3D interactive image generation and saving methods

Method 1 Draw and save with R language.

1. Start by organizing the data into a wide format.
2. Use **cld3d** function of **kml3d** package in R to convert the data.which can help you create a 3D plot using your data.

```r
library(tidyr)
set.seed(123)
test_data <- data.frame(
  stu_id = rep(1:100,each = 10),
  title_time = rep(1:10,100),
  start_time = runif(1000, min = 0, max = 10),
  submission = runif(1000, min = 0, max = 10),
  cost_time = runif(1000, min = 0, max = 10),
  score = runif(1000, min = 0, max = 10)
)
df_wide <- as.data.frame(pivot_wider(test_data, names_from = "title_time",
                       names_sep = "_",
                       values_from = c("start_time", "submission", "cost_time", "score")))
head(df_wide,5)
```

```r
library(kml3d)
cldPreg <- cld3d(df_wide, timeInData = list(start_time = 2:11,
                                            submission= 12:21,
                                            cost_time = 22:31,
                                            score = 32:41))
option = parKml3d(saveFreq = 2,nbCriterion = 15)
kml3d(cldPreg,nbClusters = 3:10,nbRedrawing = 20,toPlot="none",parAlgo = option)
```

Option 1: Save in HTML Format

3. Visualize the data using the **plotMeans3d** function.
4. After generating the 3D plot, you can save it as an interactive HTML file using the **rglwidget** function from the **rgl** package. This will allow you to interact with the 3D plot in a web browser.

```r
library(rgl)
plotMeans3d(cldPreg,3)
rglwidget <- rglwidget()
#saveWidget(rglwidget,"3dplot.html",selfcontained = T)
```

Option 2: Save as ASY Format

3. **Use the `kml3d` function to cluster the data.**

   Begin by using the `kml3d` function to cluster your data. This function will group similar data points together based on the specified variables, preparing the data for 3D visualization.

4. **Visualize and Save the Plot**.

   After clustering, use the **`plot3dPdf`** function to generate a 3D plot of the clustered data. This function creates a detailed 3D representation, showing how the data points are grouped in three-dimensional space. Once the plot is generated, use the **`drawScene.rgl`** function to visualize it interactively, allowing you to rotate, zoom, and explore the data from different angles. Finally, save the plot as an ASY file using the **`saveTrianglesAsASY`** function, a vector-based format suitable for high-quality rendering and further manipulation

```
### Creation of the scene
scene <- plot3dPdf(cldPreg, 3)
drawScene.rgl(scene)
### Export in '.asy' file
# saveTrianglesAsASY(scene,"3dplot.asy")
```

```
#htmltools::includeHTML("3dplot.html")
```

Option 3: Convert ASY Format to PRC Format

1. **Select the ASY File to Convert.**

   Begin by selecting the ASY file that you wish to convert to PRC format. Ensure that the file is located in a directory on your computer where you can easily access it.

2. **Open the Windows Terminal in the Relevant Directory**.

   Navigate to the directory where your ASY file is saved. Open the **Command Prompt** (or **Windows Terminal**) in that directory.

3. **Enter the Conversion Command**.

   Once the terminal is open, type the following command to convert the ASY file into PRC format. Ensure that you replace **`yourfile.asy`** with the name of your actual ASY file:

```
asy -f prc yourfile.asy
```

4. **Check the Output.**

   After running the command, you should find the resulting PRC file in the same directory.

Additional Information : Save 3d interactive images in pdf

Insert with Adobe Acrobat Pro software

1. Download and install Adobe Acrobat Pro.
2. Launch Adobe Acrobat Pro and open the PDF file where you want to insert the 3D image.
3. Click on “All tools” in the menu bar, then select the “Rich Media” option.

4. Choose “Add 3D,” then drag the cursor on the page to draw a rectangular frame to define the insertion location.
5. Select the 3D file (in PRC format) you want to insert, then click “OK.”
6. After inserting the 3D image, save the modified PDF file.

You can refer to the official tutorial:<https://helpx.adobe.com/acrobat/using/adding-3d-models-pdfs-acrobat.html>

Insert with LaTeX software

1. Open the LaTeX software and confirm the PRC format file to be inserted.
2. Insert the following code:

```
\documentclass[a4paper]{article}
\usepackage{media9}  % Import the package for embedding multimedia
\usepackage{tikz}    % Import the package for drawing graphics

\begin{document}

    \begin{figure}[!htb]  % Create a figure environment
        \centering  % Center the figure on the page

        % Embed the 3D model with the specified parameters
        \includemedia[
            3Dtoolbar,  % Display the 3D toolbar
            3Dmenu,     % Display the 3D menu
            3Dcoo=90 45 45,  % Set the 3D view angle (rotation)
            3Droo=500,  % Set the distance of the 3D view
            3Dlights=CAD,  % Set the 3D lighting effect to CAD mode
        ]{
            \tikz\node[draw,minimum width=1\linewidth,minimum height=0.55\linewidth,
          opacity=1]{\ttfamily(cylinder.u3d)};
          % Use TikZ to draw a frame and specify the U3D file
        }{merged_output167.prc}  % This is the PRC format file to be inserted

        \caption{An Example}  % Add a caption to the figure
        \label{3d_plot(cost_time_+submission_).prc}
    % Set a label for the figure for referencing
    \end{figure}

\end{document}
```

Method 2: Drawing and Saving in ASY Format with Asymptote Software

1. **Organize the Data**: Arrange the data in a wide format for proper drawing generation.
2. **Create a Text File**: Use a text editor to write the ASY drawing code, defining the shapes, axes, and other elements.

3. **Save as ASY Format**: Change the file extension from **.txt** to **.asy** to indicate it's an ASY file.
4. **Open and View**: Double-click the **.asy** file to open it in Asymptote software, which will generate the graphic.

```
// Import the three library and support package esvect, set the drawing size to 10 cm,
// and define the perspective projection angle.
import three;
usepackage("esvect");
size(10cm);
currentprojection = perspective(0, -1, 0);

// Draw a box centered at origin O with dimensions 4*X, 2*Y, and 2*Z, using black color.
draw(box(O, (4*X + 2*Y + 2*Z)), black);

// Manually draw tick marks along the x-axis and add labels
for (int i = 1; i <= 10; ++i) {
    draw((i*(4/9) - (4/9), 0, 0) -- (i*(4/9) - (4/9), 0, 0.1), black);  // Draw tick mark
    label("$" + string(i) + "$", ((i - 1) * (4/9), 0, -0.1));  // Add tick label
}

// Add labels for axes
label("$Time$", (2, -0.2, -0.3));
label("$Start\\_time$", (-0.8, 0.8, -0.2));
label("$Cost\\_time$", (-1, 0.2, 1));

// Define points along a trajectory, corresponding to 10 points on the x-axis
// Data for mean trajectory 1
triple[] points1 = {
    (0, 1.2854, 1.0317), (0.444, 1.2733, 1.0636), (0.889, 1.1557, 0.9849),
    (1.333, 0.9393, 0.7098), (1.778, 0.6901, 0.7400), (2.222, 1.1934, 1.1598),
    (2.667, 0.7585, 0.7544), (3.111, 1.3180, 1.2271), (3.556, 0.8518, 1.2682),
    (4, 0.6302, 0.5948)
};

// Set parameters for red pen and draw the trajectory
pen pen1 = red + linewidth(2.5) + opacity(1);
for (int i = 1; i < points1.length; ++i) {
    draw(points1[i - 1] -- points1[i], pen1);
    // Connect points with red trajectory line
}

// Data for mean trajectory 2
triple[] points2 = {
    (0, 0.9273, 1.1582), (0.444, 0.9746, 1.2064), (0.889, 1.0767, 1.3062),
    (1.333, 1.2769, 1.2268), (1.778, 1.3901, 1.3616), (2.222, 1.1137, 1.1303),
    (2.667, 1.3831, 1.3848), (3.111, 0.8136, 0.9386), (3.556, 1.2295, 0.9569),
    (4, 1.1739, 1.1957)
};

// Set parameters for blue pen and draw the trajectory
pen pen2 = blue + linewidth(2.5) + opacity(1);
for (int i = 1; i < points2.length; ++i) {
    draw(points2[i - 1] -- points2[i], pen2);
    // Connect points with blue trajectory line
}

// Data for mean trajectory 3
triple[] points3 = {
    (0, 0.7438, 0.7713), (0.444, 0.7015, 0.6749), (0.889, 0.7201, 0.6494),
    (1.333, 0.7397, 1.0765), (1.778, 0.9036, 0.8776), (2.222, 0.6303, 0.6506),
    (2.667, 0.8296, 0.8324), (3.111, 0.7397, 0.6987), (3.556, 0.9020, 0.7289),
    (4, 1.1341, 1.1504)
};

// Set parameters for green pen and draw the trajectory
pen pen3 = green + linewidth(2.5) + opacity(1);
for (int i = 1; i < points3.length; ++i) {
    draw(points3[i - 1] -- points3[i], pen3);
    // Connect points with green trajectory line
}

// Sample trajectory data
triple[] samplePoints = {
    (0, 1.1081, 0.4501), (0.444, 1.0595, 0.7750), (0.889, 0.8090, 1.1456),
    (1.333, 0.9284, 0.9881), (1.778, 0.8919, 0.3366), (2.222, 0.9103, 1.3661),
    (2.667, 1.1143, 0.6339), (3.111, 1.6299, 1.2540), (3.556, 1.7876, 1.8351),
    (4, 0.6861, 0.8901)
};

// Set parameters for light red pen and draw the sample trajectory
pen samplePen = red + linewidth(0.3) + opacity(0.1);
for (int i = 1; i < samplePoints.length; ++i) {
    draw(samplePoints[i - 1] -- samplePoints[i], samplePen);
    // Connect points with light red trajectory line
}
```

Note:

HTML, **ASY**, and **PRC** are three common file formats that can store interactive images, each with its own unique advantages and uses in different application scenarios and fields.

HTML format: The HTML format allows users to interact with 3D objects in a browser by rotating, zooming, and panning. It enables the display and interaction of 3D graphics through WebGL or other JavaScript libraries (such as Three.js), making it suitable for graphic presentation in online environments. It is widely used in website development, online reports, e-commerce, and virtual presentations.

ASY format: The ASY (Asymptote) format is primarily used for generating high-precision mathematical and technical graphics. It is particularly suited for academic papers, technical reports, and LaTeX documents, allowing the display of mathematical formulas, geometric figures, and data visualizations in high-quality vector graphics. ASY enables users to draw complex graphics using code, providing precise control, and is widely used in research within technical fields.

PRC format: The PRC format is primarily used for storing complex 3D models and is widely applied in 3D modeling, CAD , and product design. It can precisely store the geometric data and physical properties of models, and supports data exchange and collaboration in large-scale engineering projects. Additionally, PRC format can be embedded in PDF files, making it convenient to display 3D models in design and engineering documents. As a result, it has significant applications in industrial design, architectural modeling, manufacturing, and 3D printing.