DAFT Package Challenge

Visualizing the Scurvy Data Generating Process

DAFT Package Challenge: Beautiful Probabilistic Graphical Models

The Challenge: Make Scurvy History Beautiful

Core Question: How can we use the DAFT package to create visually appealing Directed Acyclic Graphs (DAGs) that tell a compelling data story?

The Challenge: You'll reproduce and enhance the scurvy DAG from the course materials, learning how to customize colors, shapes, and styling to make your probabilistic graphical models more attractive and informative.

Learning Objectives: By the end of this 30-minute challenge, you'll be able to: - Install and import the DAFT package - Create basic probabilistic graphical models - Customize node colors, shapes, and styling - Use the DAFT documentation effectively

Background: The Scurvy Story

Scurvy was a devastating disease that affected sailors on long voyages. The cure was discovered in 1747, but due to a misunderstanding about the cause, the cure was lost for over 150 years. The story involves three different understandings of the data generating process:

1. **1747 Understanding:** Lemons prevent scurvy (correct!)
2. **Misguided Belief:** Acid kills bacteria that causes scurvy (wrong!)
3. **1928 Understanding:** Vitamin C prevents scurvy (the real mechanism)

Your Mission: Create a Beautiful Scurvy DAG

Your task is to recreate the 1928 understanding of scurvy using DAFT, but make it visually stunning with custom colors and styling.

Step 1: Environment Setup

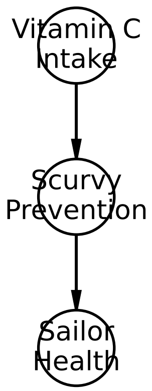
First, let's install the DAFT package in your virtual environment:

```
pip install 'daft-pgm'
```

Step 2: Basic DAG Creation

Start by creating a simple DAG that shows the relationship between Vitamin C and scurvy prevention:

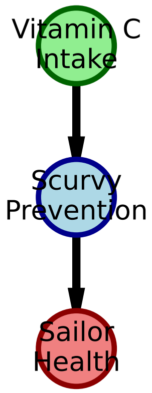
```
import daft
import matplotlib.pyplot as plt

# Create the PGM object
pgm = daft.PGM(dpi=150)

# Add nodes for our scurvy model
pgm.add_node("vitamin_c", "Vitamin C\nIntake", 1, 2)
pgm.add_node("scurvy", "Scurvy\nPrevention", 1, 1)
pgm.add_node("health", "Sailor\nHealth", 1, 0)

# Add edges to show relationships
pgm.add_edge("vitamin_c", "scurvy")
pgm.add_edge("scurvy", "health")

# Render the basic DAG
pgm.render();
```



Step 3: Make It Beautiful

Now let's enhance the visual appeal by customizing colors and styling. Use the DAFT documentation to explore different `plot_params` options:

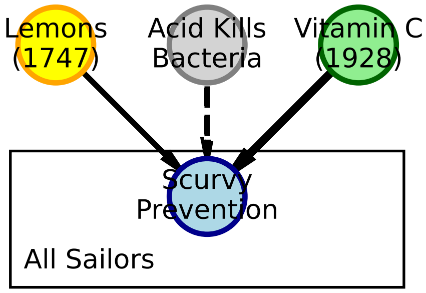
```
# Create a new, more beautiful PGM
pgm_beautiful = daft.PGM(dpi=150, alternate_style="outer")

# Add nodes with custom styling
pgm_beautiful.add_node("vitamin_c", "Vitamin C\nIntake", 1, 2,
                       plot_params={'facecolor': 'lightgreen', 'edgecolor': 'darkgreen', 'li

pgm_beautiful.add_node("scurvy", "Scurvy\nPrevention", 1, 1,
                       plot_params={'facecolor': 'lightblue', 'edgecolor': 'darkblue', 'line

pgm_beautiful.add_node("health", "Sailor\nHealth", 1, 0,
                       plot_params={'facecolor': 'lightcoral', 'edgecolor': 'darkred', 'line

# Add edges with custom styling
pgm_beautiful.add_edge("vitamin_c", "scurvy", plot_params={'color': 'green', 'linewidth': 3}
pgm_beautiful.add_edge("scurvy", "health", plot_params={'color': 'blue', 'linewidth': 3})

# Render the beautiful DAG
pgm_beautiful.render();
```



Step 4: Add Historical Context

Let's create a more complex DAG that shows the historical progression of understanding:

```
# Create a historical DAG showing the evolution of understanding
pgm_historical = daft.PGM(dpi=150, alternate_style="outer")

# 1747 Understanding (correct but incomplete)
pgm_historical.add_node("lemons_1747", "Lemons\n(1747)", 0.5, 2,
                        plot_params={'facecolor': 'yellow', 'edgecolor': 'orange', 'linewidth

# Misguided belief
pgm_historical.add_node("acid_belief", "Acid Kills\nBacteria", 1.5, 2,
                        plot_params={'facecolor': 'lightgray', 'edgecolor': 'gray', 'linewid

# 1928 Understanding (complete)
pgm_historical.add_node("vitamin_c_1928", "Vitamin C\n(1928)", 2.5, 2,
                        plot_params={'facecolor': 'lightgreen', 'edgecolor': 'darkgreen', 'li

# Common outcome
pgm_historical.add_node("scurvy_prevention", "Scurvy\nPrevention", 1.5, 1,
                        plot_params={'facecolor': 'lightblue', 'edgecolor': 'darkblue', 'line

# Add edges
pgm_historical.add_edge("lemons_1747", "scurvy_prevention", plot_params={'color': 'green', 'l
pgm_historical.add_edge("acid_belief", "scurvy_prevention", plot_params={'color': 'red', 'lin
pgm_historical.add_edge("vitamin_c_1928", "scurvy_prevention", plot_params={'color': 'blue',

# Add a plate to show this affects many sailors
pgm_historical.add_plate([0.2, 0.5, 2.6, 0.8], label="All Sailors", shift=-0.1)

# Render the historical DAG
pgm_historical.render();
```



Challenge Extensions

Option 1: Color Psychology

Experiment with different color schemes that convey the right emotions: - Use warm colors (reds, oranges) for problems - Use cool colors (blues, greens) for solutions - Use neutral colors (grays) for misconceptions

Option 2: Shape Customization

Try different node shapes and sizes: - Use `aspect` parameter to control node width - Use `scale` parameter to control node size - Experiment with `alternate=True` for different shapes

Option 3: Advanced Styling

Explore more advanced customization: - Add custom fonts with `fontsize` parameter - Use `plot_params` to customize every visual aspect - Add plates to show repeated structures

Key DAFT Parameters to Explore

Based on the DAFT documentation, here are key parameters to experiment with:

Node Parameters: - `plot_params`: Dictionary of matplotlib parameters for styling - `aspect`: Controls node width (default: 1.0) - `scale`: Controls node size (default: 1.0) - `fontsize`: Text size in the node - `alternate`: Use alternative node shape (True/False)

Edge Parameters: - `plot_params`: Dictionary of matplotlib parameters for edge styling - `color`: Edge color - `linewidth`: Edge thickness - `linestyle`: Edge style (‘-’, ‘–’, ‘:’, etc.)

PGM Parameters: - `dpi`: Resolution for rendering - `alternate_style`: Style for alternate nodes (“inner” or “outer”)

Submission Requirements

Create a Jupyter notebook or Quarto document that includes:

1. **Installation code** for the DAFT package
2. **At least three different DAGs** showing your creativity
3. **Comments explaining** your design choices
4. **A brief reflection** on what you learned about data visualization

Resources

- DAFT Documentation
- DAFT Examples
- Matplotlib Colors

Reflection Questions

1. How do different color choices affect the emotional impact of your DAG?
2. What makes a DAG “beautiful” versus just functional?
3. How might you use DAFT in your future data analysis projects?

This challenge should take approximately 30 minutes to complete. Focus on experimentation and creativity rather than perfection!