Eclipse Exploration Simulation

May 4, 2025

[1]:

```
import pandas as pd
import numpy as np
```

[2]:

```
def expected_explore_values(
    tiles_df: pd.DataFrame,
    sector: int,
    draco: bool = False,
    planta: bool = False,
    explored: list = []
) -> pd.Series:
    """
    Compute the *average* (expected) value of each attribute for a single␣
 ↪sector.
    This does NOT do random sampling- it just averages all tiles of the given␣
 ↪sector.

    Parameters:
    -----------
    tiles_df : pd.DataFrame
        DataFrame of all tiles (with columns like 'TileNumber', 'Sector', etc.)
    sector : int
        Which sector to filter on (1, 2, or 3).
    draco : bool
        Whether Draco's special exploration rules are in effect (placeholder␣
 ↪logic).
    planta : bool
        Whether Planta's special exploration rules are in effect (placeholder␣
 ↪logic).

    Returns:
    --------
    pd.Series
        A Series containing the average of each numeric column for the chosen␣
 ↪sector.
    """
    #explored = [] add if you have any
    if len(explored) >= 0:
```

```python
        sector_tiles = tiles_df[~tiles_df.TileNumber.isin(explored)]

    # Filter down to tiles in the requested sector
    sector_tiles = tiles_df[tiles_df['Sector'] == sector].copy()

    # Example: modify "sector_tiles" if Draco’s or Planta’s exploration changes stats
    if draco:
        # Place any Draco-specific logic here
        pass

    if planta:
        # Place any Planta-specific logic here
        pass

    # Calculate the average (mean) of all numeric columns
    return sector_tiles.mean(numeric_only=True)


def simulate_full_game(
    tiles_df: pd.DataFrame,
    n_players: int,
    draco: bool = False,
    planta: bool = False,
    sector_1_explores: int = 6,
    sector_2_explores: int = 6,
    random_seed: int = None
) -> pd.DataFrame:
    """
    Simulates random draws of tiles for an entire Eclipse game.
    Returns the actual sampled tiles so that you can compute
    the 'effective' or realized average (or any other statistics) from the game.

    Key assumptions:
      - Sector 3 tiles are limited based on number of players (the exact formula can vary).
      - Sector 1 and Sector 2 can each be explored up to 6 times (or any limit you choose).
      - Draws are without replacement within each sector.
      - If draco or planta are True, modifies the tile stats or logic accordingly (placeholders).

    Parameters:
    -----------
    tiles_df : pd.DataFrame
        DataFrame of all tiles.
    n_players : int
```

```
        Number of players in the game.  Affects how many Sector 3 tiles are␣
↪available.
    draco : bool
        Draco's special exploration rules toggle (placeholder).
    planta : bool
        Planta's special exploration rules toggle (placeholder).
    sector_1_explores : int
        Maximum number of times Sector 1 can be explored.
    sector_2_explores : int
        Maximum number of times Sector 2 can be explored.
    random_seed : int
        Random seed for reproducibility.

    Returns:
    --------
    pd.DataFrame
        A DataFrame of all the tile pulls that occurred in this simulated game.
    """
    if random_seed is not None:
        np.random.seed(random_seed)

    # Copy so we don’t mutate the original
    df = tiles_df.copy()

    # -- Optional: Apply Draco/Planta logic prior to sampling.
    #    For example, you might tweak the values in certain columns
    #    or remove certain tiles for Draco/Planta, etc.
    if draco:
        # Insert Draco-specific changes here if needed:
        pass
    if planta:
        # Insert Planta-specific changes here if needed:
        pass

    # Separate tiles by sector
    sector_1_tiles = df[df["Sector"] == 1].copy()
    sector_2_tiles = df[df["Sector"] == 2].copy()
    sector_3_tiles = df[df["Sector"] == 3].copy()

    # Shuffle them
    sector_1_tiles = sector_1_tiles.sample(frac=1).reset_index(drop=True)
    sector_2_tiles = sector_2_tiles.sample(frac=1).reset_index(drop=True)
    sector_3_tiles = sector_3_tiles.sample(frac=1).reset_index(drop=True)

    # The number of Sector 3 tiles typically depends on # of players.
    # Adjust this formula as suits your game variant.
    sector_3_draw_count = n_players * 4  # need to turn this into a dict of {"Player Count" : "Sector 3 Tiles"}

    # Draw up to the allowed maximum from each sector
    draws_1 = sector_1_tiles.head(sector_1_explores)  # up to 6 from sector 1 by default
    draws_2 = sector_2_tiles.head(sector_2_explores)  # up to 6 from sector 2 by default
    draws_3 = sector_3_tiles.head(sector_3_draw_count)

    # Combine everything into one "result" DataFrame
    all_draws = pd.concat([draws_1, draws_2, draws_3], ignore_index=True)

    ## need to assign draws to players maybe? whats a good way to analyze this?

    # Return the resulting draws for further analysis.
    return all_draws


# Example usage (you can comment out or remove):
# tiles_df = pd.read_csv("my_eclipse_tiles.csv")
# expected_values_sector3 = expected_explore_values(tiles_df, 3, draco=False, planta=False)
# simulation_results = simulate_full_game(tiles_df, n_players=4, draco=True, planta=False, random_seed=42)
# print(expected_values_sector3)
# print(simulation_results)
def ancient_expected_explore_values(
    tiles_df: pd.DataFrame,
    sector: int,
    draco: bool = False,
    planta: bool = False,
    explored: list = []
) -> pd.Series:
    """
    Compute the *average* (expected) value of each attribute for a single sector.
    This does NOT do random sampling- it just averages all tiles of the given sector.

    Parameters:
    -----------
    tiles_df : pd.DataFrame
        DataFrame of all tiles (with columns like 'TileNumber', 'Sector', etc.)
    sector : int
        Which sector to filter on (1, 2, or 3).
    draco : bool
        Whether Draco's special exploration rules are in effect (placeholder␣
↪logic).
    planta : bool
        Whether Planta's special exploration rules are in effect (placeholder␣
↪logic).

    Returns:
    --------
    pd.Series
        A Series containing the average of each numeric column for the chosen␣
↪sector.
    """
    #explored = [] add if you have any
    if len(explored) >= 0:
        sector_tiles = tiles_df[~tiles_df.TileNumber.isin(explored)]

    # Filter down to tiles in the requested sector
    sector_tiles = tiles_df[tiles_df['Sector'] == sector].copy()

    # Example: modify "sector_tiles" if Draco’s or Planta’s exploration changes␣
↪stats
    if draco:
        # Place any Draco-specific logic here
        pass

    if planta:
        # Place any Planta-specific logic here
        pass

    # Calculate the average (mean) of all numeric columns
    return sector_tiles.groupby('AncientResistance').mean(numeric_only=True)
```

[3]:
```
tiles_df = pd.read_csv("eclipse_tiles.csv")
```

[4]:
```
tiles_df
```

[4]:
```
    Unnamed: 0  TileNumber  Sector  AncientResistance  Materials  Science  \
0            0         313       3                  0          0        0
1            1         105       1                  1          0        1
2            2         106       1                  0          1        1
3            3         107       1                  0          0        0
4            4         102       1                  0          0        1
..         …           …     …                 …        …       …
57          57         399       3                  0          0        0
58          58         321       3                  0          0        0
59          59         304       3                  0          1        0
```

```
60          60         394       3                  0          1         0
61          61         393       3                  0          0         1

    Money  White  AdvMaterials  AdvScience  …  AdvWhite  DiscoveryTile  \
0       0      1             0           0  …         0           True
1       1      0             1           0  …         0           True
2       0      0             0           0  …         0          False
3       1      0             1           0  …         0          False
4       0      0             0           0  …         0          False
..     …    …           …         …  …       …             …
57      0      0             0           0  …         0           True
58      1      0             0           0  …         0          False
59      0      0             0           0  …         0          False
60      0      0             0           0  …         0          False
61      0      0             0           0  …         0          False

    VictoryPoints  BlackHole  Wormhole  Anomalies  Supernova  Nebula  \
0               1      False     False      False      False   False
1               3      False     False      False      False   False
2               2      False     False      False      False   False
3               2      False     False      False      False   False
4               3      False     False      False      False   False
..             …       …       …        …        …     …
57              0       True     False      False      False   False
58              1      False     False      False      False   False
59              1      False     False      False      False   False
60              1      False     False      False      False   False
61              1      False     False      False      False   False

    AncientHive  Pulsar
0             0   False
1             0   False
2             0   False
3             0   False
4             0   False
..          …      …
57            0   False
58            0   False
59            0   False
60            0    True
61            0    True

[62 rows x 21 columns]
```

```
[5]: tiles_df.drop('Unnamed: 0', axis = 1, inplace = True)
```

```
[6]: ancient_expected_explore_values(tiles_df, 3)
```

[6]:

```
                    TileNumber  Sector  Materials   Science     Money  \
AncientResistance
0                   339.962963     3.0   0.259259  0.296296  0.185185
1                   325.000000     3.0   0.400000  0.400000  0.200000
2                   301.000000     3.0   0.000000  1.000000  1.000000
3                   319.000000     3.0   1.000000  1.000000  0.000000

                       White  AdvMaterials  AdvScience  AdvMoney  AdvWhite  \
AncientResistance
0                   0.296296      0.148148    0.111111  0.222222  0.037037
1                   0.200000      0.000000    0.000000  0.200000  0.000000
2                   0.000000      1.000000    0.000000  0.000000  0.000000
3                   0.000000      1.000000    0.000000  1.000000  0.000000

                    DiscoveryTile  VictoryPoints  BlackHole  Wormhole  \
AncientResistance
0                        0.407407       0.962963   0.074074  0.074074
1                        1.000000       1.200000   0.000000  0.000000
2                        1.000000       2.000000   0.000000  0.000000
3                        1.000000       2.000000   0.000000  0.000000

                    Anomalies  Supernova  Nebula  AncientHive    Pulsar
AncientResistance
0                    0.037037   0.074074     0.0          0.0  0.074074
1                    0.000000   0.000000     0.2          0.0  0.000000
2                    0.000000   0.000000     0.0          0.0  0.000000
3                    0.000000   0.000000     0.0          1.0  0.000000
```

[7]:

```
expected_explore_values(tiles_df, 3)
```

[7]:

```
TileNumber           336.000000
Sector                 3.000000
AncientResistance      0.294118
Materials              0.294118
Science                0.352941
Money                  0.205882
White                  0.264706
AdvMaterials           0.176471
AdvScience             0.088235
AdvMoney               0.235294
AdvWhite               0.029412
DiscoveryTile          0.529412
VictoryPoints          1.058824
BlackHole              0.058824
Wormhole               0.058824
Anomalies              0.029412
Supernova              0.058824
```

```
Nebula                  0.029412
AncientHive             0.029412
Pulsar                  0.058824
dtype: float64
```

[280]:

[281]:

[ ]:

[286]:

[284]:

[275]:

[ ]:

[8]:

```
mask = tiles_df[tiles_df.Sector == 3]

mask.groupby("Sector")["AncientResistance"].value_counts() #/mask.shape[0]
```

[8]:

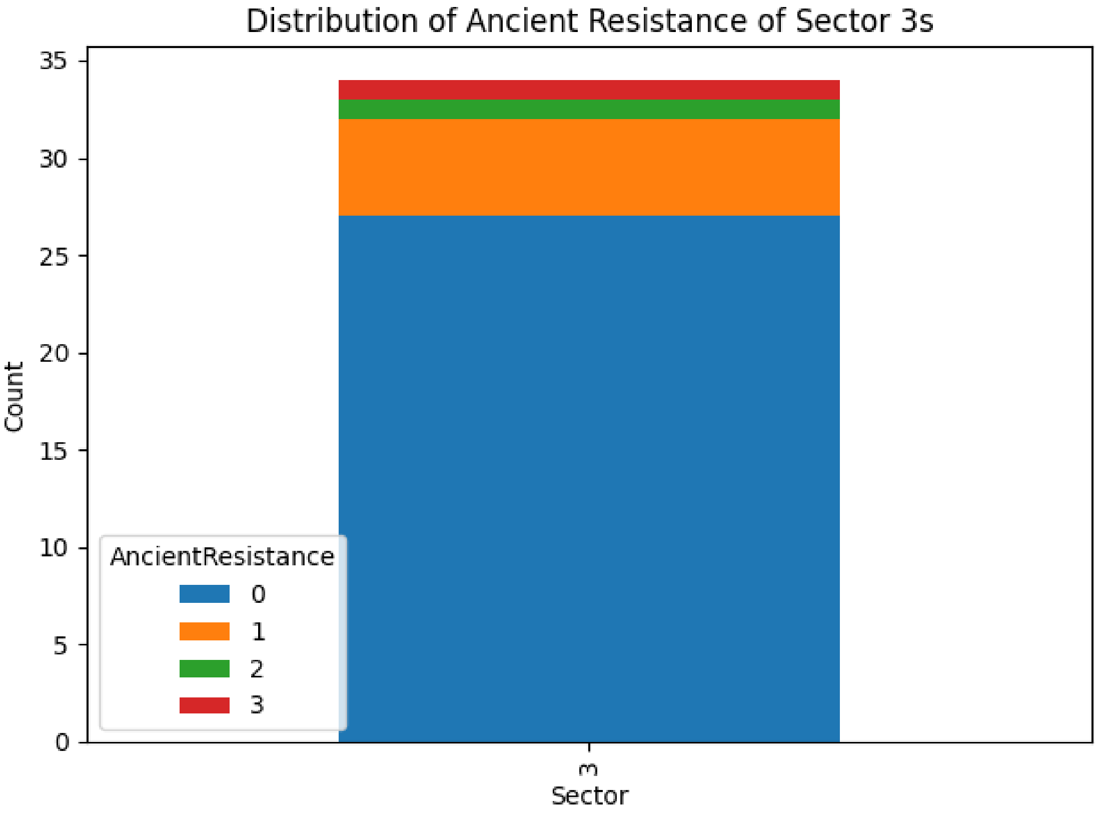
```
Sector  AncientResistance
3       0                    27
        1                     5
        2                     1
        3                     1
Name: count, dtype: int64
```

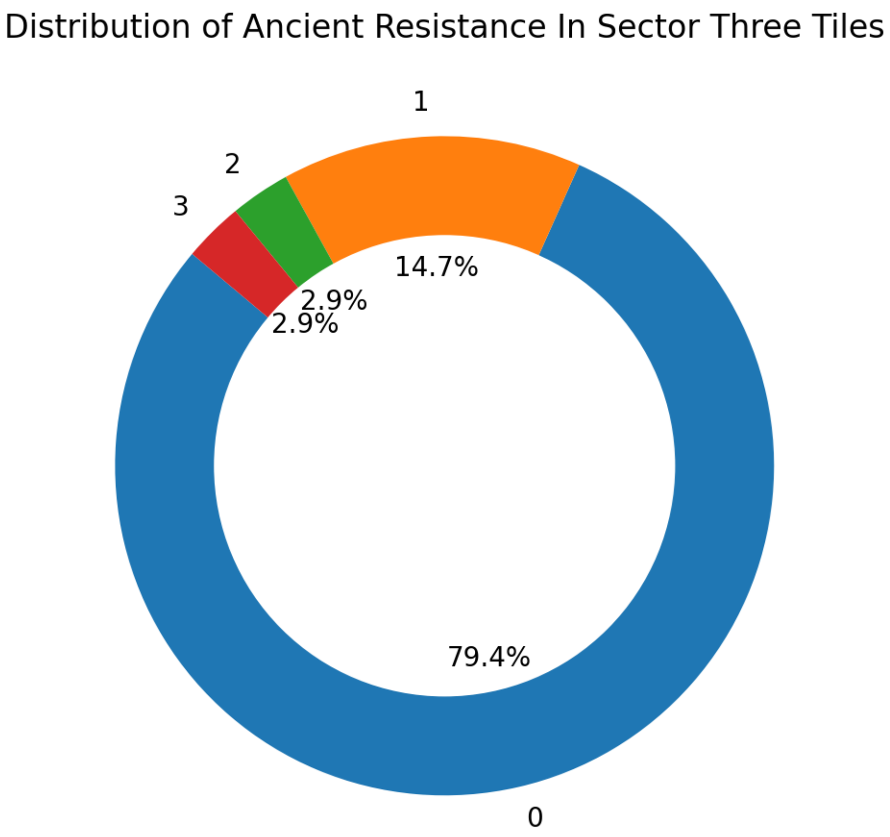
[9]:

```
import matplotlib.pyplot as plt

# 1) Calculate group counts
group_counts = mask.groupby("Sector")["AncientResistance"].value_counts().
 ↪rename("Count").reset_index()

# 2) Create a pivoted table where:
#    - rows = Sector
#    - columns = AncientResistance levels
#    - values = Count
pivot_df = group_counts.pivot_table(index="Sector",
                                    columns="AncientResistance",
                                    values="Count",
                                    fill_value=0)

################################################################################
#                           STACKED BAR CHART                                  #
```

```
###############################################################################
plt.figure(dpi=200)  # High-resolution figure
pivot_df.plot(kind='bar', stacked=True)
plt.title("Distribution of Ancient Resistance of Sector 3s")
plt.xlabel("Sector")
plt.ylabel("Count")
plt.legend(title="AncientResistance")
plt.tight_layout()
plt.show()

###############################################################################
#                              DONUT CHART                                    #
###############################################################################
# For a donut chart, it's more common to look at an overall breakdown.
# Below, we get overall counts for AncientResistance (across all Sectors).
overall_counts = mask["AncientResistance"].value_counts()

plt.figure(dpi=200)  # High-resolution figure

# Basic pie chart
wedges, texts, autotexts = plt.pie(
    overall_counts,
    labels=overall_counts.index,
    autopct='%1.1f%%',
    startangle=140
)

# Draw a circle in the center to make it look like a donut
centre_circle = plt.Circle((0,0), 0.70, fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

plt.title("Distribution of Ancient Resistance In Sector Three Tiles")
plt.tight_layout()
plt.show()
```

```
<Figure size 1280x960 with 0 Axes>
```


Distribution of Ancient Resistance of Sector 3s
Count
0
5
10
15
20
25
30
35
Sector
3
AncientResistance
0
1
2
3



```
[19]: tiles_df.groupby('Sector').mean()
```

```
[19]:         TileNumber  AncientResistance  Materials   Science     Money  \
      Sector
      1       113.090909           0.636364   0.272727  0.454545  0.545455
      2       221.764706           0.529412   0.411765  0.352941  0.411765
      3       336.000000           0.294118   0.294118  0.352941  0.205882

                 White  AdvMaterials  AdvScience  AdvMoney  AdvWhite  DiscoveryTile  \
      Sector
      1       0.181818      0.272727    0.090909  0.454545  0.181818       0.545455
      2       0.117647      0.235294    0.176471  0.176471  0.117647       0.588235
      3       0.264706      0.176471    0.088235  0.235294  0.029412       0.529412

              VictoryPoints  BlackHole  Wormhole  Anomalies  Supernova    Nebula  \
      Sector
      1            2.363636   0.000000  0.000000   0.090909   0.000000  0.000000
      2            1.176471   0.000000  0.058824   0.058824   0.000000  0.058824
      3            1.058824   0.058824  0.058824   0.029412   0.058824  0.029412
```

```
        AncientHive    Pulsar
Sector
1          0.000000  0.000000
2          0.058824  0.000000
3          0.029412  0.058824
```

```
[ ]:
```

```
[23]: simulate_full_game(tiles_df, 3)
```

```
[23]:     TileNumber  Sector  AncientResistance  Materials  Science  Money  White  \
0          110       1                  0          0        0      0      0
1          105       1                  1          0        1      1      0
2          189       1                  0          1        0      1      0
3          106       1                  0          1        1      0      0
4          107       1                  0          0        0      1      0
5          102       1                  0          0        1      0      0
6          210       2                  0          1        0      1      0
7          209       2                  0          0        1      0      0
8          214       2                  1          0        1      0      0
9          208       2                  0          0        0      0      0
10         289       2                  0          1        1      0      0
11         207       2                  0          0        0      0      0
12         395       3                  1          0        0      0      0
13         394       3                  0          1        0      0      0
14         306       3                  0          1        0      1      0
15         318       3                  0          0        0      0      1
16         381       3                  0          0        1      0      0
17         396       3                  0          0        0      0      0
18         323       3                  0          0        0      0      0
19         315       3                  0          0        0      0      0
20         309       3                  0          0        0      1      0
21         316       3                  0          0        0      0      0
22         307       3                  0          0        0      1      0
23         313       3                  0          0        0      0      1

    AdvMaterials  AdvScience  AdvMoney  AdvWhite  DiscoveryTile  \
0              0           0         1         1           True
1              1           0         0         0           True
2              0           0         0         1          False
3              0           0         0         0          False
4              1           0         1         0          False
5              0           0         0         0          False
6              0           0         0         0          False
7              0           0         1         0          False
8              1           0         0         1           True
```

```
9              0           0         0         0           True
10             0           0         0         1          False
11             0           0         0         0           True
12             0           0         0         0           True
13             0           0         0         0          False
14             0           0         0         0          False
15             1           0         0         0          False
16             0           0         0         0           True
17             0           0         0         0           True
18             0           0         0         0           True
19             0           0         0         0           True
20             0           1         0         0          False
21             0           0         0         0           True
22             0           1         0         0          False
23             0           0         0         0           True

    VictoryPoints  BlackHole  Wormhole  Anomalies  Supernova  Nebula  \
0               2      False     False      False      False   False
1               3      False     False      False      False   False
2               2      False     False       True      False   False
3               2      False     False      False      False   False
4               2      False     False      False      False   False
5               3      False     False      False      False   False
6               1      False     False      False      False   False
7               1      False     False      False      False   False
8               1      False     False      False      False   False
9               1      False     False      False      False   False
10              1      False     False       True      False   False
11              1      False     False      False      False   False
12              0      False     False      False      False    True
13              1      False     False      False      False   False
14              1      False     False      False      False   False
15              1      False     False      False      False   False
16              2      False      True      False      False   False
17              0       True     False      False      False   False
18              1      False     False      False      False   False
19              1      False     False      False      False   False
20              1      False     False      False      False   False
21              1      False     False      False      False   False
22              1      False     False      False      False   False
23              1      False     False      False      False   False

    AncientHive  Pulsar
0             0   False
1             0   False
2             0   False
3             0   False
```

```
4              0   False
5              0   False
6              0   False
7              0   False
8              0   False
9              0   False
10             0   False
11             0   False
12             0   False
13             0    True
14             0   False
15             0   False
16             0   False
17             0   False
18             0   False
19             0   False
20             0   False
21             0   False
22             0   False
23             0   False
```

```
[ ]:
```