Exploring Combinatorics Geometrically and with Code

Introduction

Combinatorics is the study of possibilities and arrangements. From a given set, we can determine:

- How many subsets it has.
- How elements can be arranged linearly or in other configurations.

We'll approach combinatorics geometrically and illustrate its principles with programming examples. Additionally, we'll integrate concepts like using astronomical data from Hubble Telescope images to demonstrate practical applications.

Combinatorics Basics: Subsets

Example: A Set of Three Elements `{A, B, C}`

1. **Possible subsets**:
   - `{}` (empty set)
   - `{A}`, `{B}`, `{C}`
   - `{A, B}`, `{A, C}`, `{B, C}`
   - `{A, B, C}`
2. **Counting subsets**: Each element can either be included ("yes") or excluded ("no"), giving 2 choices per element.
   For a set of NNN elements, the total subsets are 2N2^N2N.

MATLAB Example: Generating Subsets

matlab
Copiar código

```
% Define the set
elements = {'A', 'B', 'C'};

% Generate all subsets
n = numel(elements);
subsets = {};
for i = 0:(2^n - 1)
    binary = dec2bin(i, n);
```

```
    subset = elements(logical(binary - '0'));
    subsets = [subsets; subset];
end

% Display subsets
disp('Subsets:');
disp(subsets);
```

Python Example: Generating Subsets
python
Copiar código
```
from itertools import chain, combinations

# Define the set
elements = ['A', 'B', 'C']

# Generate all subsets
def get_subsets(elements):
    return list(chain.from_iterable(combinations(elements, r) for r in
range(len(elements) + 1)))

# Print subsets
subsets = get_subsets(elements)
print("Subsets:")
for subset in subsets:
    print(subset)
```

Geometric Modeling of Subsets

Subsets can be represented geometrically:

- For 0 elements: A single point.
- For 1 element: Two points connected by a line.
- For 2 elements: Four points forming a square.
- For 3 elements: Eight points forming a cube.

Observation: Each additional element doubles the subsets, visually represented by adding a new dimension.

MATLAB Example: Visualizing Subsets as a Cube

matlab

Copiar código

```
% Define points in 3D for subsets of {A, B, C}
points = [
    0, 0, 0; % {}
    1, 0, 0; % {A}
    0, 1, 0; % {B}
    0, 0, 1; % {C}
    1, 1, 0; % {A, B}
    1, 0, 1; % {A, C}
    0, 1, 1; % {B, C}
    1, 1, 1; % {A, B, C}
];

% Plot points
figure;
scatter3(points(:,1), points(:,2), points(:,3), 'filled');
text(points(:,1), points(:,2), points(:,3), ...
    {'{}', '{A}', '{B}', '{C}', '{A, B}', '{A, C}', '{B, C}', '{A, B,
C}'});
xlabel('A'); ylabel('B'); zlabel('C');
grid on;
title('Geometric Representation of Subsets');
```

Python Example: Visualizing Subsets as a Cube

python

Copiar código

```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Define points in 3D for subsets of {A, B, C}
points = [
```

```
    (0, 0, 0), # {}
    (1, 0, 0), # {A}
    (0, 1, 0), # {B}
    (0, 0, 1), # {C}
    (1, 1, 0), # {A, B}
    (1, 0, 1), # {A, C}
    (0, 1, 1), # {B, C}
    (1, 1, 1)  # {A, B, C}
]

# Plot points
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
x, y, z = zip(*points)
ax.scatter(x, y, z, c='b', marker='o')

# Annotate points
labels = ['{}', '{A}', '{B}', '{C}', '{A, B}', '{A, C}', '{B, C}',
'{A, B, C}']
for point, label in zip(points, labels):
    ax.text(point[0], point[1], point[2], label)

ax.set_xlabel('A')
ax.set_ylabel('B')
ax.set_zlabel('C')
plt.title('Geometric Representation of Subsets')
plt.show()
```

Applications in Astronomy: Analyzing Hubble Data

Astronomers often work with subsets of stars, galaxies, or light spectra. By treating these as sets, combinatorial techniques can help model relationships or probabilities.

Example: Subsets of Stars

- Given 4 stars observed by Hubble: {S1,S2,S3,S4}\{S_1, S_2, S_3, S_4\}{S1,S2,S3,S4}
- Compute all possible groupings (subsets) to study properties like brightness combinations.

MATLAB Example: Astronomical Subset Analysis

matlab

Copiar código

```
% Define star data (e.g., brightness values)
stars = {'S1', 'S2', 'S3', 'S4'};
brightness = [2.1, 3.5, 1.8, 4.2];

% Generate subsets
n = numel(stars);
subsets = {};
for i = 0:(2^n - 1)
    binary = dec2bin(i, n);
    subset = stars(logical(binary - '0'));
    subsets = [subsets; subset];
end

% Analyze brightness for each subset
disp('Subset Brightness Analysis:');
for i = 1:length(subsets)
    subset = subsets{i};
    idx = ismember(stars, subset);
    fprintf('Subset: {%s}, Total Brightness: %.2f\n', strjoin(subset,
', '), sum(brightness(idx)));
end
```

Conclusion

This exploration combines combinatorics, geometry, and real-world applications in astronomy. By using programming, we can model abstract mathematical ideas and extend them into practical domains, such as analyzing astronomical datasets.

1. Terminal and non-terminal processing domains:

The idea of terminal domains suggests that we are limited by what our current physics can observe, measure, and model, based on our technological tools and instruments.

Non-terminal domains would be those that are not directly accessible to our current senses or technology, but are still part of reality. In these domains, quantum phenomena or structures could exist that indirectly affect the visible universe.

2. Non-terminal processing in physics:

The concept of "processing" can be understood as the laws and dynamics that govern the evolution of the universe. In a non-terminal domain, the rules could be more general or different from the physical laws we know.

In this context, we could imagine that there are levels of existence or dimensions where the current physical laws do not apply in the same way or are a "projection" of deeper and more abstract rules.

3. The KBC supervoid and the Hubble bubble:

The low-density region (the KBC supervoid) could offer an interesting framework to speculate on how these non-terminal domains could influence the observable universe.

If our galaxy is in a less dense region, that implies that gravitational interactions and other properties of space-time could be different from those in denser regions. This opens questions about whether this bubble is a reflection of physical processes operating from a non-terminal domain.

4. Limited instruments:

As you rightly point out, our ability to explore these phenomena is restricted by current technology and theories. Current instruments are designed to operate within the framework of the terminal domain (observable dimensions and accessible energies).

If there are phenomena associated with non-terminal domains, we would need a new generation of instruments or theoretical approaches that connect these realities.

5. Connection to quantum and processing at invisible scales:

Quantum physics already shows hints of phenomena that have no clear analogy in the macroscopic world, such as entanglement or wave-particle duality.

If we expand these ideas, we could think of scales or dimensions where quantum interactions have properties that are not only invisible, but operate in domains that are not directly connected to the "terminals" of the observable universe.

In essence, what is described suggests that physics could benefit from a broader approach, where the limitations of terminal domains are seen as challenges to explore. At a theoretical level, this involves combining tools from cosmology, quantum physics, and possibly fields such as information or quantum computing to model what might be "out of reach."