[i] < (edge + 2) * np.pi)

        universe = (r[i] > uni * np.pi) * (r[i] < (uni + 2) * np.pi) * randomthetas[j]

        Z[i, j] = r[i] * darkcenter * (1 + 2 * universe)

plt.pcolor(X, Y, Z, shading="nearest")

plt.show()

fig1.savefig("./Drawing For Publications/expandinguniverse.jpg")
```

- **Purpose**: Generates a 2D circular wave plot based on polar coordinates (`r` and `theta`), with a darker center and random variations in wave amplitude.
- **HU Context**: The 2D wave likely represents the propagation of metric waves in the HU framework, illustrating wavefronts in a planar projection that could be used to conceptualize spacetime distortions.

3. **3D Circular Wave Plotting**:

```
fig = plt.figure()

fig.set_size_inches(8, 8)

ax = fig.add_subplot(projection='3d')

n = 200

r = np.linspace(0, 10 * np.pi, n)

p = np.linspace(0, 2 * np.pi, n)

R, P = np.meshgrid(r, p)
```

Z = np.sin(R) / (R**2 + 0.01)

X, Y = R * np.cos(P), R * np.sin(P)

ax.plot_surface(X, Y, Z, cmap=plt.cm.Paired)

ax.set_zlim(0, 0.02)

plt.show()

- **Purpose**: Creates a 3D plot of a circular wave using `plot_surface`, where wave amplitude decays with radial distance.
- **HU Context**: The 3D wave demonstrates spatial distribution and decay, representing HU's vision of metric waves that vary as a function of distance within a 4D hypersphere.

4. **Color Map Definitions**:

cmaps = OrderedDict()

cmaps['Perceptually Uniform Sequential'] = ['viridis', 'plasma', 'inferno', 'magma', 'cividis']

Further categories and color maps follow

- **Purpose**: Defines different color map categories, including perceptually uniform and diverging schemes.
- **HU Context**: Selecting effective color maps enhances wave visualization, allowing for clear distinctions in amplitude and phase changes, which could aid in representing the complex wave phenomena hypothesized by HU.

5. **Color Gradient Visualization**:

def plot_color_gradients(cmap_category, cmap_list):

    # Plots color gradients for a list of color maps

    ...

for cmap_category, cmap_list in cmaps.items():

    plot_color_gradients(cmap_category, cmap_list)

plt.show()

- **Purpose**: Visualizes color gradients for each defined colormap, aiding in the selection of color schemes for wave representations.
- **HU Context**: Choosing optimal color gradients ensures that wavefronts and amplitude variations are visually intuitive, supporting accurate interpretations of wave interactions in the HU framework.

Summary and Initial Documentation for
`AAA_Final_Derivation_of_Laws-3D.ipynb`
Overview
This notebook presents symbolic derivations related to the laws of motion and force within the Hypergeometrical Universe (HU) framework. The derivations appear to involve transformations, field calculations, and force equations based on metric wave propagation and the interaction of wave functions in three dimensions.

Notebook Purpose
The purpose of `AAA_Final_Derivation_of_Laws-3D.ipynb` is to derive expressions for fields, wave vectors, and forces in a 3D framework that aligns with HU Theory. This involves constructing symbolic representations for key quantities like velocity vectors, Lorentz transformations, and force components. The resulting formulas are intended to provide theoretical foundations for motion and field interactions as described in the HU model.

Cells and Their Purpose

1. **Introduction and Setup**

   # BEGINNING - The Hypergeometrical Universe Theory - Derivation of Laws of Nature

   - **Purpose**: A title cell indicating the start of HU's law derivations, establishing the theoretical context for the subsequent symbolic work.
   - **HU Context**: Introduces the HU framework, aiming to derive motion laws based on metric wave interactions and transformations.

2. **Imports and Symbol Definition**

   ```
   import sympy as sp

   # Symbol definitions for constants and variables

   pi, m_0, x, Q, lambda1, R0, c, N, v1, v2, gamma_v1, gamma_v2, P1, P2 =
   sp.symbols('...')

   V1_hat = sp.MatrixSymbol('V1_hat', 3, 1)

   V2_hat = sp.MatrixSymbol('V2_hat', 3, 1)
   ```

R0_hat = sp.MatrixSymbol('R0_hat', 3, 1)

r_hat = sp.MatrixSymbol('r_hat', 3, 1)

- **Purpose**: Imports `sympy` for symbolic calculations and defines constants, vectors, and matrices used in deriving force and motion laws.
- **HU Context**: Symbols represent constants (e.g., `pi`, `c`) and dynamic quantities (e.g., velocity vectors `V1`, `V2`) relevant to HU's model of 3D interactions within the 4D universe.

3. **Matrix and Vector Definition for Velocity and Position**

Velocity vectors and position vectors in terms of unit vectors

V1 = v1 * V1_hat

V2 = v2 * V2_hat

R0_vect = R0 * R0_hat

Position vectors

r1 = V1 * R0 / c + x * r_hat

r2 = V1 * R0 / c - R0_vect + x * r_hat

- **Purpose**: Defines velocity and position vectors in 3D space using symbolic unit vectors and velocity components.
- **HU Context**: These definitions allow for analyzing interactions of metric waves within a specified 3D space, central to understanding force and motion in HU.

4. **Projection and Lorentz Transformation Matrices**

M1 = One + (gamma_v1 - 1) * V1 * V1.T / (v1**2)

M2 = One + (gamma_v2 - 1) * V2 * V2.T / (v2**2)

- **Purpose**: Constructs projection matrices based on Lorentz transformations, adjusting for relative velocities and direction.
- **HU Context**: These transformations facilitate the modeling of metric wave propagation in HU, accommodating velocity effects that alter wave behavior relative to the observer's frame.

5. **Wave Vector and Field Definitions**

k1 = (2 * pi / lambda1) * r1.T * M1 / P1

k2 = (2 * pi / lambda1) * r2.T * M2 / P2

Phi1 = sp.cos(k1 * r1)

Phi2 = N / (1 + (k2 * r2))

- **Purpose**: Defines wave vectors (`k1`, `k2`) and fields (`Phi1`, `Phi2`) based on position and transformations, with `Phi1` as a cosine wave and `Phi2` an amplitude-modified field.
- **HU Context**: Fields and wave vectors represent metric waves generated by Fundamental Dilators in HU, capturing the oscillatory nature of space in the 4D hyperspherical model.

6. **Derivatives of Fields**

   Phi1_diff = k1path.T * k1path_diff

   Phi2_diff = -N * k2path_diff / (k2path.subs(x, 0))**2

   - **Purpose**: Calculates derivatives of fields (`Phi1`, `Phi2`) with respect to position, essential for determining the rate of change in wave intensity and force direction.
   - **HU Context**: These derivatives are foundational to HU's force derivation, as they quantify how field intensity varies across space due to metric wave dynamics.

7. **Force Derivation**

   # Detailed markdown derivation of force, starting from motion laws and tanh(alpha)

   - **Purpose**: Provides a theoretical derivation for force, incorporating hyperbolic functions (`tanh`) and relativistic velocity corrections.
   - **HU Context**: Establishes the force law within HU's framework, integrating metric wave effects and relativistic adjustments that HU posits for 4D space interactions.

8. **Force Simplification and Output**

   Force = m_0 * c**2 * (1 - v1**2 / c**2) * x_result / lambda1**2 * dr2dR

   - **Purpose**: Derives and simplifies the force formula, outputting a finalized force expression as a function of relevant HU parameters.
   - **HU Context**: The resulting force formula is meant to apply to Fundamental Dilators, representing HU's explanation for particle interactions within the hyperspherical universe model.

9. **Output Display and Formatting**

print(Force)

cleanForce(Force)

- **Purpose**: Prints and formats the derived force expression for readability, preparing it for use in further calculations or theoretical presentations.
- **HU Context**: Finalizes the HU force law expression, allowing it to be applied or referenced in HU-based analyses of physical phenomena.

Summary and Initial Documentation for
`AAA_Final_Derivation_of_Laws-3D_4_Body_1.ipynb`
Overview
This notebook extends the derivation of motion and force laws within the Hypergeometrical Universe (HU) framework to a four-body interaction scenario in a 3D spatial projection of a 4D hyperspherical universe. It involves detailed symbolic calculations for fields, wave vectors, and force derivations, focusing on interactions between multiple bodies (or Fundamental Dilators) in this extended framework.

Notebook Purpose
The purpose of `AAA_Final_Derivation_of_Laws-3D_4_Body_1.ipynb` is to derive laws governing interactions among four bodies within HU's model, using symbolic mathematics. This includes defining the dilaton field, calculating derivatives of the field for two interacting bodies, and deriving force expressions based on wave propagation and transformation matrices. The derivations are structured to handle multiple interactions, reflecting the complex dynamic behaviors HU suggests for metric waves across a 4D spatial manifold.

Cells and Their Purpose

1. **Introduction to Four-Body Interaction Model**

   BEGINNING - The Hypergeometrical Universe Theory - Derivation of Laws of Nature

   ### Interaction Model in a 4D Spatial Manifold

   #### Approximation of the Dilaton Field

   - **Purpose**: Introduces the derivation of laws for interactions among four bodies in HU's framework, outlining steps for defining and differentiating the dilaton field.
   - **HU Context**: Sets the stage for approximating and calculating the dilaton field generated by multiple Fundamental Dilators, representing HU's hypothesized force interactions within the hyperspherical manifold.

2. **Imports and Symbol Definition**

import sympy as sp

Symbol definitions for constants and variables

pi, m_0, x, Q, lambda1, R0, c, N, v1, v2, gamma_v1, gamma_v2, P1, P2, delta, R_4D, P1_hat, P2_hat, G0, kg, h = sp.symbols('...')

- **Purpose**: Imports `sympy` for symbolic calculations and defines variables and constants required for calculating forces and fields among multiple bodies.
- **HU Context**: Symbols represent wave propagation, particle positions, and transformations, critical for modeling complex interactions in the HU's 4D structure.

3. **Vector and Field Definition for Body Interactions**

V1 = v1 * V1_hat

V2 = v2 * V2_hat

r1 = V1 * R0 / c + x * r_hat

r2 = V1 * R0 / c - R0_vect + x * r_hat

- **Purpose**: Defines velocity and position vectors for bodies in the system, incorporating symbolic placeholders for relativistic effects and projections.
- **HU Context**: These vectors form the basis for evaluating field intensities and interactions among the bodies, reflecting HU's particle-wave dynamics in 4D.

4. **Projection and Lorentz Transformation Matrices**

M1 = One + (gamma_v1 - 1) * V1 * V1.T / (v1**2)

M2 = One + (gamma_v2 - 1) * V2 * V2.T / (v2**2)

- **Purpose**: Constructs projection matrices to account for relativistic transformations in the body interactions, modifying wave and field properties based on velocities.
- **HU Context**: Essential for modeling how metric waves and interactions vary with velocity, providing relativistic corrections in the multi-body system.

5. **Dilaton Field Derivatives for Body 1 and Body 2**

Step-by-step calculation of first and second derivatives of Phi fields for bodies 1 and 2

- **Purpose**: Defines and differentiates dilaton fields from each body, using these fields to approximate force interactions.

- **HU Context**: These fields represent metric disturbances generated by each Fundamental Dilator, and the differentiation process captures how field strength and direction change across space.

6. **Force Derivation Among Bodies**

   # Calculation of force using hyperbolic functions for relativistic adjustments

   - **Purpose**: Provides a comprehensive derivation of force between bodies, incorporating `tanh` functions and relativistic terms that are adjusted by hypergeometric constants.
   - **HU Context**: The force derivation aligns with HU's concept of forces as metric waves between bodies, extending the force law to account for interactions in a multi-body setup.

7. **Final Force Expression and Output Formatting**

   ```
   Force = m_0 * c**2 * (1 - v1**2 / c**2) * x_result / lambda1**2 * dr2dR

   print(Force)

   cleanForce(Force)
   ```

   - **Purpose**: Simplifies and formats the final force expression for readability, preparing it for presentation or further use in calculations.
   - **HU Context**: This force expression models HU's interpretation of interactions, allowing predictions on how metric waves propagate and affect each Fundamental Dilator in the 4-body system.

Summary and Initial Documentation for
`AAA_Final_Derivation_of_Laws-3D_4_Body_2.ipynb`
Overview
This notebook continues the derivation of force and motion laws for a 4-body interaction model in the Hypergeometrical Universe (HU) framework. Building upon previous interactions among bodies, this notebook focuses on additional symbolic calculations to capture the dynamics of metric wave interactions and field derivatives for the bodies in question.

Notebook Purpose
The purpose of `AAA_Final_Derivation_of_Laws-3D_4_Body_2.ipynb` is to extend force derivations within HU's 4-body interaction model, involving advanced symbolic calculations to account for field derivatives, projection matrices, and relativistic effects. This notebook derives field contributions and forces from one body to another, continuing the multi-body interaction analysis central to HU's model of cosmic dynamics.

Cells and Their Purpose

1. **Introduction**

   # BEGINNING - The Hypergeometrical Universe Theory - Derivation of Laws of Nature

   - **Purpose**: Introduces the continuation of HU's force law derivation, specifying that the focus is on advanced field derivatives and force computations in a 4-body scenario.
   - **HU Context**: Establishes the theoretical basis for expanding HU's interaction model across multiple Fundamental Dilators (bodies) within the 4D spatial framework.

2. **Imports and Symbol Definition**

   import sympy as sp

   # Define constants, variables, and unit vectors for the interaction model

   pi, m_0, x, Q, lambda1, R0, c, N, v1, v2, gamma_v1, gamma_v2, P1, P2, delta, R_4D, P1_hat, P2_hat, G0, kg, h = sp.symbols('...')

   - **Purpose**: Imports `sympy` for symbolic computation and defines constants, unit vectors, and other variables used in the derivation.
   - **HU Context**: Each symbol corresponds to key parameters in HU's model, including wave properties, relativistic adjustments, and spatial transformations for the 4-body setup.

3. **Vector and Position Definitions**

   V1 = v1 * V1_hat

   V2 = v2 * V2_hat

   r1 = V2 * R0 / c + R0_vect + x * r_hat

   r2 = V2 * R0 / c + x * r_hat

   - **Purpose**: Defines velocity and position vectors for bodies involved in the interaction, including relativistic corrections.
   - **HU Context**: These vectors form the foundation for defining wave fields and derivatives, representing how each body affects the local metric waves within HU's hyperspherical universe.

4. **Projection and Transformation Matrices**

   M1 = One + (gamma_v1 - 1) * V1 * V1.T / (v1**2)

M2 = One + (gamma_v2 - 1) * V2 * V2.T / (v2**2)

- **Purpose**: Constructs transformation matrices using relativistic factors, enabling Lorentz-corrected transformations for the interaction model.
- **HU Context**: These matrices adjust wave propagation properties based on each body's relative velocity, allowing HU to model transformations in 4D space with appropriate relativistic adjustments.

5. **Field Derivative Calculations for Body 2**

   # Step-by-step calculation of derivatives for Phi fields for Body 2 interactions

   - **Purpose**: Calculates first and second derivatives of `Phi2`, representing the field generated by Body 2, to assess field strength and direction.
   - **HU Context**: These derivatives quantify how field intensity changes across the 3D projection of the 4D space, offering insights into the metric wave contributions from Body 2.

6. **Force Derivation and Calculations**

   # Derivation of force expressions using hyperbolic functions for relativistic terms

   - **Purpose**: Continues the force derivation process by incorporating relativistic effects, hyperbolic functions, and field alignment.
   - **HU Context**: The derived force expressions capture the interactions between bodies through metric wave propagation, adjusted for relativistic velocities. This is central to HU's interpretation of gravitational-like forces as wave interactions within 4D space.

7. **Final Force Expression and Output**

   Force = m_0 * c**2 * (1 - v2**2 / c**2) * x_result / lambda1**2 * dr2dR

   print(Force)

   cleanForce(Force)

   - **Purpose**: Derives and simplifies the final force expression, then formats it for readability and output.
   - **HU Context**: This final force formula encapsulates HU's predictions for force interactions within a 4-body system, where wave dynamics and relativistic corrections define interaction intensities and directions.

This documentation provides detailed insights into each cell's role in deriving complex force interactions among bodies in the HU model, with emphasis on symbolic differentiation and relativistic corrections for multi-body systems.

Let's begin with the first notebook,
`AAA_Final_DimensionalityProbDistribution.ipynb`.

The `AAA_Final_DimensionalityProbDistribution.ipynb` notebook focuses on calculating the surface area of a hypersphere with a unit radius across a range of dimensions (from 3 to 20). This visualization and calculation serve as an exploration of how hyperspherical geometry evolves with dimensionality—a core concept in the Hypergeometrical Universe Theory (HU) to understand probability distributions across higher-dimensional spaces.

Notebook Overview

1. **Calculation of Hyperspherical Surface Area**:

    - The notebook uses the formula for the surface area of a hypersphere in (n)-dimensional space: [S = \frac{2 \pi^{n/2}}{\Gamma(n/2)}] where (\Gamma) is the gamma function.
    - The surface areas are calculated for dimensions ranging from 3 to 20, highlighting how the surface area behavior changes with increasing dimensionality.

2. **Visualization**:

    - A plot is generated to show the surface area of a unitary radius hypersphere against dimensionality.
    - This provides an insight into how space "stretches" or "contracts" across dimensions, possibly relevant to understanding the spatial constraints of particles or metric waves in higher-dimensional HU contexts.

Purpose within HU Theory
In the context of HU, this analysis may be used to illustrate the constraints or distribution of metric waves (or other spatial phenomena) as they traverse or exist within a multidimensional space. The probability distribution associated with these surface areas could be tied to the distribution of forces or fields in a hyperspherical geometry.

The `AAA_Final_FermiParadox.ipynb` notebook addresses the Fermi Paradox through the lens of an epoch-dependent gravitational constant, (G), based on the Hypergeometrical Universe Theory (HU). This approach suggests that the gravitational constant varies over cosmic time, impacting the conditions for potential life-supporting environments across epochs.

Notebook Overview

1. **Fermi Paradox and Gravitational Decay**:

    - The title and code suggest an exploration of how the decay of (G) over the universe's age could influence the evolution of potentially habitable planets.
    - Specifically, it seems to model how a decreasing (G) affects various epochs, potentially explaining why we have not observed extraterrestrial civilizations.

2. **Visualization of (G) Decay**:

    - The notebook includes a plot showing (G/G_0) (normalized gravitational constant) as a function of the universe's age. Multiple initial values of (G) over time ranges (e.g., 2 to 10 billion years) are plotted to observe the decay trend.
    - This decay could theoretically explain periods where gravitational conditions would either favor or hinder the development of advanced civilizations.

3. **Parameters and Usage**:

    - The code refers to an external parameters file (`parameters.py`) and an image save location (`imgAddress`). These likely provide configuration for constants and file paths, useful for further analysis or presentations.

Purpose within HU Theory

In HU, the concept of a time-varying (G) aligns with the theory's stance on epoch-dependent forces and the dynamic nature of cosmic conditions. This notebook's approach may suggest that early epochs with higher (G) values were unsuitable for life due to rapid gravitational evolution, while later, more stable epochs allow for civilizations to emerge—a novel explanation for the Fermi Paradox.

The `AAA_Final_FrameDragging_BlackHole_Sun.ipynb` notebook explores frame-dragging effects—specifically, the Lense-Thirring precession—caused by the Sun and by a hypothetical supermassive black hole. The calculations estimate the impact of the rotation of these massive objects on the orbit of Mercury, which serves as a probe to study relativistic effects near strong gravitational fields.

Notebook Overview

1. **Frame Dragging by the Sun**:

    - The notebook calculates the Lense-Thirring precession effect due to the Sun's rotation. This effect causes a small perturbation in Mercury's orbit, resulting in a shift of its perihelion over time.
    - Parameters such as the Sun's angular momentum and Mercury's orbital elements (semi-major axis and eccentricity) are used to quantify this precession in terms of arcseconds per century.

2. **Frame Dragging by a Black Hole**:

    - A similar calculation is performed for a hypothetical supermassive black hole with a mass of one billion solar masses.
    - This scenario is theoretical and aims to illustrate how a significantly larger mass and rotation rate would amplify the frame-dragging effect, yielding much higher precession values.
    - The black hole's angular momentum is estimated based on a simplified moment of inertia model, acknowledging the hypothetical nature of the assumption for educational purposes.

3. **Output Interpretation**:

    - The results are given in terms of arcseconds per century for the Sun and in rotations per year for the black hole case, highlighting the dramatic difference in frame-dragging effects between the two objects.

Purpose within HU Theory
In the context of the Hypergeometrical Universe Theory (HU), examining frame-dragging effects could relate to how metric deformations propagate in the 4D spatial manifold. Since HU aims to unify gravitational and electromagnetic effects through a Van der Waals-like force, understanding frame-dragging might help elucidate the interaction of fundamental dilators with massive rotating bodies, which could affect the propagation of metric waves near such objects.

The `AAA_Final_GravitationalLensing_HU.ipynb` notebook examines gravitational lensing from a Hypergeometrical Universe (HU) perspective. The notebook compares gravitational lensing calculations using both Newtonian mechanics and modified equations aligned with HU principles, aiming to capture how light (or particles) deflects around massive objects.

Notebook Overview
1. **Calculation Setup**:

    - Constants such as the gravitational constant (G), speed of light (c), and the Sun's mass are defined. The calculations also set up an initial hyperbolic trajectory around a central mass (e.g., the Sun) with adjustable parameters.
    - The notebook defines functions for calculating the trajectory (`r` and its derivatives) based on HU modifications and Newtonian gravity, adjusting for eccentricity, semi-major axis, and other orbital parameters.

2. **Deflection Angle Optimization**:

- The notebook uses numerical integration and optimization to compute deflection angles, simulating gravitational lensing effects.
- Separate error functions (`error_functionHU` and `error_functionNewton`) are provided to evaluate the deviation of the observed gravitational acceleration from theoretical values in HU and Newtonian frameworks.
- Optimization functions minimize these errors to find the best fit for gravitational deflection angles under both HU and traditional assumptions, revealing any differences in the predicted paths.

3. **Visualization of Trajectories**:

   - The notebook includes plotting functions that visualize the hyperbolic trajectory around a central mass, illustrating how the deflection angle varies based on the selected theoretical model.
   - Plots show HU and Newtonian acceleration values to highlight the differences in predictions between the two models, with HU theory potentially modifying the lensing behavior.

Purpose within HU Theory
In HU, gravitational lensing reflects the effects of a Van der Waals-type force rather than traditional spacetime curvature. This notebook's calculations and plots serve to compare the classical gravitational lensing predictions against HU-based modifications, potentially revealing nuanced differences in how metric waves propagate near massive objects. By examining deflection angles under both approaches, this notebook provides insights into HU's distinctive predictions for gravitational lensing phenomena.

The `AAA_Final_Gravitational_Light_Deflection_ClassicalMechanics.ipynb` notebook uses classical mechanics principles to calculate the deflection angle of light (or photons) as it passes close to a massive body, like the Sun. This approach contrasts with the relativistic predictions of general relativity, offering an alternative calculation for gravitational light deflection based on potential energy integrals.

Notebook Overview

1. **Classical Mechanics Setup**:

   - The notebook defines symbols and constants for gravitational constant (G), the Sun's mass (M), photon mass (set arbitrarily for calculation), speed of light (c), and the impact parameter (b), corresponding to the Sun's radius.
   - An expression for the differential of potential energy in terms of distance is set up to model the gravitational effect on the photon's trajectory.

2. **Integral Calculation**:

- The integral for gravitational potential energy is evaluated over an infinite range, representing the entire path of light passing by the Sun. The result approximates the net gravitational influence along this trajectory in a classical mechanics framework.
- This integral provides a baseline for comparing energy contributions, contrasting with general relativity's more complex geodesic calculations.

3. **Deflection Angle**:

   - The deflection angle is calculated as a ratio of the change in potential energy ((\Delta P)) to a maximum possible photon energy, offering an alternative angle of light deflection around massive bodies.
   - The result is converted to arcseconds, allowing comparison to empirical values and the relativistic predictions for solar light bending.

Purpose within HU Theory
This notebook exemplifies HU's perspective on gravitational light deflection using classical mechanics, which HU may view as a byproduct of metric wave interactions in the 4D manifold rather than spacetime curvature. The calculation potentially provides insights into whether classical mechanics alone, under HU assumptions, can account for observed light deflection angles without invoking general relativity's spacetime distortion.

The `AAA_Final_HyperbolaMaterialBody_HU.ipynb` notebook focuses on modeling the trajectory of a material body on a hyperbolic path around a massive object, likely intended as an approximation for studying gravitational interactions in the context of the Hypergeometrical Universe (HU). This includes calculating trajectories, angular velocities, and the deflection angle within a modified gravitational framework inspired by HU.

Notebook Overview
1. **Hyperbolic Trajectory Setup**:

   - Constants for gravitational interactions (e.g., (G), (M), (c)) are defined along with initial parameters for the hyperbolic trajectory.
   - Functions `calc_ab` and `calc_ae` compute the semi-major axis (a) and eccentricity (e) for a hyperbolic orbit based on initial velocity and gravitational constants.

2. **Derivation of Radial and Angular Velocities**:

   - The notebook defines functions that calculate derivatives (r), (\dot{r}), and (\ddot{r}) based on theta ((\theta)) to model the radial component of motion.
   - Another function calculates angular velocities and accelerations, taking into account HU adjustments for trajectories under non-relativistic assumptions.

3. **Optimization of Trajectory**:

    - Using optimization techniques, the notebook adjusts the parameters to minimize the deviation from expected results under HU and Newtonian gravity.
    - Functions `error_functionHU` and `error_functionNewton` evaluate the fit of the calculated trajectory under each model, iteratively optimizing for the best representation of the hyperbolic path.

4. **Wave Summation**:

    - Additional symbolic calculations explore summing two wave functions, presumably for analyzing HU's concept of metric wave interference or other wave interactions around a massive object.
    - This component could be examining wave interference effects as part of the trajectory, aligning with HU's use of metric waves as the basis for gravitational interactions.

5. **Simulation of Orbital Motion**:

    - The notebook sets up a system of differential equations to simulate photon or particle motion around a central body, solving for trajectory and angular dynamics over time.
    - This final section aims to visualize the material body's hyperbolic path in relation to gravitational forces, showing how it is affected by parameters set in HU.

Purpose within HU Theory
This notebook applies HU principles to analyze hyperbolic orbits, contrasting HU-based predictions against classical mechanics. It captures how HU's wave-based modifications impact orbital paths, providing a platform for further exploration of gravitational lensing or deflection effects under HU's alternative framework.

The `AAA_Final_Initial4DRadius.ipynb` notebook derives the initial 4D radius of the universe based on energy conservation principles, density calculations, and HU assumptions about the universe's structure and initial conditions.

Notebook Overview

1. **Density and Initial Radius Derivation**:

    - The notebook begins with a step-by-step derivation relating the initial radius (R_{\text{initial}}) to the speed of light, gravitational constant (G), and initial density.

- The energy conservation equation incorporates the gravitational potential energy and kinetic energy of mass in the universe's initial contraction layer, yielding an expression for (\rho_{\text{current}}), the density as a function of the initial radius: [\rho_{\text{current}} = \frac{c^2}{G \cdot 4 \pi R_{\text{current}}^2}]
- This density is then used to determine the total mass (\text{TotalMass}) of the universe and solve for the initial radius.

2. **Calculations Using Physical Constants**:

   - Using `astropy` constants for parameters such as the mass of fundamental particles (proton and electron), speed of light, and Planck's constant, the notebook calculates various densities, including those of black holes and neutron stars, and the initial density of the universe.
   - The initial radius is computed to be approximately 525 light-seconds, which gives an idea of the universe's scale at the beginning of expansion according to HU.

3. **Outputs and Interpretation**:

   - Key calculated values include cell length, black hole density, neutron star density, and total mass of the universe.
   - The final computed initial 4D radius, displayed in light-seconds, aligns with HU's framework of a lightspeed expanding hyperspherical universe (LEHU), representing an early cosmological scale that is foundational to the theory.

Purpose within HU Theory
This notebook attempts to model the universe's initial conditions, using HU's framework to define its scale, density, and structure at the point of origin. The initial radius provides a baseline for understanding subsequent metric expansion in HU, connecting the early density and mass distribution to the observable universe's evolution.

The `AAA_Final_MercuryPerihelionCalculation_HU.ipynb` notebook investigates the perihelion precession of Mercury's orbit using the Hypergeometrical Universe (HU) theory. This approach calculates and compares the predicted precession rates under HU and general relativity (GR), assessing how HU's modifications might explain observed deviations in orbital precession.

Notebook Overview
1. **Deriving Orbital Radius and Velocity**:

   - Using symbolic calculations, the notebook defines the radial distance (r(\theta)), its derivatives, and angular velocities as functions of eccentricity (e), gravitational parameter (GM), and angular momentum (h). These expressions are then used to compute the theoretical acceleration under HU.

- The notebook derives (\theta)-dependent values such as (\dot{r}) and (\ddot{r}) for evaluating perihelion precession.

2. **Error Function and Optimization**:

   - Functions `error_functionHU` and `HU_Accel` define the error in predicted acceleration under HU by comparing it with numerical results derived from orbital mechanics.
   - These functions are optimized using `scipy` to find the best values of parameters that minimize the deviation from HU's expected results, improving accuracy in calculating the precession rate.

3. **Calculating Precession Rates**:

   - The notebook computes precession rates per century for various planets, with Mercury as the primary focus.
   - `calculatePrecession` function calculates perihelion precession rates in HU and GR, allowing comparisons with observed values. Results are presented in arcseconds per century.

4. **Visualization of Precession Comparisons**:

   - Bar charts compare HU and GR predictions with observational data for each planet. Error bars provide a measure of uncertainty in observed values.
   - This visualization highlights how closely HU predictions align with GR and observations, particularly in the case of Mercury.

5. **Summary of Observed and Calculated Data**:

   - The notebook includes several dataframes with planetary parameters, observed and theoretical precession rates, and relativistic contributions.
   - This tabulation offers a comprehensive view of the performance of HU and GR theories against empirical data, demonstrating where HU provides comparable or divergent predictions.

Purpose within HU Theory
This notebook extends HU's applications to planetary motion by predicting perihelion precession, a classical test of relativistic theories. HU's perihelion precession calculation may suggest an alternative approach to gravity that aligns with general relativity for small corrections but diverges in significant ways in its interpretation of underlying mechanisms.

The `AAA_Final_OpticalPathOfAncientPhotons.ipynb` notebook explores the redshift and path of photons as they propagate through the universe, using a unique approach

based on the Hypergeometrical Universe (HU) framework. This approach models how ancient photons' wavelengths and trajectories are influenced by the expanding 4D manifold, providing visual insights into wavelength stretching (redshift) over cosmic time.

Notebook Overview

1. **Cosmological Distance and Luminosity Calculations**:

    - The notebook uses `astropy`'s cosmology package to calculate luminosity distances based on Planck 2015 data for redshift values up to 1.5. This distance scale is later applied in HU-specific calculations for comparison.
    - Observational data for supernovae distances is imported to further correlate redshift with the HU model, particularly for high redshift values (e.g., (z = 1.917)).

2. **Defining HU-Specific Functions**:

    - Functions like `alpha(z)`, `RofT(z)`, and `DistanceOfT(z)` calculate the cosmological angle, 4D radius of the universe, and distance to specific epochs, respectively.
    - These functions represent HU's model of distance scaling and angle-based metrics for understanding redshift and cosmic structure.

3. **Plotting Ancient Photon Paths**:

    - The `plotAncientPhotons` function visualizes photon trajectories by generating concentric circles and rays, symbolizing the expansion and travel path of photons in 4D space.
    - This visualization demonstrates how photon paths are bent or shifted over time due to the universe's geometry, illustrating redshift effects on photons as they move through the universe.

4. **Projection of Light Propagation in 4D**:

    - Multiple sinusoidal plots and projections illustrate the wavelength stretching phenomenon by projecting a 4D sinusoidal wavelength onto a 3D observable wavelength. Rotational matrices are used to model the effects of light propagation through different angles.
    - The notebook's plots show the gradual change in photon wavelengths (interpreted as redshift) as they travel through the 4D space and project onto a 3D observer's view.

5. **Rotation Matrix and Redshift Animation**:

    - Using rotation matrices, the notebook creates a series of visualizations for various angles to show how different redshifts correspond to different projection angles.

This yields a sequence of images that animate the concept of redshift as seen in 3D.
- These frames are saved for visualization as an animation, helping to clarify HU's interpretation of how light wavelength changes over cosmic time.

Purpose within HU Theory
This notebook provides a cosmological view of redshift under HU, showing how ancient photons travel and experience wavelength stretching due to the 4D expansion. The projections from 4D to 3D and the modeling of redshift add depth to HU's explanation for cosmic redshift without assuming traditional relativistic frameworks.

The `AAA_Final_SN1a.ipynb` notebook focuses on analyzing Type Ia supernovae (SN1a) data, which is used as a standard candle for measuring cosmic distances. This analysis explores the relationship between redshift and luminosity distance within the Hypergeometrical Universe (HU) framework. By comparing HU-based calculations with observational SN1a data, the notebook aims to evaluate how well HU predictions align with known cosmological observations, particularly regarding the expansion rate of the universe.

Notebook Overview
1. **Data Loading and Preparation**:
    - Observational data for SN1a is imported, providing redshift and luminosity distance measurements for various supernovae events.
    - The data is cleaned and prepared for HU model fitting, focusing on observed luminosity distances versus theoretical predictions.

2. **Cosmological Distance Calculation**:
    - Using HU principles, the notebook defines functions to calculate theoretical luminosity distances at different redshifts. This approach assumes a lightspeed-expanding hyperspherical universe, which introduces unique distance scaling based on the 4D radius and expansion mechanics specific to HU.
    - Functions like `calculateLuminosityDistance_HU` compute distances across redshift values, allowing for HU's distinct model to be compared against standard cosmological predictions.

3. **Curve Fitting and Model Optimization**:
    - The notebook employs curve-fitting techniques to match HU predictions with observational SN1a data.
    - By adjusting parameters related to the expansion and 4D curvature, the notebook finds the best fit for HU distances, analyzing residuals to assess fit quality.

4. **Plotting Results and Comparisons**:

    - Various plots display observational data points alongside HU-based predictions, highlighting the model's alignment with observed SN1a distances across a wide range of redshifts.
    - Comparative plots with traditional cosmological models, such as ΛCDM, provide insight into HU's performance relative to standard theories, particularly for explaining late-time cosmic acceleration without invoking dark energy.

5. **Error Analysis**:

    - An error analysis section evaluates the residuals and differences between HU predictions and observational data. This quantifies how well the HU framework explains SN1a-based distance measurements compared to established cosmological models.

Purpose within HU Theory
This notebook applies HU's alternative cosmology to SN1a data, attempting to reconcile observed cosmic expansion with HU's unique principles. By fitting HU's predictions to SN1a data, the notebook investigates whether HU can account for cosmic acceleration without dark energy, testing its validity against one of modern cosmology's critical datasets.

The `AAA_Final_SN1a-DualOptimization.ipynb` notebook is a continuation and expansion of the analysis on Type Ia supernovae (SN1a) data, with a focus on optimizing the fit of Hypergeometrical Universe (HU) model parameters to SN1a observational data using a dual optimization approach. This analysis further investigates how HU's model can explain cosmic expansion patterns without dark energy by refining parameters across two dimensions of optimization.

Notebook Overview

1. **Data Import and Preparation**:

    - Similar to the previous SN1a notebook, this notebook loads observational SN1a data, preparing it for model fitting by ensuring data accuracy and compatibility with HU's distance-redshift relations.
    - The data includes luminosity distances at various redshifts, which serve as standard candles for measuring cosmological distances.

2. **Dual Optimization Setup**:

    - This notebook introduces a dual optimization process, refining parameters across two aspects of HU's theoretical framework to improve model fit.

- Two optimization functions are defined to minimize residuals between HU-predicted luminosity distances and observed SN1a data, providing a more precise calibration of HU's parameters.

3. **Model Fitting and Parameter Tuning**:

   - The dual optimization routine iterates over selected HU parameters, updating them to minimize deviations between the model and observational data. This process aims to achieve a closer alignment with the observed cosmic expansion data.
   - Parameters such as the universe's 4D expansion rate and initial conditions are tuned for the best possible fit, revealing how HU can account for redshift patterns in SN1a data.

4. **Comparison with Standard Cosmology**:

   - Comparative plots display HU's optimized predictions alongside observational data and traditional cosmological models (e.g., ΛCDM) to gauge HU's performance.
   - The plots and fitting results reveal insights into HU's ability to model cosmic expansion similarly to standard models, potentially without the need for dark energy.

5. **Residual and Error Analysis**:

   - Error metrics and residuals between the model predictions and observed data are calculated and visualized. This section quantifies the accuracy of the dual-optimized HU model, assessing its robustness in explaining SN1a-derived distances.

Purpose within HU Theory
This notebook strengthens HU's cosmological model by applying a more refined fitting approach to SN1a data. The dual optimization allows for a more nuanced comparison between HU predictions and observed cosmic expansion, further testing HU's hypothesis of cosmic evolution without dark energy.

In fact, optimizing both the 4D radius and the Absolute Luminosity dependence yields a worse fitting to the data. The SN1a_short seems to be the best fitting. Optimization is an art.

The `AAA_Final_SN1a-short.ipynb` notebook provides a streamlined analysis of Type Ia supernovae (SN1a) data, particularly focusing on comparing observational data with predictions based on the Hypergeometrical Universe (HU) model. This version of the SN1a analysis emphasizes essential calculations and model comparisons, aiming to confirm HU's applicability in explaining cosmic expansion without dark energy, while reducing the complexity seen in more comprehensive notebooks.

Notebook Overview

1. **SN1a Data Analysis and Preparation**:

    - This notebook imports SN1a data, ensuring it is properly formatted and cleaned for HU model fitting.
    - Key SN1a observations, particularly luminosity distances across various redshifts, are the primary data points for model comparison.

2. **HU Model Distance Calculation**:

    - Using HU's framework, the notebook computes theoretical luminosity distances for each redshift, applying HU's unique scaling principles that rely on a lightspeed-expanding hyperspherical geometry.
    - Functions similar to `calculateLuminosityDistance_HU` (from other SN1a notebooks) predict distances, using HU's parameters to generate a model distance-redshift relationship.

3. **Comparison of Observational Data with HU Predictions**:

    - Observational data points are plotted alongside HU model predictions, enabling direct visual comparison across a wide redshift range.
    - The notebook uses concise plots to illustrate how HU's predictions align with the observed cosmic expansion patterns, providing a simplified yet comprehensive view of HU's potential validity.

4. **Error and Residual Analysis**:

    - The notebook includes an error analysis to quantify deviations between HU model predictions and observational data.
    - Residuals between observed SN1a luminosity distances and HU predictions are calculated and visualized, offering a summary of HU's performance in modeling observed cosmic expansion without dark energy.

Purpose within HU Theory

This concise notebook allows for a rapid assessment of HU's applicability to SN1a data, focusing on validating HU's framework through simplified analysis. By streamlining calculations, this version highlights HU's potential to explain cosmic expansion efficiently, particularly aiming to demonstrate that the HU model aligns well with empirical data for SN1a events.

The `AAA_Final_SN1a-SingleOptimization.ipynb` notebook provides an analysis of Type Ia supernovae (SN1a) data with a focus on optimizing Hypergeometrical Universe (HU) model parameters through a single optimization approach. This analysis aims to validate the HU

framework by fitting its predictions to observational SN1a data, testing how effectively the model explains cosmic expansion without the need for dark energy.

Notebook Overview

1. **Data Loading and Initial Preparation**:

    - SN1a data, including redshift and luminosity distances, is imported to serve as a foundation for the model fit.
    - The data is preprocessed to ensure it is compatible with the HU model calculations, focusing on distances that correlate with cosmic expansion measurements.

2. **Single Optimization Process**:

    - This notebook uses a single optimization function to fit HU parameters, adjusting the theoretical luminosity distances calculated by HU to best align with observed SN1a data.
    - This process minimizes residuals between HU predictions and empirical data points, refining parameters such as the universe's 4D expansion rate.

3. **Plotting Model Fits**:

    - The notebook generates comparative plots showing HU model predictions against observational SN1a data, allowing for direct visual assessment.
    - Plots include the redshift-distance relationship, with HU predictions overlaid to illustrate how closely the model aligns with actual data.

4. **Error and Residual Analysis**:

    - The notebook evaluates residuals to measure the accuracy of HU's single-optimized fit, highlighting the model's potential to replicate cosmic expansion patterns without dark energy.
    - This section quantifies HU's performance relative to empirical SN1a distances, showing the feasibility of HU's approach.

Purpose within HU Theory

This notebook's single optimization of HU parameters against SN1a data further tests HU's compatibility with observational cosmology. The analysis demonstrates whether a straightforward optimization approach can bring HU's model in line with observed cosmic expansion, potentially offering a dark energy-free explanation.

The `AAA_Final_SN1a_TiredLight.ipynb` notebook examines the Type Ia supernovae (SN1a) data through the concept of the "tired light" hypothesis within the Hypergeometrical

Universe (HU) framework. This hypothesis suggests that redshift occurs due to light losing energy over cosmic distances rather than due to an expanding universe. This analysis assesses whether HU's tired light model can match SN1a observations without invoking cosmic expansion or dark energy.

Notebook Overview

1. **Data Import and Preprocessing**:

    - The notebook imports SN1a luminosity distance data against redshift, representing standard candle measurements used to infer cosmic distances.
    - Data preprocessing ensures that only accurate and relevant data points are retained for analysis within the HU framework.

2. **Tired Light Model Calculations**:

    - HU's tired light calculations are applied to compute theoretical luminosity distances. This model adjusts the energy loss rate of photons based on distance traveled, producing redshifts without requiring an expanding universe.
    - Functions are defined to calculate the tired light effect on light over cosmic distances, with parameters tuned to match observed redshifts.

3. **Comparison with Observational Data**:

    - Plots compare HU's tired light model predictions with observational SN1a data, particularly focusing on how well the model aligns with redshift-luminosity distance relationships.
    - This visualization assesses the tired light model's feasibility as an alternative to dark energy-driven expansion.

4. **Error and Residual Analysis**:

    - Residuals are calculated between HU tired light predictions and observational data, quantifying the model's accuracy.
    - The error analysis provides insight into the model's ability to explain SN1a data within the HU framework, testing whether it can account for cosmic redshift without dark energy or expansion.

Purpose within HU Theory

This notebook applies the HU model's tired light hypothesis to one of cosmology's key datasets, Type Ia supernovae, challenging the traditional expansion-driven redshift interpretation. It examines whether a non-expanding, tired light approach can provide comparable results, suggesting an alternative explanation for observed redshifts in distant astronomical objects.

The `AAA_Final_The_Mass_and_Radius_of_the_Universe.ipynb` notebook calculates the mass and radius of the universe using parameters defined within the Hypergeometrical Universe (HU) framework. It specifically considers adjustments to the gravitational constant (G), permeability (\mu_0), and permittivity (\epsilon_0), factoring in Earth's velocity with respect to the Cosmic Microwave Background (CMB).

Notebook Overview

1. **Gravitational and Electromagnetic Constant Adjustments**:

    - The notebook starts by correcting the gravitational constant (G), magnetic permeability (\mu_0), and electric permittivity (\epsilon_0) based on Earth's velocity relative to the CMB (369.82 km/s).
    - This adjustment applies a coefficient derived from relativistic considerations, affecting the constants used in HU calculations.

2. **Mass of the Universe Calculation**:

    - The notebook computes the mass of the universe, (M), using the HU-specific formula: [M = \frac{2c^2 R_0}{3G}] where (R_0) is the universe's radius (set to 14.04 billion light-years). This formula derives from HU's interpretation of cosmic mass distribution in a 4D hyperspherical geometry.

3. **Volume and Initial Radius of the Universe**:

    - The volume of the universe is calculated based on its radius (R_0), and the initial radius is determined by inverting the relationship to the initial density.
    - Using this density, the notebook calculates an initial radius that aligns with HU's hypothesis, converting the result into light-years per second.

4. **De Broglie Wavelength**:

    - Additional calculations for the de Broglie wavelength and associated parameters are suggested, likely to relate particle properties with cosmological measurements in HU.

Purpose within HU Theory

This notebook aims to provide a quantitative estimate of the universe's mass and radius in alignment with HU's theoretical structure, relying on modifications to fundamental constants based on cosmic velocities. This approach offers a basis for further HU analyses related to cosmic scale and density.

The `AAA_Final_TullyFisher.ipynb` notebook analyzes the Tully-Fisher relation, a well-known empirical relationship in astronomy that connects the luminosity of spiral galaxies to

their rotational velocities. This analysis aims to explore this relationship within the Hypergeometrical Universe (HU) framework by examining data from the GHASP (Gassendi H-alpha survey of SPirals) database and applying linear regression to extract relevant coefficients.

Notebook Overview

1. **Data Loading and Preparation**:

    - The notebook loads GHASP galaxy data from an Excel file, which includes properties such as maximum rotational velocities (`V_RC_Max`, `V_Model_R25`, and `V_TF_max`) and luminosity values (`MK`, `MH`).
    - Columns for logarithmic values of velocities are created to facilitate regression analysis, converting velocities to a log scale commonly used in Tully-Fisher studies.

2. **Data Visualization**:

    - The notebook creates scatter plots to examine the relationship between logarithmic rotational velocities and absolute magnitudes for (K)-band and (H)-band luminosities (`MK` and `MH`), as well as other parameters like `log_MB_and_J`.
    - These visualizations help identify linear trends between luminosity and rotational velocity, providing a basis for further regression analysis.

3. **Linear Regression Analysis**:

    - A linear regression model is applied to predict the Tully-Fisher relation coefficients, specifically focusing on the relationship between `log_V_TF_max` (maximum rotational velocity) and `log_M_bar` (a measure of baryonic mass).
    - The regression coefficients (slope and intercept) are extracted, allowing the notebook to quantify the Tully-Fisher relationship within this dataset.

4. **Result Interpretation and Plotting**:

    - The regression results are plotted, with the line of best fit overlaid on the scatter plot of `log_V_TF_max` versus `log_M_bar`.
    - The model coefficients (\alpha) and (\beta) from the linear regression are displayed on the plot, providing the parameters for the Tully-Fisher relation in this analysis.

Purpose within HU Theory

This notebook contributes to HU by examining the Tully-Fisher relation through data-driven analysis, helping to understand how galaxy dynamics (rotational velocity) relate to luminosity within the context of HU's cosmology. The findings may provide insights into the mass

distribution in galaxies under HU's alternative framework, potentially challenging or refining interpretations within standard cosmology.

The `AAA_Final_TwinParadox.ipynb` notebook addresses the twin paradox by calculating time dilation effects during accelerated and coasting phases of a journey in space, using symbolic calculations. This analysis examines relativistic effects on time experienced by twins under varying acceleration, and it considers how this affects the time difference experienced by an astronaut relative to an observer on Earth.

Notebook Overview

1. **Symbolic Setup for Time Dilation**:
    - Symbols are defined for acceleration ((a)), speed of light ((c)), velocity ((v)), and time ((t)), as well as for integration variables.
    - Time dilation expressions are formulated using hyperbolic functions (`cosh`, `atanh`), which are suited to relativistic acceleration scenarios.

2. **Integral Calculation for Accelerated Motion**:
    - An integral expression is set up for the accelerated phase of the journey, with time dilation calculated over a defined period based on the acceleration.
    - The notebook applies the condition (a \cdot t < c) (to ensure velocity remains subluminal) and simplifies the result for different journey segments.

3. **Differentiation and Limits**:
    - Derivatives and limits are taken for time dilation expressions, specifically to explore behavior at extreme velocities and durations.
    - The calculations include transformations based on the acceleration parameter and maximum coasting speed.

4. **Example Calculations for Accelerated and Coasting Phases**:
    - The notebook evaluates time dilation over one day of accelerated motion, as well as for an extended coasting period, such as ten years.
    - It calculates both the total time on Earth and on the spacecraft, allowing a comparison between the time experienced by the traveling twin and the Earth-bound twin.

5. **Output Summary**:
    - The final results display the time difference accumulated between the twins over the journey, with values expressed in years. This aligns with HU's critique of

traditional time dilation interpretations by providing a tangible example of relativistic effects.

Purpose within HU Theory
This notebook explores the twin paradox from the HU perspective, emphasizing the differences in time experienced by observers in varying states of motion. By calculating time dilation for accelerated and inertial phases, it sheds light on HU's stance toward relativistic motion and the effects of acceleration, potentially challenging or refining traditional interpretations of time dilation.

The `AAA_Final_VelocityEffectOnFD_Tunneling_DilatonField.ipynb` notebook visually explores the impact of absolute velocity on the tunneling process of the Fundamental Dilator (FD) and its associated dilaton waves within the Hypergeometrical Universe (HU) framework. This analysis includes visualizations of 3D and 4D waveforms and their interactions with the FD, providing insights into how velocity affects the manifestation of the FD in different dimensions.

Notebook Overview

1. **Setup and Visual Representation of FD Waves**:

    - The notebook sets up visualization parameters, defining mathematical functions to represent 3D and 4D waveforms associated with the FD.
    - A main figure is created to show these waveforms, focusing on how velocity influences their shape and propagation, with 3D wave components plotted alongside their 4D counterparts.

2. **Wave Interactions and Annotations**:

    - Using trigonometric and hyperbolic functions, the notebook models sinusoidal (3D) and hyperbolic (4D) waves to represent the spatial effects of FD tunneling and dilaton wave propagation.
    - Annotation tools are used to highlight specific points and components in the visualization, marking the transition points where the FD would appear to "tunnel" or shift.

3. **Transformation and Projection**:

    - The notebook applies transformations that vary with velocity to observe how the FD's 3D and 4D manifestations interact. These transformations adjust the wave paths, simulating the effect of increasing velocity on FD projections into 3D space.

- This projection simulates how the dilaton field, which represents gravitational effects in HU, shifts under different velocities, indicating a change in FD's influence as perceived in 3D.

4. **Additional Simulation and Plotting**:

   - Additional plotting commands are used to overlay several frames of the transformed waves, demonstrating how increasing velocity distorts the FD's spatial projection.
   - The notebook finalizes with a plot of varying waveform projections, saving the final image to illustrate the velocity effect on the FD's tunneling and dilaton field behavior.

Purpose within HU Theory
This notebook provides a visual and mathematical interpretation of the FD's behavior under velocity changes, showing how dilaton waves shift in both 3D and 4D spaces. This serves HU's goal of understanding how gravitational effects manifest as wave interference patterns, with velocity impacting the FD's observable effects in lower-dimensional projections.

The `AAA_Final_VenusPerihelionCalculation_HU.ipynb` notebook investigates the perihelion precession of Venus's orbit using both classical mechanics and the Hypergeometrical Universe (HU) model. This approach calculates the theoretical precession rates and compares the outcomes from HU with those of general relativity (GR), providing insights into how HU could account for orbital anomalies typically attributed to relativistic effects.

Notebook Overview
1. **Symbolic Calculations for Orbital Elements**:

   - Symbols and functions are defined to represent orbital parameters, such as radial distance (r) and angular velocity (\omega).
   - The notebook uses symbolic differentiation to derive expressions for (r), (\dot{r}), and other orbital dynamics that determine the perihelion precession.

2. **Numerical Calculations Using Venus's Orbital Parameters**:

   - The orbital parameters of Venus, including its semi-major axis and eccentricity, are plugged into the calculations to estimate precession rates.
   - Numerical methods such as integration and optimization are used to solve for optimized orbital precession in both HU and GR frameworks.

3. **Error Minimization in HU Model**:

- An error function (`error_functionHU`) is defined to evaluate the difference between theoretical and numerical accelerations, optimizing HU's fit for the orbital precession of Venus.
- The error minimization process finds the best fit for the HU model, aligning its precession predictions more closely with observational data.

4. **Comparison of Precession Rates**:

    - The notebook calculates precession rates in arcseconds per century for Venus under both HU and GR models.
    - Comparative visualizations show the precession rates predicted by HU and GR, allowing direct comparison with observed data.

5. **Graphical Simulations of Orbits**:

    - Polar and Cartesian plots of the orbital trajectory are included, with additional simulation data to illustrate Venus's orbit and highlight any precession over time.
    - These visualizations reinforce how HU's alternative gravitational model might produce observable deviations in planetary orbits.

Purpose within HU Theory
This notebook extends HU's gravitational framework to planetary motion by calculating perihelion precession. By examining Venus's orbit, it tests HU's validity in predicting precession rates that match GR predictions, providing an alternative model for gravitational effects traditionally explained by relativity.

The `AAA_GoldenCopy_BlackholiumUniverseFunction_20211120.ipynb` notebook appears to explore a theoretical model involving "Blackholium," a concept related to the density and formation of black holes within the context of the Hypergeometrical Universe (HU). The analysis includes computations on gravitational corrections, baryonic acoustic oscillations (BAO), energy and density profiles, and physical properties like sound velocity in a hypothetical Blackholium universe.

Notebook Overview

1. **Gravitational Corrections and Fundamental Constants**:

    - The notebook adjusts constants such as the gravitational constant (G), magnetic permeability (\mu_0), and electric permittivity (\epsilon_0), based on Earth's velocity relative to the Cosmic Microwave Background (CMB).
    - This correction, which applies a relativistic coefficient, refines these constants for application in the HU model.

2. **Energy and Density Profiles of the Universe**:

- Various energy profiles are plotted, showing energy density evolution from the Big Bang to present time.
- Density values across different epochs are calculated, labeled by key phases such as “densityBlackholium” and “densityNeutronium,” representing different stages in cosmic evolution.

3. **Sound Velocity and Proton Fraction**:

   - The notebook models sound velocity and proton fraction as functions of cosmic time, with visualizations comparing these properties across epochs.
   - These calculations potentially relate to conditions in a dense “Blackholium” state, simulating sound wave transmission properties under HU assumptions.

4. **Baryonic Acoustic Oscillations (BAO) Calculation**:

   - BAO-related measurements and theoretical calculations are performed, using a subtended angle and radius (130 Mpc) for the oscillations.
   - The notebook calculates BAO-related distances under the HU framework, testing how well HU can model this critical feature of cosmic structure.

5. **Plots and Visualizations**:

   - Multiple plots illustrate energy profiles, density variations, sound velocity, and proton fraction over cosmic time.
   - These visualizations provide a dynamic view of how the HU model explains physical properties under extreme densities, with a focus on conditions that might give rise to structures like Blackholium.

Purpose within HU Theory
This notebook offers a comprehensive exploration of HU’s cosmological model, focusing on the properties of matter and sound propagation in an extremely dense universe. It contributes to HU by quantifying the behavior of a hypothesized “Blackholium” universe, examining whether HU can replicate key features like BAO and cosmic density variations over time.

The `AAA_NeutronModelSpiral.ipynb` notebook visualizes a spiral model of the neutron, using disk-like representations for components such as protons and electrons in a 3D space. This visualization integrates HU's approach by depicting tunneling phases and time dilation effects within a cylindrical coordinate system, creating a dynamic, spiral-like structure that represents fundamental particles.

Notebook Overview
1. **Setup and Constants Definition**:

- Constants such as the speed of light (c), hydrogen atom mass (m_h), Compton wavelength (\lambda_t), and Compton period (T_t) are defined.
- A radius (R) is set to approximate the particle's size, and calculations for angular and tangential velocities are included to simulate rotational effects, resulting in a time dilation factor.

2. **Phase Shift Calculation with Time Dilation**:

   - A function `calculate_phase_shift` computes the phase shift for each tunneling event, adjusted by the time dilation factor to account for relativistic effects at different rotational speeds.

3. **Disk Rotation and Spiral Plotting**:

   - The function `rotate_disk` applies rotation to each disk in the spiral structure, setting orientation based on calculated phase shifts.
   - The `plot_disk` function then visualizes each disk at specific radial and angular positions within the cylindrical coordinate system, alternating between colors (green and yellow) to represent dilation and contraction phases.

4. **3D Spiral Representation**:

   - A 3D spiral line is added to visually guide the viewer through the particle's structure, representing the trajectory of tunneling or transition phases.
   - The visualization is completed with axis labels and customized view angles to emphasize the particle's 3D cylindrical configuration.

Purpose within HU Theory
This notebook illustrates HU's view of fundamental particles as dynamic structures, using a spiral model to show the neutron's internal phases. By including time dilation and phase shifts, it demonstrates how HU envisions the spatial and temporal dynamics of particles, which could be applied to explain the neutron's behavior in a 4D spatial manifold.

The `AAA_Relativistic_Classical_KE_Ratio.ipynb` notebook examines the relationship between relativistic and classical kinetic energy, particularly for particles at high velocities. The calculations and plots provide insights into how kinetic energy behaves as velocities approach the speed of light, using both theoretical formulas and specific examples, such as protons at Large Hadron Collider (LHC) energies.

Notebook Overview

1. **Symbolic Setup for Integrals and Ratios**:

- The notebook defines symbolic variables and integrands for calculating integrals related to energy expressions, allowing for manipulation of kinetic energy formulas in both classical and relativistic contexts.

2. **Relativistic vs. Classical KE Ratio Calculation**:

   - Using a velocity range up to nearly the speed of light, the notebook calculates the ratio of relativistic to classical kinetic energy.
   - A plot of this ratio shows how relativistic kinetic energy diverges from classical kinetic energy as velocity increases, illustrating the limitations of classical mechanics at high speeds.

3. **Proton Velocity and Kinetic Energy at High Energies**:

   - The notebook includes specific calculations for a proton with 938 MeV of kinetic energy and a high-energy proton at 6.5 TeV (typical of LHC experiments).
   - These calculations convert kinetic energy from electron volts (eV) to velocity and express this as a fraction of the speed of light, demonstrating the relativistic effects at high energies.

4. **Energy-to-Velocity Conversion Function**:

   - A function `energy_to_velocity` converts kinetic energy (in eV) to velocity for a particle with given mass, applying the relativistic kinetic energy formula.
   - The example calculation for a proton at 6.5 TeV shows how relativistic velocity approaches, but remains below, the speed of light, reinforcing the constraints imposed by relativistic mechanics.

5. **Cosmic Ray Energy Distribution**:

   - An additional section generates a Gaussian distribution of cosmic ray energies to simulate a probability density for high-energy particles.
   - The plot visualizes the distribution of cosmic ray energies, which can be used for further studies on particle velocities in a cosmic context.

Purpose within HU Theory
This notebook provides HU with a comparison of classical and relativistic energy models, relevant to high-energy particle physics. It explores how kinetic energy behaves under different theoretical frameworks, reinforcing HU's approach to relativistic phenomena and particle dynamics.

The `AAA_ReverseField_Truster.ipynb` notebook investigates the dynamics of a charged particle under the influence of an oscillating electric field, with a focus on calculating the particle's velocity and the associated radiation reaction force. This analysis is relevant to

HU's study of particle motion in intense fields and how radiation reaction forces influence the particle's response over time.

Notebook Overview

1. **Setup of Electric Field and Radiation Reaction**:

    - Constants like charge (q), mass (m), electric field amplitude (E_0), field frequency (\omega), elementary charge (e), vacuum permittivity (\epsilon_0), and speed of light (c) are defined for an electron.
    - The notebook defines an oscillating electric field (E_t = E_0 \sin(\omega t)) and calculates the resulting acceleration (a_t) and jerk (rate of change of acceleration).

2. **Radiation Reaction Force Calculation**:

    - The Abraham-Lorentz radiation reaction force is calculated as: [F_{\text{rad}} = \frac{e^2}{6 \pi \epsilon_0 c^3} \frac{d^3p}{dt^3}]
    - Using the jerk of the acceleration, this force quantifies the particle's resistance to changes in velocity due to energy loss from radiation.

3. **Relativistic Velocity Calculation**:

    - The notebook computes relativistic velocity for the particle under the electric field, accounting for time dilation as the particle's speed approaches the speed of light.
    - It applies iterative steps to calculate velocity over a defined time interval, considering the increasing mass factor (\gamma) as velocity rises.

4. **Plotting Results**:

    - Plots are generated to display the electric field (E(t)), particle acceleration (a(t)), and radiation reaction force (F_{\text{rad}}) over time.
    - A secondary plot presents the relativistic velocity of the particle as it responds to the electric field, showing how the field influences particle motion under relativistic constraints.

Purpose within HU Theory

This notebook explores how a charged particle responds to high-frequency oscillating fields and the effect of radiation reaction forces on its velocity. Such calculations could support HU's study of field-particle interactions, specifically where high-intensity fields and relativistic effects are significant.

The `AAA_SchwarzschildDerivationOfBinetForm.ipynb` notebook derives the Binet form for a particle's orbit within a Schwarzschild metric, relevant to studying gravitational

effects on planetary orbits. By reformulating the Schwarzschild metric in terms of the inverse radial coordinate (u = \frac{1}{r}), this notebook seeks to derive the equations of motion for orbital paths, particularly capturing the relativistic precession observed in planetary orbits.

Notebook Overview

1. **Metric and Variable Setup**:

    - Variables and symbols are defined for gravitational constant (G), mass (M), speed of light (c), energy (E), angular momentum (L), and proper time (\tau).
    - The radial coordinate (r) is expressed in terms of (u = \frac{1}{r}), allowing for orbital equations to be reformulated in terms of (u) and the angular coordinate (\phi).

2. **Equations of Motion**:

    - The notebook calculates derivatives of (r) with respect to (\tau) (proper time), obtaining expressions for (\frac{dr}{d\tau}) and (\frac{d\phi}{d\tau}) based on the Schwarzschild metric.
    - Using the metric relation for a massive particle, the notebook derives an expression for (\left(\frac{dr}{d\tau} \right)^2), incorporating the relativistic terms from the Schwarzschild solution.

3. **Conversion to Binet Form**:

    - By substituting (\frac{d\phi}{d\tau}) into the derived expression for (\frac{dr}{d\tau}), the notebook begins to reformulate the orbital motion in terms of (u) and (\phi), aiming to isolate (\frac{d^2u}{d\phi^2}).
    - The Binet form, which expresses the second derivative of (u) with respect to (\phi), is ultimately obtained. This form is essential for studying relativistic corrections in orbit equations.

4. **Relativistic Orbital Precession**:

    - The resulting Binet form encapsulates the relativistic effects that lead to orbital precession. In practical terms, this allows for calculating perihelion shifts in orbits, as observed in planets like Mercury under strong gravitational fields.

Purpose within HU Theory

This notebook contributes to HU by providing a classical relativistic framework for orbit precession, essential for validating HU's gravitational assumptions. By deriving the Binet form, the notebook provides a foundation for understanding how relativistic modifications impact orbital paths, which may be comparable or contrasting with HU's predictions on gravity.

The `AAA_Sympy_DerivationOfBinetForm.ipynb` notebook uses symbolic computation with SymPy to derive the Binet equation, which describes the radial distance in a gravitational field based on an inverse radius variable (u = \frac{1}{r}). The notebook is set up to analyze gravitational influences on orbital motion, particularly focusing on relativistic corrections in Schwarzschild spacetime.

Notebook Overview

1. **Symbolic Definitions and Radial Derivatives**:

    - The notebook defines variables for time (t), radial distance (r(t)), angular position (\theta(t)), gravitational constant (G), mass (M), and speed of light (c).
    - Using SymPy, derivatives of (r) and (\theta) with respect to (t) are calculated, laying the groundwork for expressing the orbital motion through the inverse radial distance (u).

2. **Angular Momentum and Radial Velocity Substitutions**:

    - Conservation of angular momentum is applied, allowing the angular velocity (\frac{d\theta}{dt}) to be expressed in terms of (u).
    - Using the chain rule, the notebook expresses the radial distance derivative (\frac{dr}{dt}) in terms of (u) and simplifies it using the angular momentum constraint.

3. **Second Derivative of Radial Distance**:

    - The notebook calculates the second derivative (\frac{d^2r}{dt^2}) in terms of (u) and (\theta), a critical step in reformulating the orbital motion equations into Binet form.
    - This involves expressing radial acceleration as a function of the inverse radial distance and its derivatives, ultimately leading to a simplified form for further analysis.

4. **Gravitational Force and Centrifugal Force Terms**:

    - The effective force equation is set up to include gravitational and centrifugal terms, leading to an equation that resembles Newton's second law with relativistic corrections.
    - By manipulating these terms, the notebook arranges them into a form consistent with the standard Binet equation, which includes relativistic adjustments based on the Schwarzschild metric.

5. **Final Binet Equation and Relativistic Modifications**:

    - The derived form of the Binet equation incorporates relativistic factors, showing the effect of gravitational and inertial forces on the orbital path.

- The notebook concludes with symbolic simplification steps, yielding a compact form of the Binet equation that can be used for further orbital analysis, such as perihelion precession in planetary orbits.

Purpose within HU Theory
This notebook provides a symbolic derivation of the Binet equation, critical for understanding relativistic orbital motion. Within HU, this derivation may help analyze orbits under HU's gravitational framework, showing how relativistic effects influence paths without relying on traditional spacetime curvature assumptions.

The `AAA_VisualizeConstellationsOnMollweide.ipynb` notebook visualizes constellations and stellar coordinates on different celestial map projections, with a focus on the Mollweide projection. This type of projection is commonly used in astronomy to display star positions across the entire sky while preserving area proportions, making it suitable for constellation mapping.

Notebook Overview

1. **Coordinate Setup for Constellations**:

    - Coordinates for the Orion constellation are defined in Cartesian 3D space (`x`, `y`, `z`) for basic plotting. These coordinates provide a foundation for both 2D and 3D scatter plots of Orion.
    - In a later cell, RA (right ascension) and DEC (declination) coordinates are defined using the `astropy` library's `ICRS` coordinate system, allowing accurate mapping of stars in various constellations.

2. **2D and 3D Visualizations of Orion**:

    - The notebook creates a simple 2D scatter plot of Orion, setting yellow star markers on a black background to enhance visibility.
    - A 3D plot of Orion's star positions is also generated, which visualizes the depth and spatial arrangement of stars within the constellation.

3. **Mollweide and Other Celestial Projections**:

    - The main focus is on mapping constellations using celestial projections. Functions from `desiutil.plots` are imported to initialize sky maps and plot star positions across different projections, including Mollweide, Hammer, and sky-binned visualizations.
    - Specific cells display scatter plots of stars projected onto the Mollweide and Hammer projections, demonstrating the positions of stars across the entire sky.

4. **Grid and Healpix Maps**:

- A grid map using random data points simulates a distribution across RA and DEC coordinates, and a Healpix map plots a random field across the celestial sphere.
- These maps illustrate potential density variations across the sky, useful for applications beyond constellation mapping, such as observing density distributions in the cosmic microwave background (CMB).

Purpose within HU Theory

This notebook provides tools for visualizing cosmic structures, potentially aiding HU's analysis of celestial distributions. By using the Mollweide projection, it helps represent star positions and constellations comprehensively, aligning with HU's goal of mapping cosmic data on a large scale.