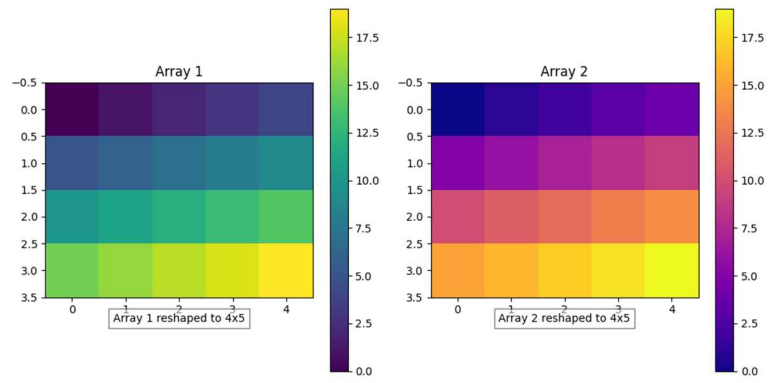
ID = 0
Score vis = 30
Score task = 90
Score human = 80

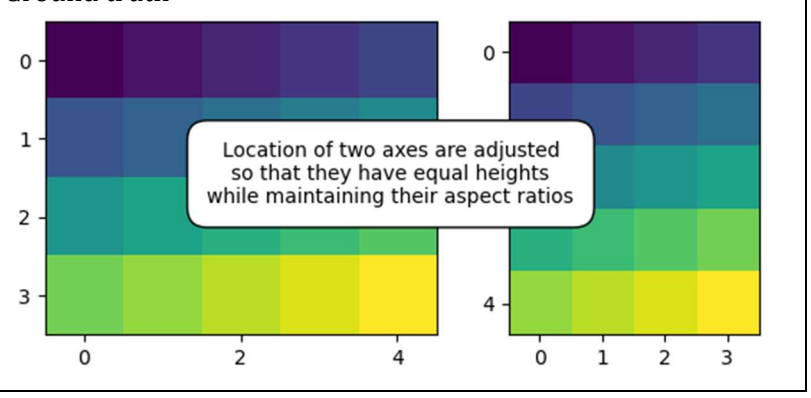
Task: Create two subplots, and in each subplot, present one of the DataFrame's columns as a two-dimensional image. Each 2D image will represent the reshaped data array associated with each column, ensuring that each entry in the array corresponds to a specific pixel intensity in the image. The axes in the graphics should be adjusted such that they have equal heights while still preserving their aspect ratios.

Style: Adjust the location of the two axes so that they are of equal height and have their specific aspect ratios maintained. Include annotations or text boxes that explain the layout adjustment, for clarity and convenience in understanding the represented data characteristics. Set stylistic features such as color maps, interpolation methods for smoother visual appearance, and box styles for annotations.

Generated



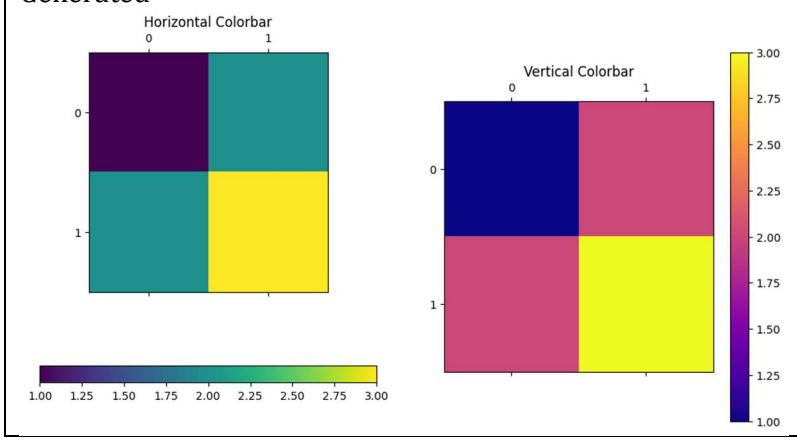
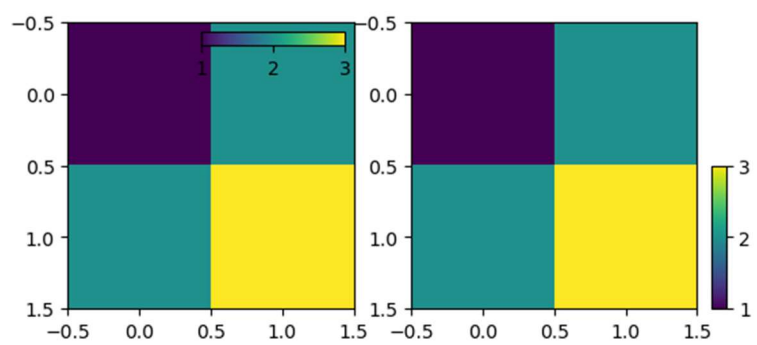
Ground truth



ID = 1
Score vis = 90
Score task = 95
Score human = 70

Task: The task involves creating a plot with two subplots side by side, each displaying a matrix representation of the 'z' values from the dataframe. Each subplot should show the matrix using different color scaling to represent varying values. Additionally, both plots should include a colorbar: one horizontal on the top right of the first plot, and one vertical on the right side of the second plot.

Style: For styling, the plots should have colorbars integrated in specific positions relative to the respective plots. The first subplot's colorbar should be at the top and horizontal, while the second's should be on the right side and vertical. The main visualizations should clearly display the gradation in colors corresponding to the interval of 'z' values. Each subplot should maintain appropriate labels or ticks that reflect the value scale for the 'z' column.

Generated	Ground truth
	

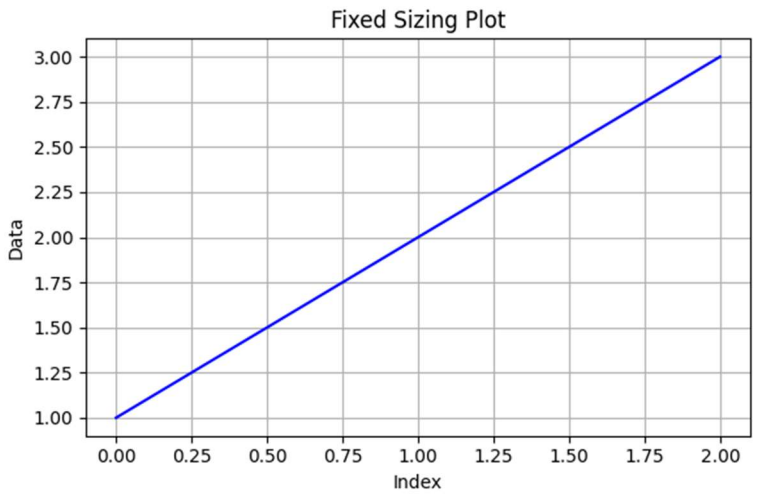
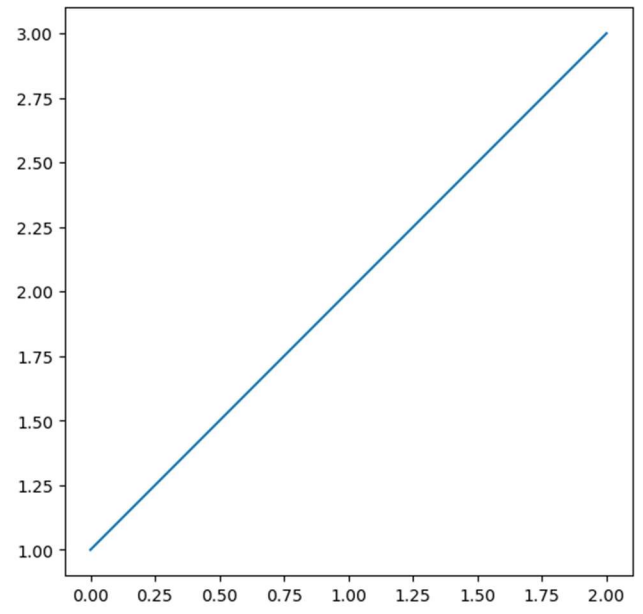
ID = 2
Score vis = 80
Score task = 90
Score human = 90

Task:

- Create two plots using the data from the DataFrame.
- Each plot should feature data from the 'Data' column plotted against its index.
- The first plot should use fixed sizing for all axes to maintain proportional dimensions regardless of the data or its range.
- The second plot should incorporate a combination of fixed and scaled sizing to adjust the plot dynamically based on the data content or frame size.

Style:

- Both plots should display a clean, simple line graph with a blue line to represent the data points.
- Axes should be minimal and styled to ensure clarity without excessive details that might clutter the visualization.
- The overall aesthetics should prioritize clear visualization of trends over decorative graphics.

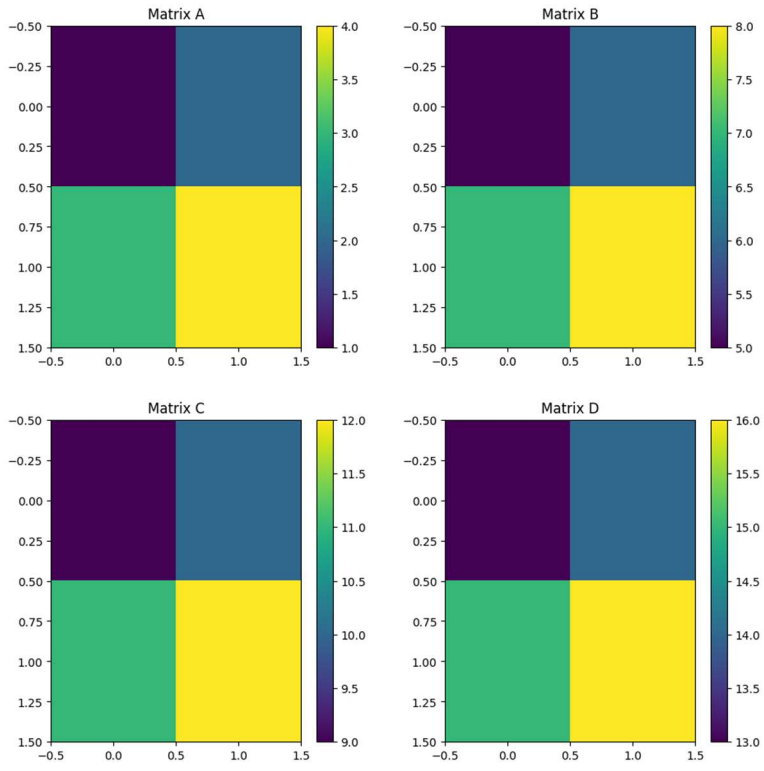
Generated	Ground truth
	

ID = 3
Score vis = 30
Score task = 90
Score human = 100

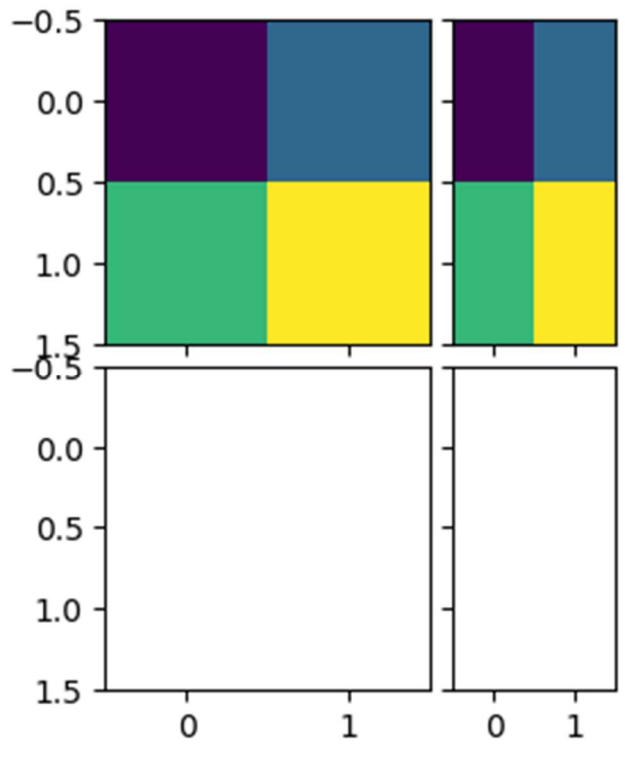
Task: Create visual plots with a 2x2 grid layout, where the elements of the first two columns of the DataFrame are reshaped into 2x2 matrices and displayed in the respective grids. Each individual plot should share axis settings and maintain an assigned aspect ratio for uniformity.

Style: fine appropriate padding between plots for clear visual separation. Ensure that each sub-plot adjusts its aspect ratio dynamically based on its position in the grid to enhance distinctiveness. Maintain consistent axis scales across all plots for a unified look.

Generated



Ground truth

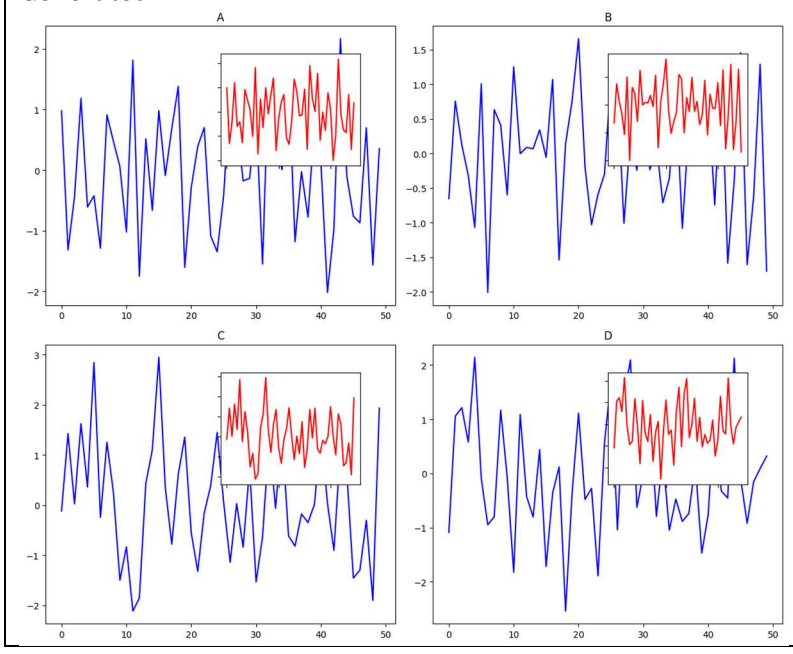


ID = 4
Score vis = 80
Score task = 95
Score human = 100

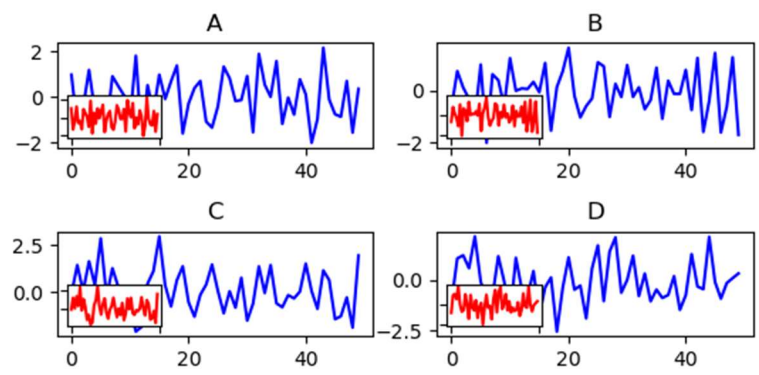
Task:
Construct a figure divided into four subplots arranged in a 2x2 grid. Each subplot should display a line plot of one of the dataframe columns against its index. Overlay a smaller inset plot within each main subplot, focusing on a specific area of the line plot, using a contrasting color to distinguish it from the main plot. Each subplot should have its corresponding column label as the title.

Style:
For the main plots in each subplot, use a consistent color (e.g., blue) and style for the lines. The insets should additionally feature a contrasting color (e.g., red) for the lines but without tick labels to maintain clarity and focus on the data representation. Adjust the layout to ensure that all elements are well-spaced and visually appealing.

Generated



Ground truth

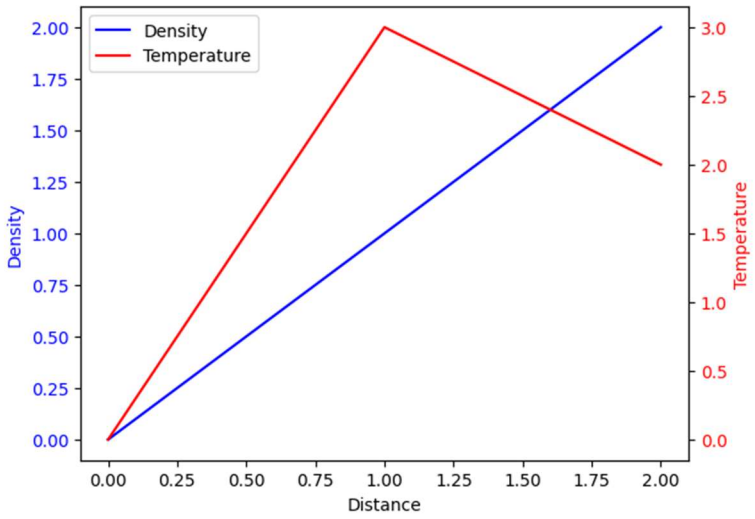


ID = 5
Score vis = 100
Score task = 95
Score human = 100

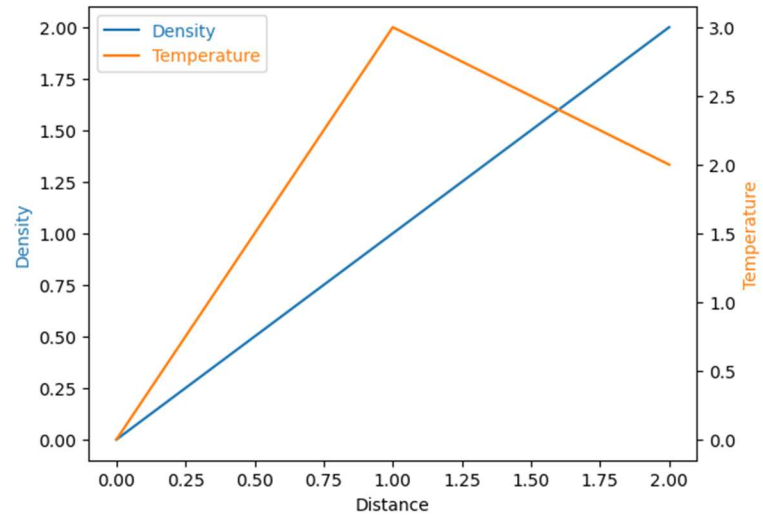
Task: Construct a line plot where Distance is plotted along the X-axis, and both Density and Temperature are plotted along two separate Y-axes. Density and Temperature should each be represented by a uniquely colored line, and each axis should contain an appropriate label for its corresponding data variable.

Style: Fine-tune the appearance of the plot by setting specific colors for the axis labels and legend text to match the colors of the lines they describe, enhancing readability and visual appeal. This customization involves adjusting color properties of labels and legend entries.

Generated



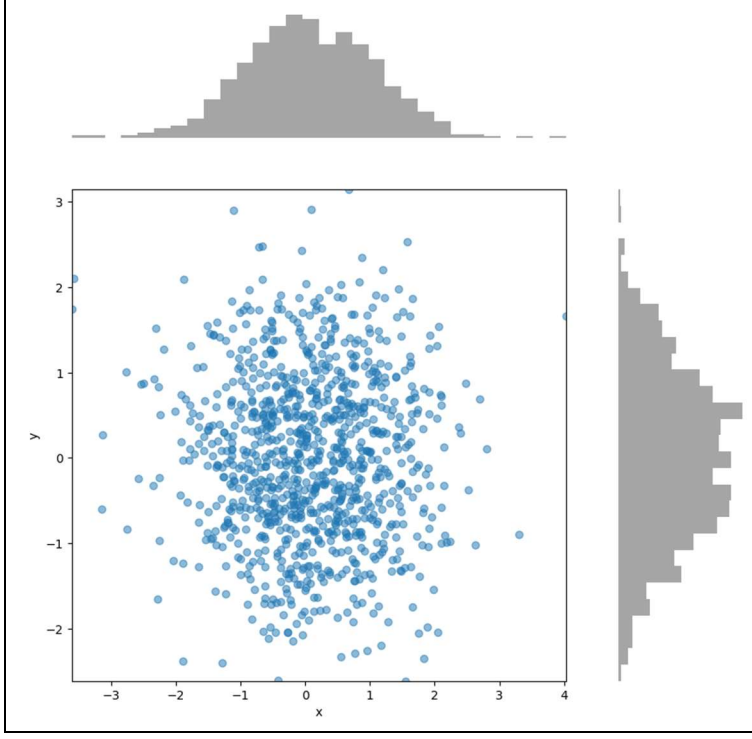
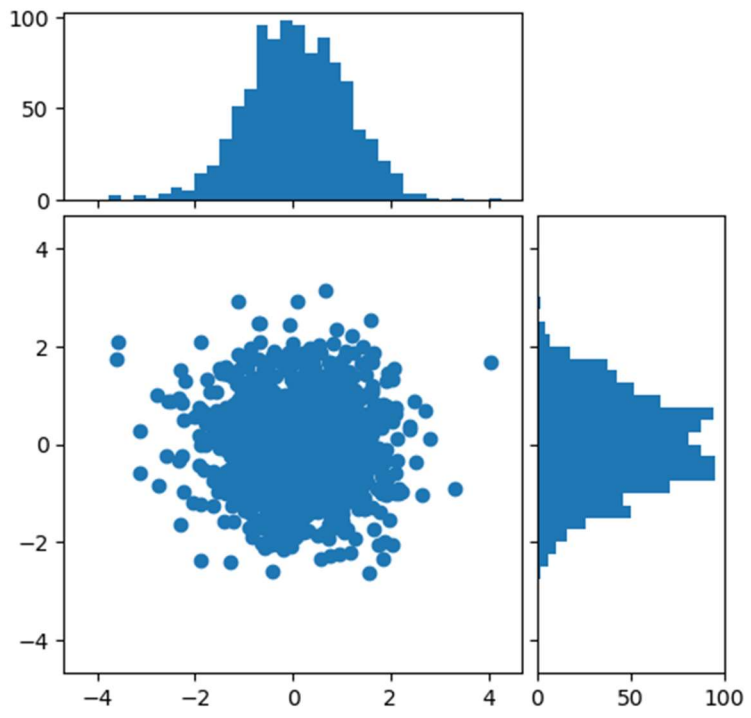
Ground truth



ID = 6
Score vis = 70
Score task = 95
Score human = 100

Task: The task requires generating a central scatter plot of the 'x' and 'y' data points to visualize their distribution. Additionally, histogram plots of these variables will be placed adjacent to the scatter plot: one above to show the distribution of 'x' values and another to the right to show the distribution of 'y' values. The histograms and scatter plot should share axis limits to align their scales and provide a cohesive visual analysis.

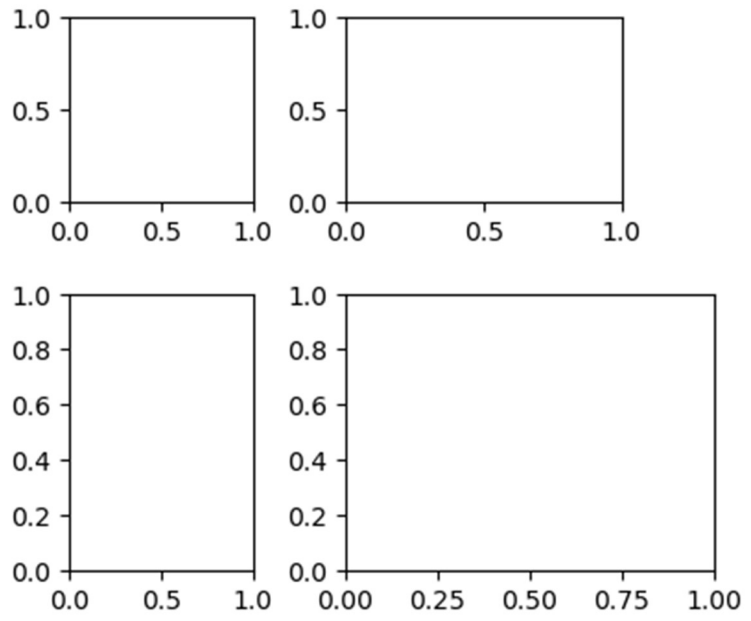
Style: The style of the plot should ensure the scatter plot is proportionate and square in shape to maintain scale accuracy between the 'x' and 'y' axes. Histograms need to be aligned such that they directly correspond to their respective axis on the scatter plot, with shared axis settings to ensure a unified view and axis limits. Adjustments to the visibility of tick labels on the histograms will be necessary to maintain a clean and focused display.

Generated	Ground truth
	

ID = 7
Score vis = 0
Score task = 0
Score human = 0

Task: description: The plot will consist of four individual plot areas arranged in a 2x3 grid based on the proportions specified in a DataFrame. The first row of the DataFrame determines the widths of the plots, and the second row determines the heights. The grids should be dynamically sized based on these values, adjusting even if the DataFrame values change.

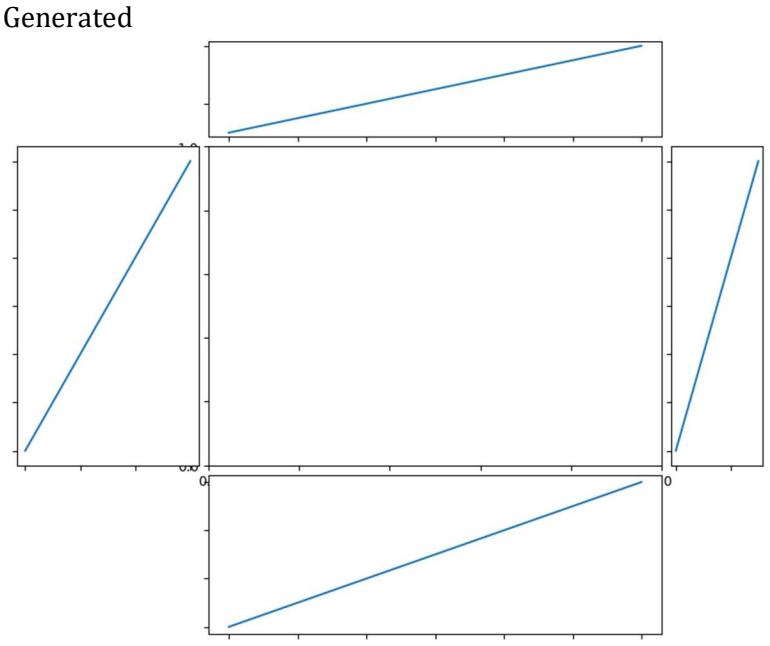
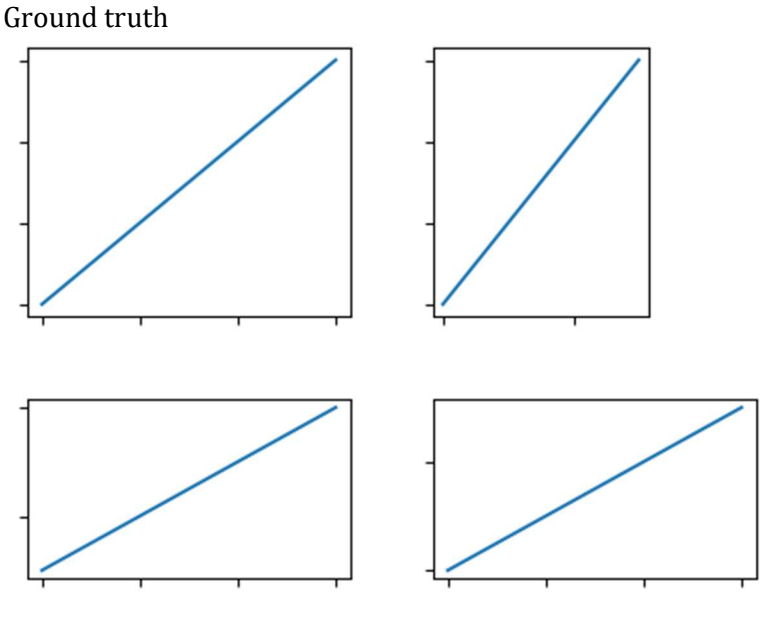
Style: description: The plot should appear as four distinct axes within a single figure window. Each plot will be labelled distinctly and positioned according to a specific layout that uses horizontal and vertical dimensions controlled by the values extracted from the DataFrame. The positioning and sizing of these plots must reflect the proportional values (or fixed sizes) derived from the DataFrame's specified columns.

Generated	Ground truth
	

ID = 8
Score vis = 70
Score task = 90
Score human = 70

Task: description: The plot to be created will involve creating a figure with four separate axes arrangement using a custom divider. Each axis will plot one of the dataframe columns (data1, data2, data3, data4). The axes are arranged in a layout that has varying size ratios both horizontally and vertically.

Style: description: Each plot should be devoid of tick labels on both the x and y axes. The plot design involves grid-based positioning of the axes without adhering to standard rectangular alignment by using scaled and fixed sizing segments.

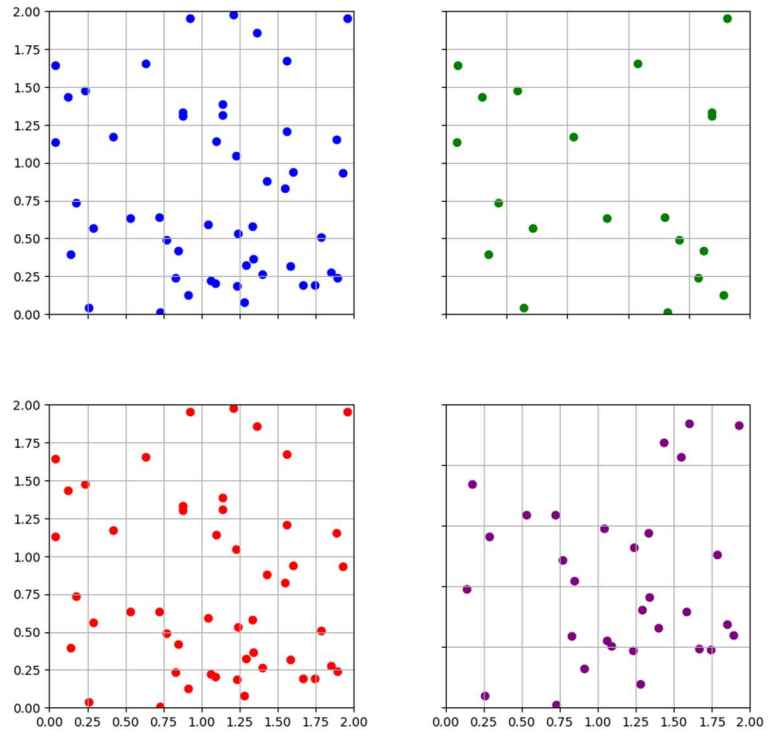
Generated	Ground truth
	

ID = 9
Score vis = 30
Score task = 90
Score human = 100

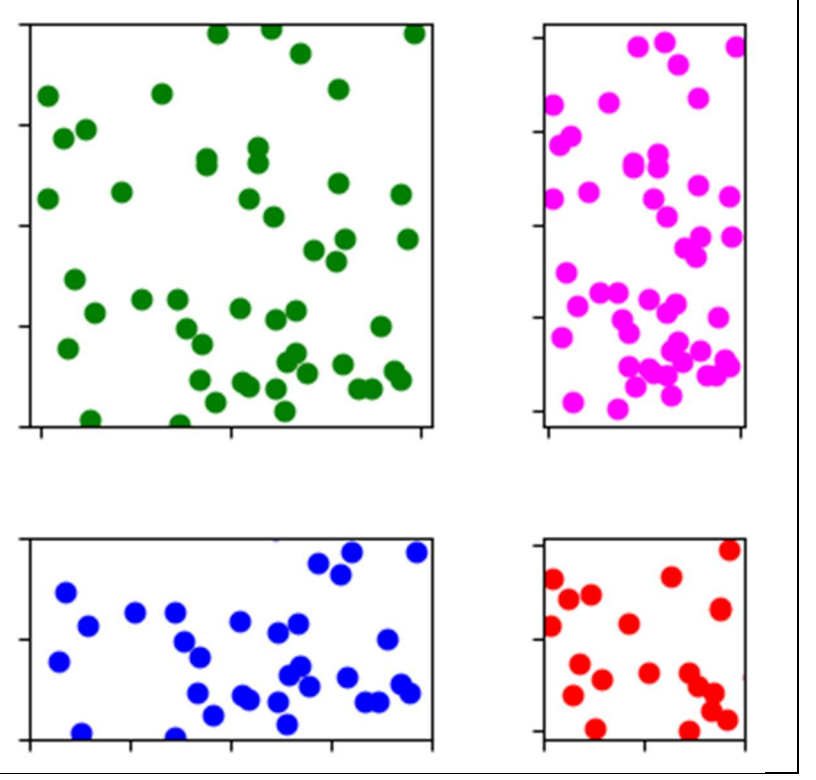
Task: The plot should consist of four subplots arranged in a 2x2 grid. Each subplot will display a scatter plot of the 'x' and 'y' columns from the DataFrame. Different subplots should feature different marker colors for distinction.

Style: Each subplot should have specific axis limits set to clearly define different ranges on each plot. The top-right subplot should showcase a limit on the x-axis up to 1, while the bottom-left should limit the y-axis up to 2, with other subplots adjusted similarly. Use a grid layout manager to distribute the subplots evenly within the plot figure, and aspect ratios should be managed to ensure the plots are square-shaped. Adjust subplot spacing and disable axis labels for a cleaner look.

Generated



Ground truth

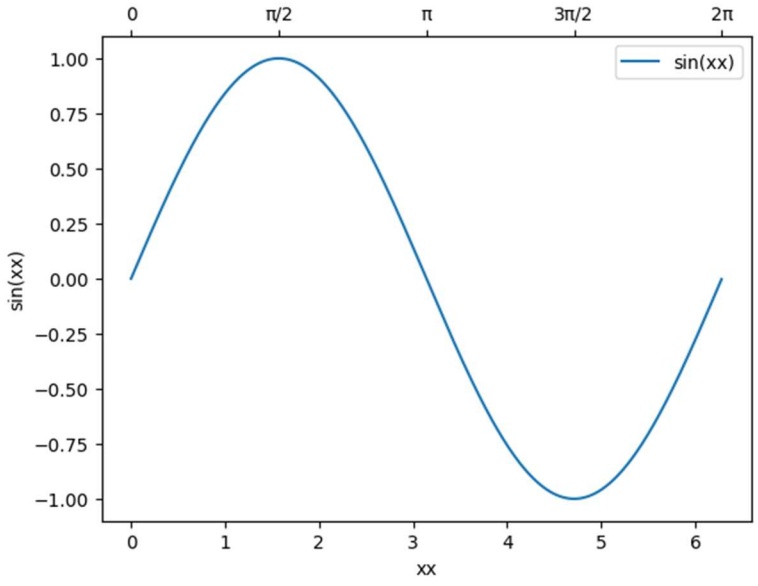


ID = 10
Score vis = 90
Score task = 95
Score human = 100

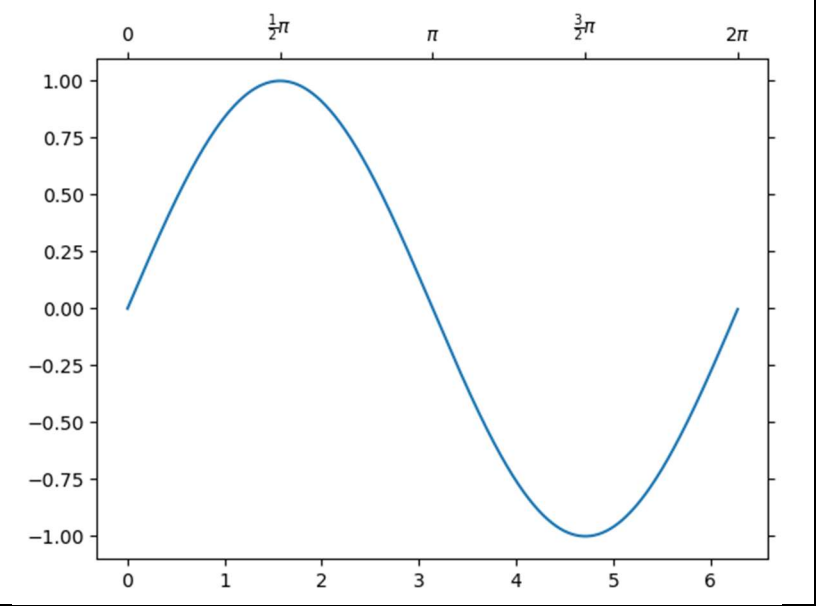
Task: The script should generate a line plot. The main plot will use the 'xx' column for the x-axis and 'sin_xx' column for the y-values. Additionally, overlay the plot with a secondary x-axis that represents the angle values in radians with specific labels like 0, π/2, π, 3π/2, and 2π, enhancing the readability of radian measures for trigonometric functions.

Style: The primary line plot should be clear and uncluttered with visible axes labels. The secondary x-axis, positioned at the top, should display the radian values distinctly while the labels on the right side of the plot should be hidden to maintain visual simplicity. The ticks and labels on the top x-axis should precisely correspond to standard radian values, improving interpretative clarity of the plot regarding trigonometric relationships.

Generated



Ground truth

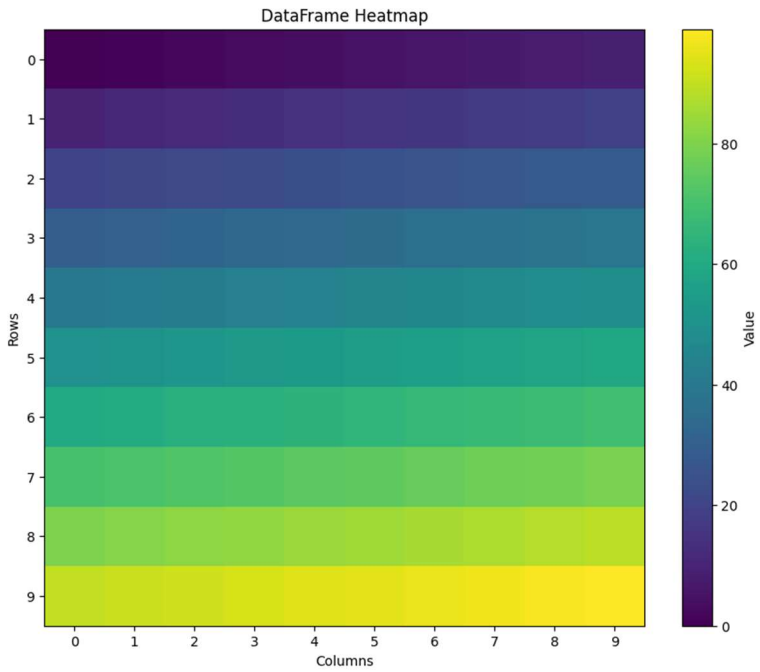


ID = 11
Score vis = 95
Score task = 100
Score human = 100

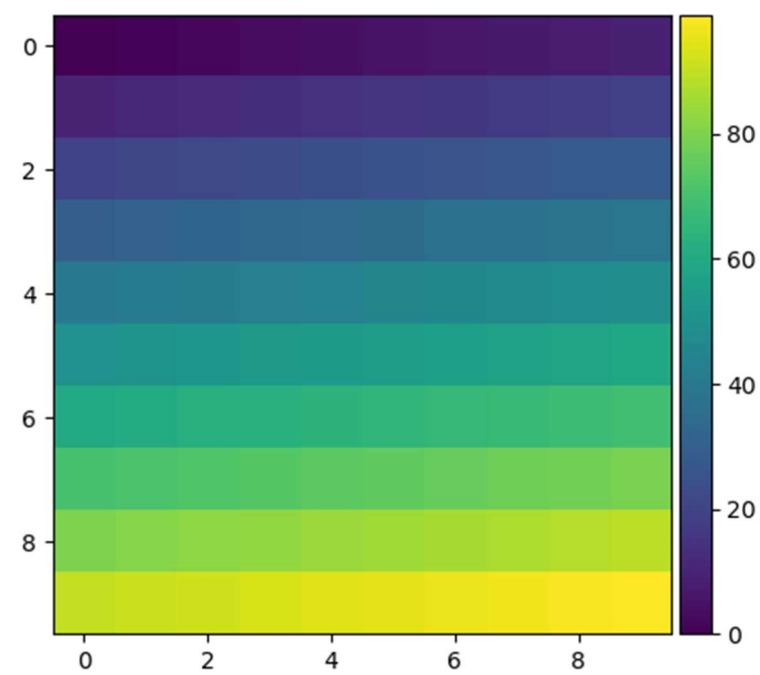
Task: The code should generate a visual representation of the DataFrame as an image plot. Each cell of the plot corresponds to a value in the DataFrame, with different colors representing different data values. A color bar should be added to the right of the plot to act as a legend, indicating the data value corresponding to each color.

Style: The plot will have a color gradient where colors change according to the values represented. The color bar on the right will help in gauging the data values based on the color shades shown in the plot. Elements like axis visibility, color palette, and layout specifications will be controlled to enhance visual clarity and appeal.

Generated



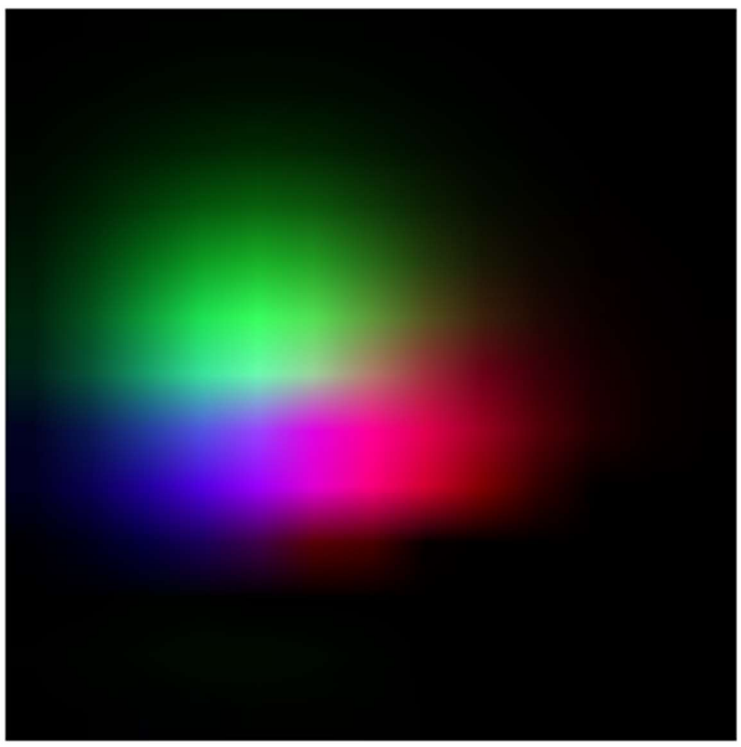
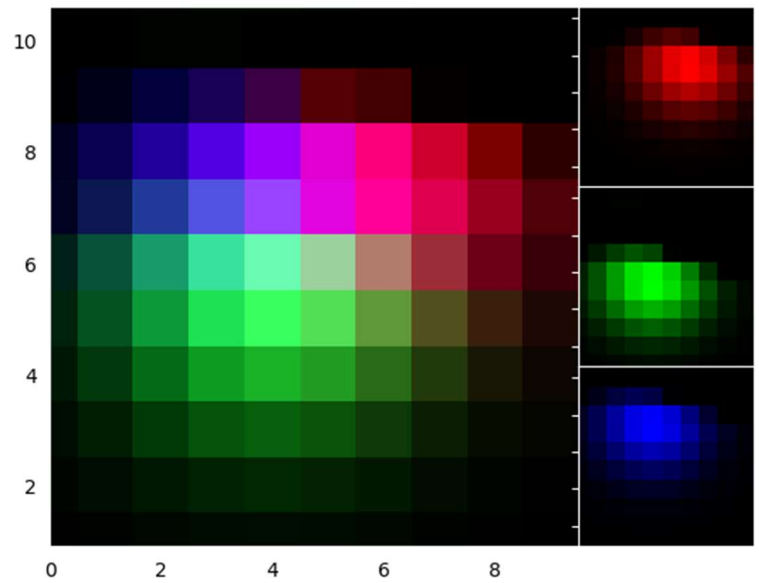
Ground truth



ID = 12
Score vis = 30
Score task = 90
Score human = 75

Task: Create a plot that visualizes the RGB values from the DataFrame as a color image. The image should be formed by reshaping each color component into a 13x13 grid, where each cell represents a combined RGB value. The overall goal is to display these colors as a cohesive plot, conveying how these separate color intensities blend together.

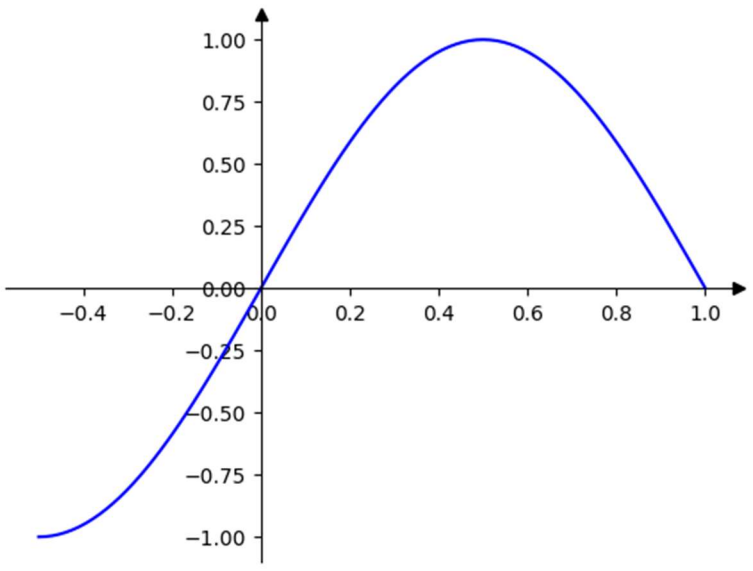
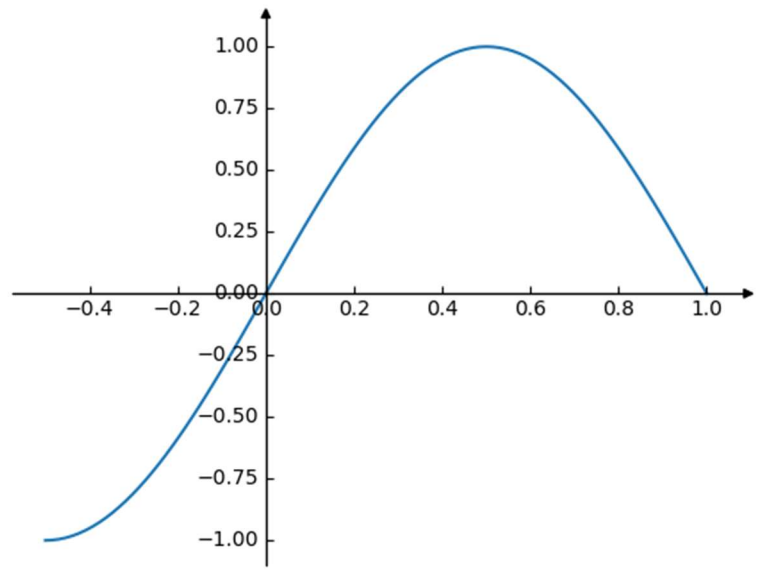
Style: In addition to the main RGB visualization, adjust the plotting area for optimal viewing. Set specific limits for the axes to focus on key areas, ensuring that the entire color range is effectively displayed. Choose a style of interpolation that minimizes pixelation, maintaining a smooth transition between colors, and set the origin of the plot appropriately for standard image display conventions.

Generated	Ground truth
	

ID = 13
Score vis = 100
Score task = 95
Score human = 100

Task: Create a 2D line plot using the dataframe's 'x' and 'y' columns. Display axes with arrows highlighting the positive direction, originating from the axis intersection at zero. Ensure that only the essential axes are visible, specifically the ones through the origin, to emphasize the curve in relation to these axis points.

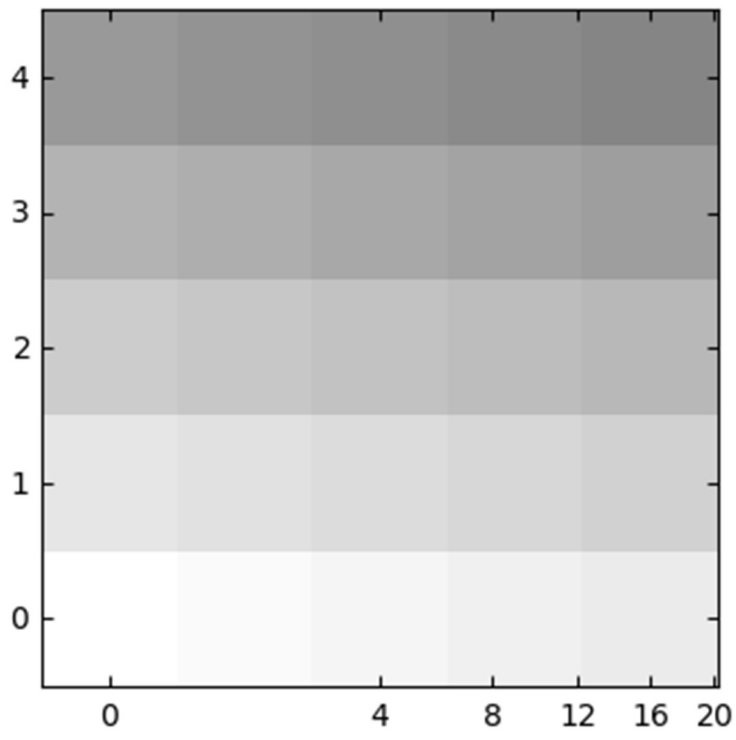
Style: he plot to have a clean and minimalistic look. Remove the top and right borders of the plot and ensure that the left and bottom axes, including their negatives, are clearly visible without arrowheads at non-zero ends. The plot line should be smooth and clearly illustrate the distribution and relationship between 'x' and 'y'. Use a color that stands out against the background for better visibility.

Generated	Ground truth
	

ID = 14
Score vis = 0
Score task = 0
Score human = 0

Task: description: Create a plot that uses a curvilinear grid helper to transform the axes. The transformation involves a custom square root scaling on the x-axis while keeping the y-axis linear. Implement an inverse transformation accordingly. Over this transformed grid, display the data from the dataframe as an image with specific styling parameters like color mapping and interpolation settings.

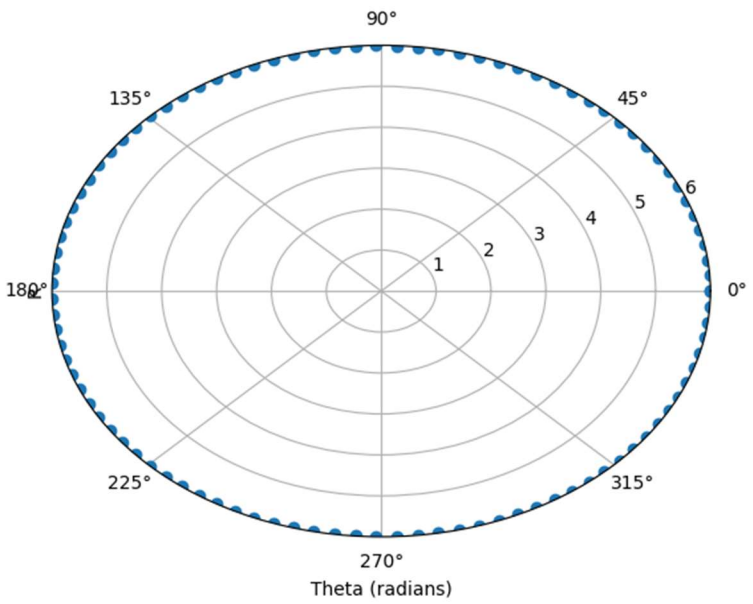
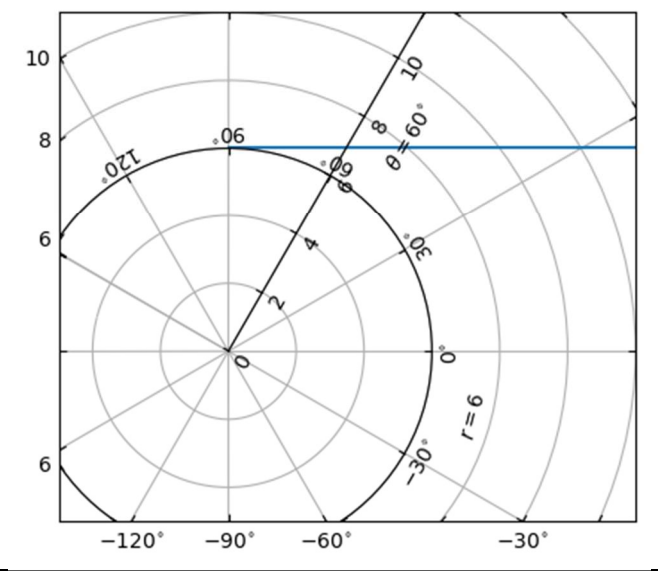
Style: description: Use a grayscale reverse color map to represent the values of the dataframe in the plot. Set specific parameters for the visual representation of the data, including the maximum value parameter for color scaling and the interpolation method for how data points are visually linked. Also, adjust the axis appearance and the origin of the plot image.

Generated	Ground truth
	

ID = 15
Score vis = 10
Score task = 90
Score human = 100

Task: velop a plot using a polar projection within a rectangular grid. The plot will map the 'Theta' values against 'R' values from the DataFrame. This will require transforming the angular data from degrees to radians and setting up a polar coordinate system in a rectangular plot area. Labels will be added to specific axes to indicate both angular and radial measurements.

Style: The style of the plot should include a grid to enhance readability, with the background consisting of concentric circles and radial lines to represent the angles. Axis labels should be formatted to indicate angles in degrees and radial distances. Display limits for the x and y axes should be explicitly set to frame the plot area properly, ensuring all data points are visible within the defined boundary of the plot. The overall aspect ratio should be adjusted to maintain proportional scaling between axes.

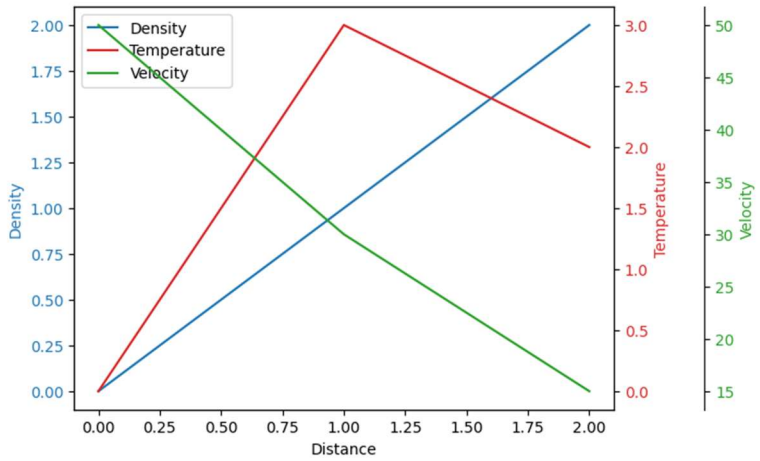
Generated	Ground truth
	

ID = 16
Score vis = 90
Score task = 95
Score human = 90

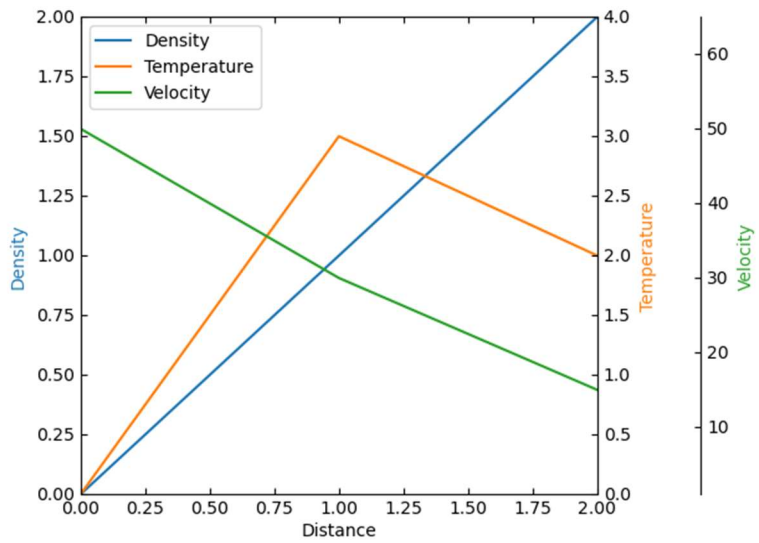
Task: Create a plot where 'Distance' will serve as the x-axis. The primary y-axis will display 'Density', and two secondary y-axes will be created; one for 'Temperature' and the other for 'Velocity'. Each variable will be represented as a separate line graph on the plot, with distinct colors to distinguish between them.

Style: The primary y-axis on the left will be colored and labelled for 'Density'. Two secondary y-axes on the right will be color-coordinated and labelled respectively for 'Temperature' and 'Velocity', with the 'Velocity' axis positioned a little more to the right. Each line representing a variable will have a unique color. Additionally, include a legend to identify each line, and customize tick mark labels and axis labels accordingly to enhance readability and visual appeal. The axis labels will also be color-coordinated to match the line they represent.

Generated



Ground truth

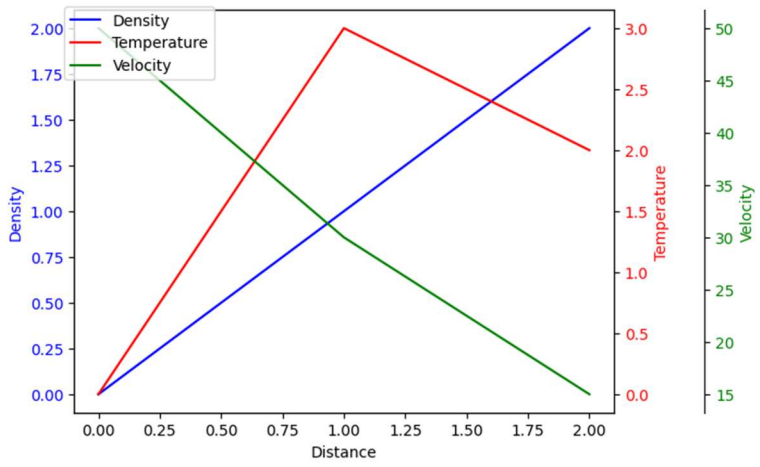


ID = 17
Score vis = 90
Score task = 95
Score human = 90

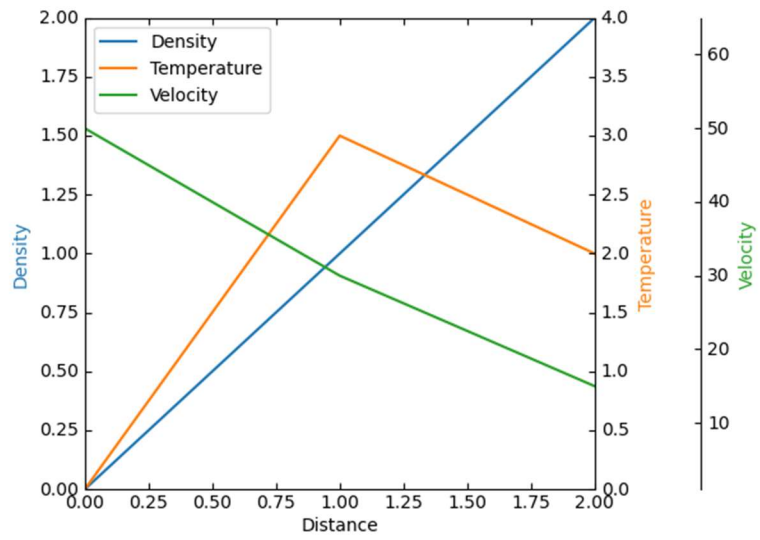
Task: Create a plot with three line series, each representing a different physical quantity (Density, Temperature, and Velocity) as a function of the Distance. The main axis (x-axis) represents the "Distance", while the main y-axis (left y-axis) represents the "Density". Add two additional y-axes: one for "Temperature" and another for "Velocity", each on the right side but with a distinct vertical display offset for the additional axis representing "Velocity". All three physical quantities should be distinguishably plotted with unique colors.

Style: Customize each axis to match the color of its corresponding data series line. Label each axis appropriately to indicate which physical quantity it represents. Set distinct limits for each y-axis based on the range of data values for Temperature and Velocity. Include a legend that explains which color corresponds to which physical quantity. Ensure the plot is clear and easy to understand, utilizing axis labels and legends effectively.

Generated



Ground truth

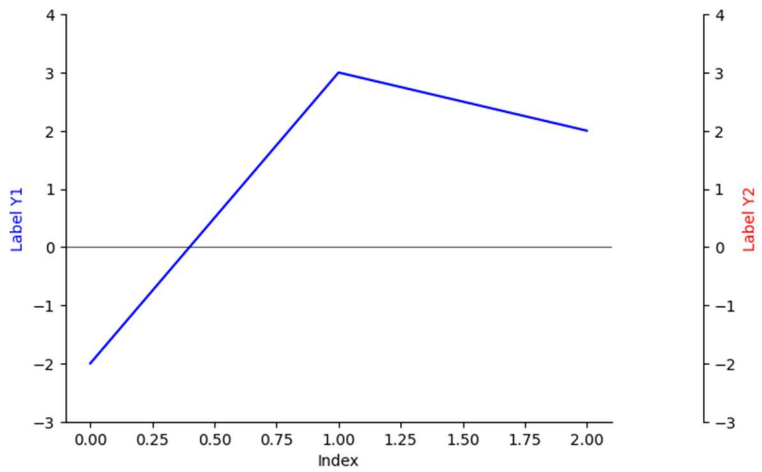


ID = 18
Score vis = 90
Score task = 90
Score human = 80

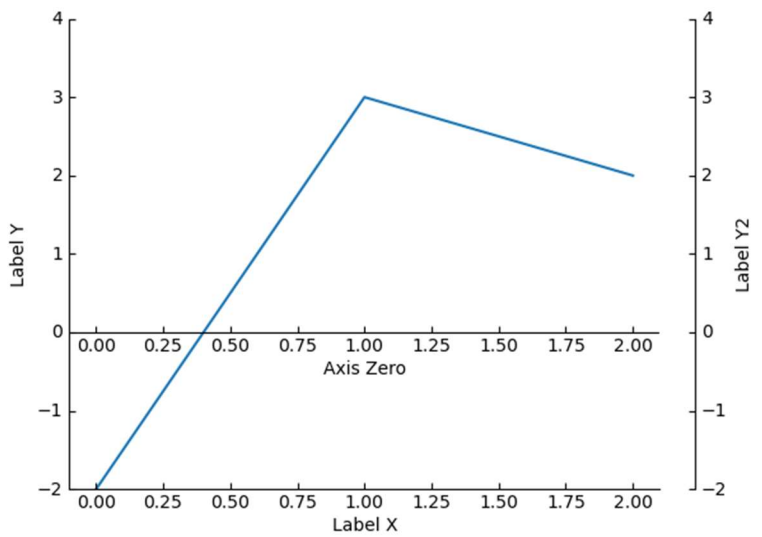
Task: The task requires creating a line plot using the values from the DataFrame. The x-axis should use index values from the DataFrame, and the y-axis should plot corresponding 'Y' values. Enhance the plot with customized axes by adding an additional y-axis on the right side that is offset from the primary y-axis. Remove visibility of the top and right borders of the plot to keep it clean. Additionally, ensure the primary x-axis (xzero axis) is highlighted to pass through y=0, indicating it as a significant reference line.

Style: The plot should prominently feature labels for both Y axes and the X axis. The right Y axis (labeled as "Label Y2") should be visibly offset from the primary frame to distinguish its measurements. Configure the visibility and labels of axes to ensure clarity in data representation. The plot should also include custom limits for the y-axis to adequately encase the data range and improve readability.

Generated



Ground truth

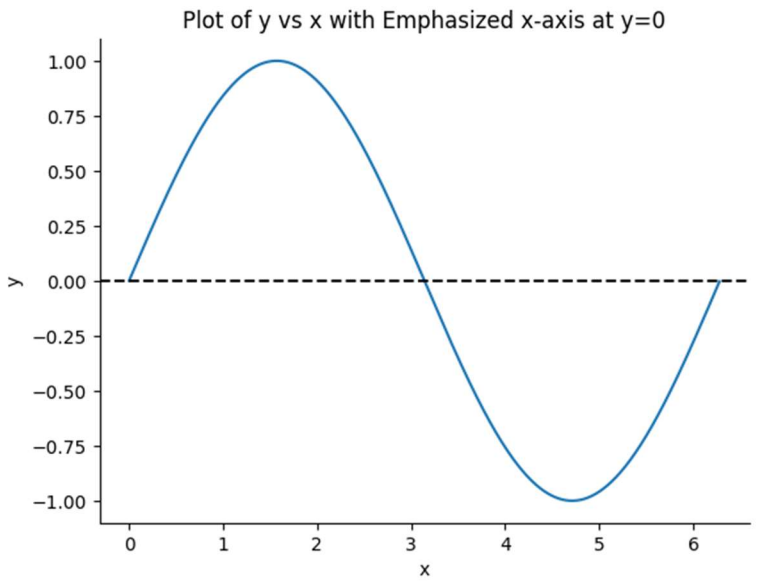


ID = 19
Score vis = 90
Score task = 90
Score human = 75

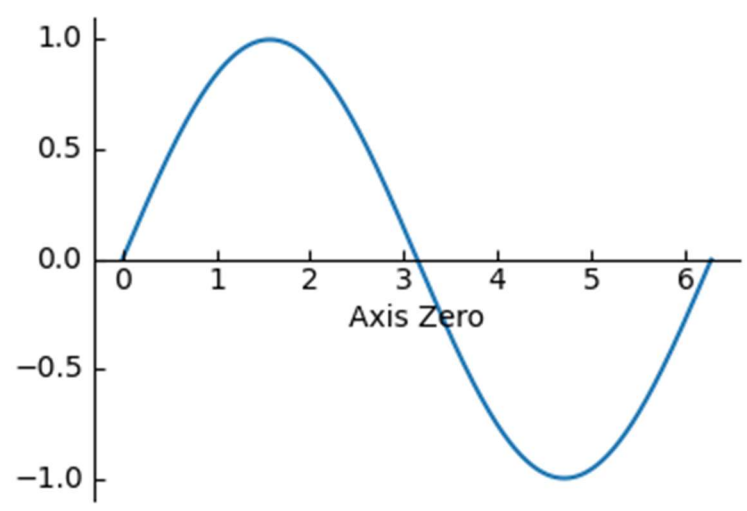
Task: The task involves creating a plot where the horizontal axis will pass through zero on the vertical scale, emphasizing the significance of the origin in the plot's context. The horizontal line (x-axis) at y=0 should be highlighted or labeled clearly. The plot should depict the relationship between the 'x' and 'y' columns of the dataframe, showing the trend or pattern described by these variables.

Style: The visual style of the plot will be minimalistic, with major focus lines such as the x-axis (at y=0) being clearly visible and emphasized. Other axes, such as the top and right borders of the plot, should be hidden to direct focus solely on the plotted data and the special x-axis. The x-axis should be labeled at a suitable position, enhancing the plot's readability and interpretative ease. The graph should effectively communicate any periodic or significant patterns present in the data.

Generated



Ground truth

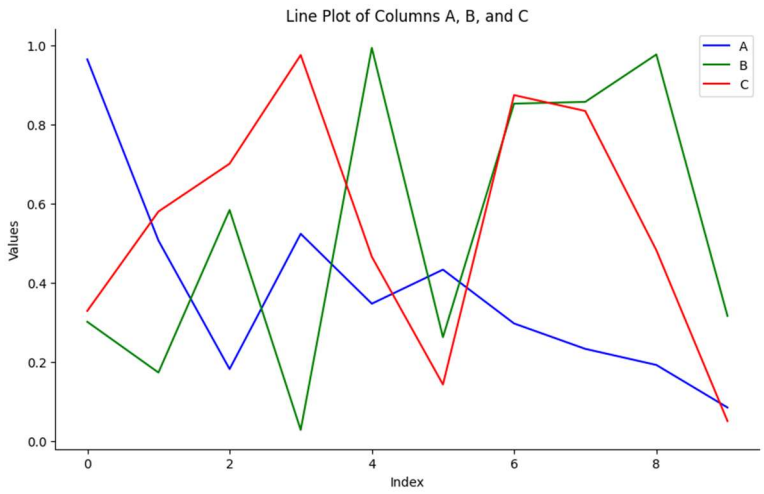


ID = 20
Score vis = 95
Score task = 90
Score human = 100

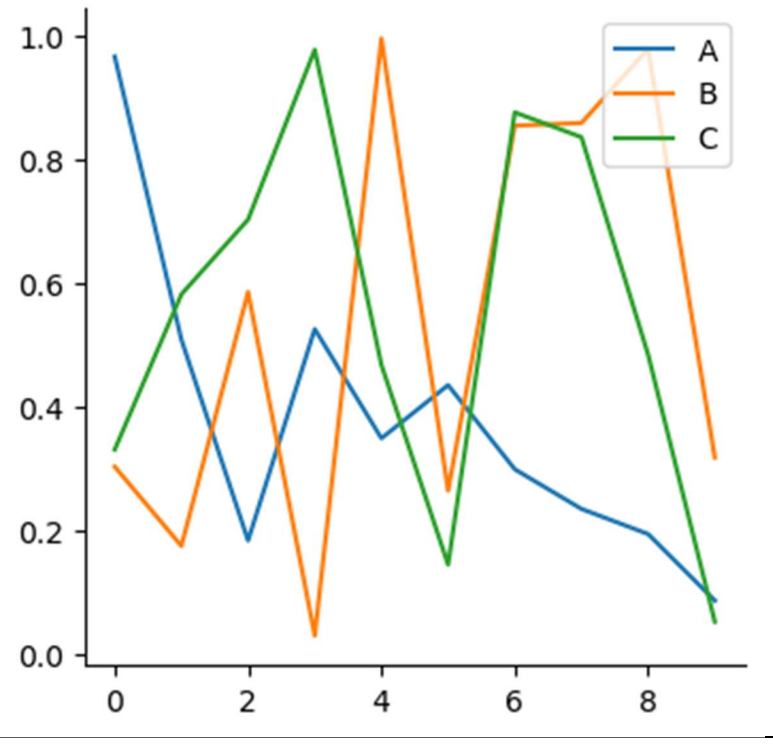
Task: The task involves generating a line plot that visually represents the trends over rows for each of the columns A, B, and C in the dataframe. This graph will display the value changes across each record for all three columns, where the x-axis represents the row index, and the y-axis represents the values of the columns.

Style: The style of the plot should ensure readability and clarity in visual differentiation between the lines representing different columns. Customize the plot by removing the top and right spines of the plot area to reduce visual clutter, enhancing the overall aesthetic and focus on the data presentation. The lines for each column should be distinctly colored to clearly differentiate between each dataset represented in the plot.

Generated



Ground truth



ID = 21
Score vis = 80
Score task = 100
Score human = 100

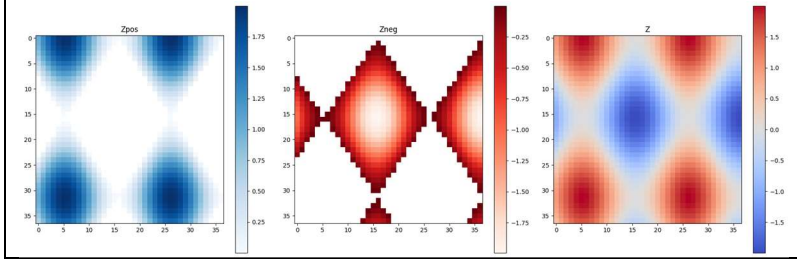
Task:

- Create a three-panel plot where each panel visually represents one of the dataframe's columns (Zpos, Zneg, Z) as 2D colored images. Each column's data must be reshaped into a square matrix format for visualization. Add a color bar for each panel to interpret the values highlighted in various colors.

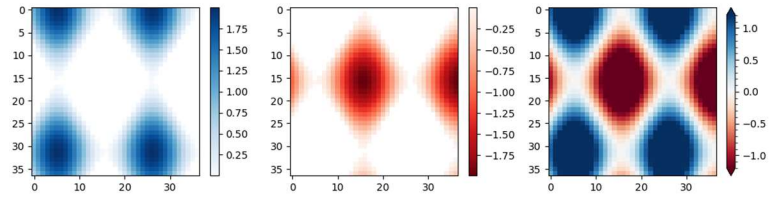
Style:

- Apply diverging color maps suitable to data characteristics for each panel (e.g., positive values in blue, negative values in red, and a range combining both for general data). Ensure no interpolation to maintain the integrity of each data point's visual representation. The final layout should clearly depict differences between positive, negative, and combined data distributions, with additional configuration for minor ticks on the color bar to enhance readability.

Generated



Ground truth

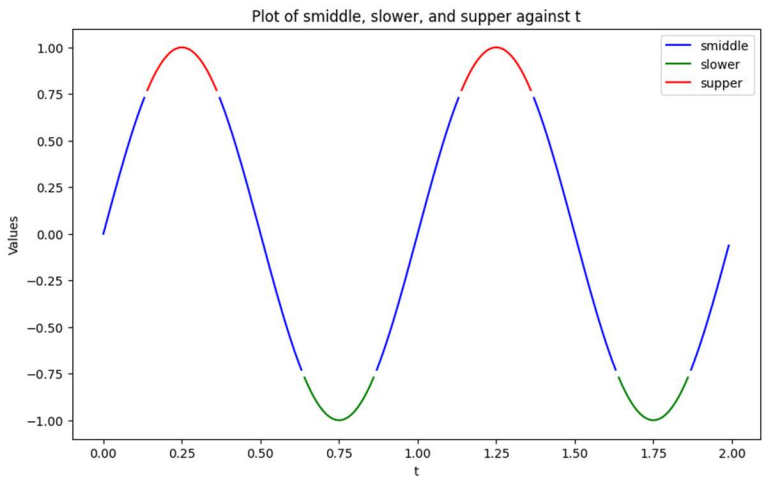


ID = 22
Score vis = 90
Score task = 95
Score human = 100

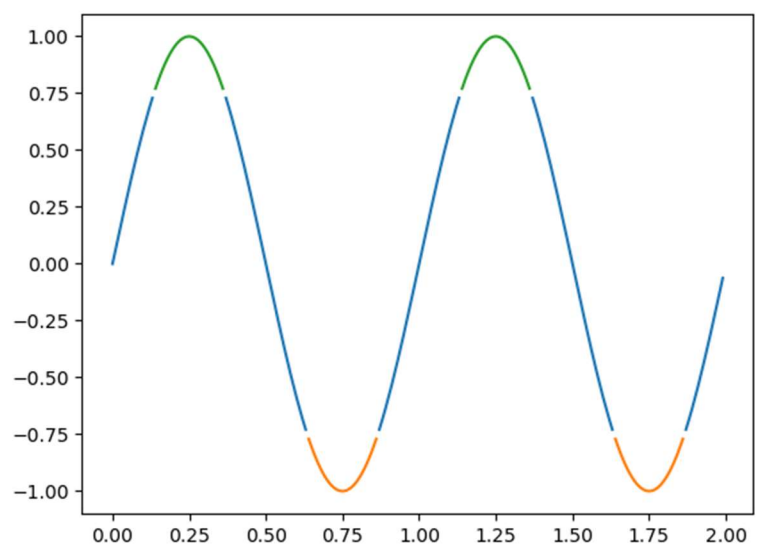
Task: hree series from the dataframe on a single graph. Each series ('smiddle', 'slower', 'supper') should be plotted against the 't' column. The three series will be represented as separate lines on the graph, showcasing how each varies over 't'.

Style: The visual representation should clearly distinguish between the three data series by using different colors for each line. The resultant plot should include appropriate axes labels and a legend, if necessary, to ensure clarity in distinguishing between the different data series. The style should be visually appealing and clear, allowing easy comparison of trends across the three series.

Generated



Ground truth

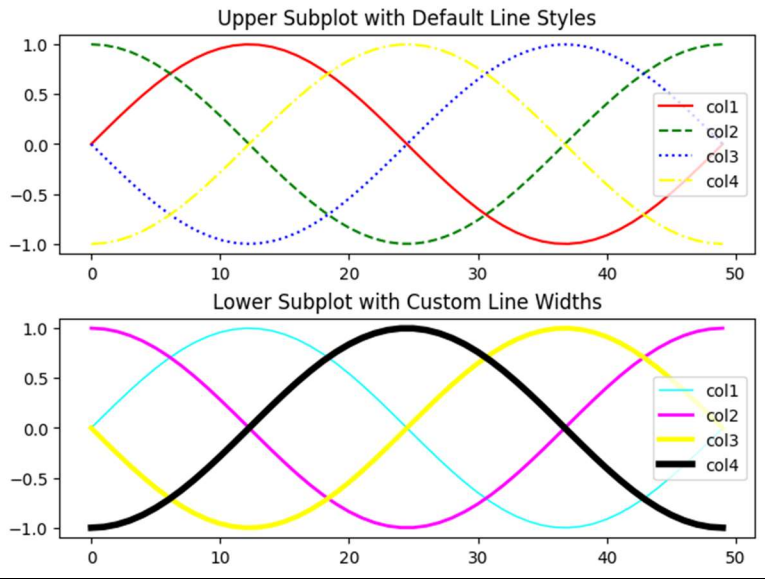


ID = 23
Score vis = 90
Score task = 95
Score human = 100

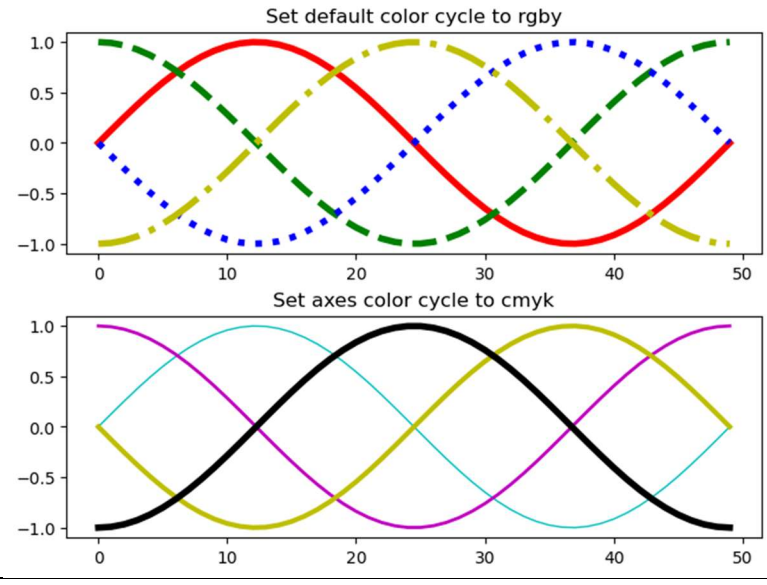
Task: Create two subplots with constrained layout. The upper subplot is configured to display a line plot of the dataframe using a predefined color and line style cycle. The lower subplot will also display a line plot of the same dataframe but with a different color and line width cycle specified for that particular axes set.

Style: The plot styles involve setting properties such as line width and color and linestyle cycles using default settings for the first subplot and using customized settings for the second subplot. Colors like red, green, blue, and yellow are used in the first subplot with solid, dashed, dotted, and dash-dot lines respectively. Meanwhile, cyan, magenta, yellow, and black colors along with increasing line widths are used for the second subplot.

Generated



Ground truth

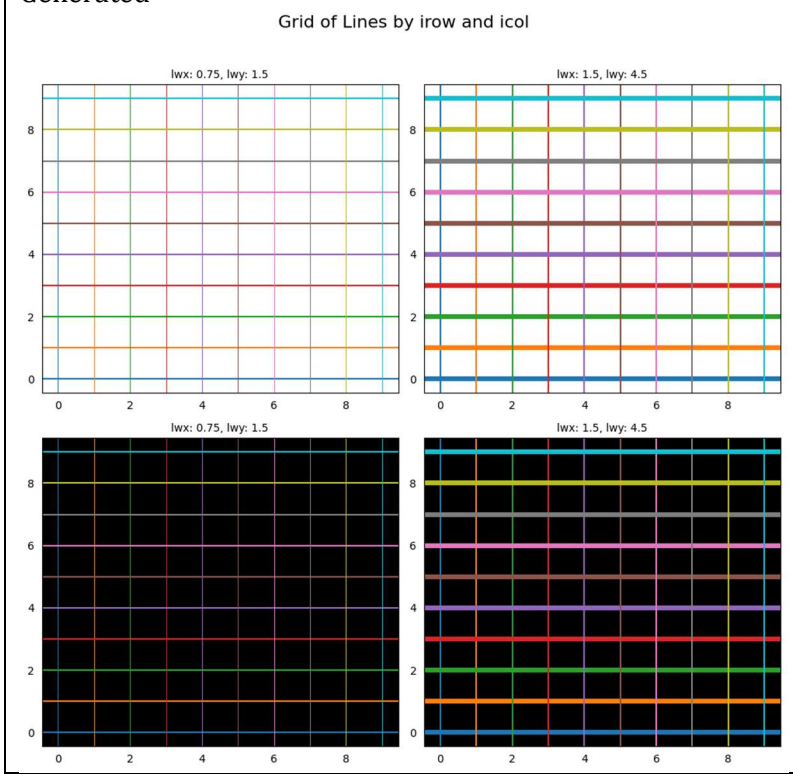


ID = 24
Score vis = 90
Score task = 95
Score human = 100

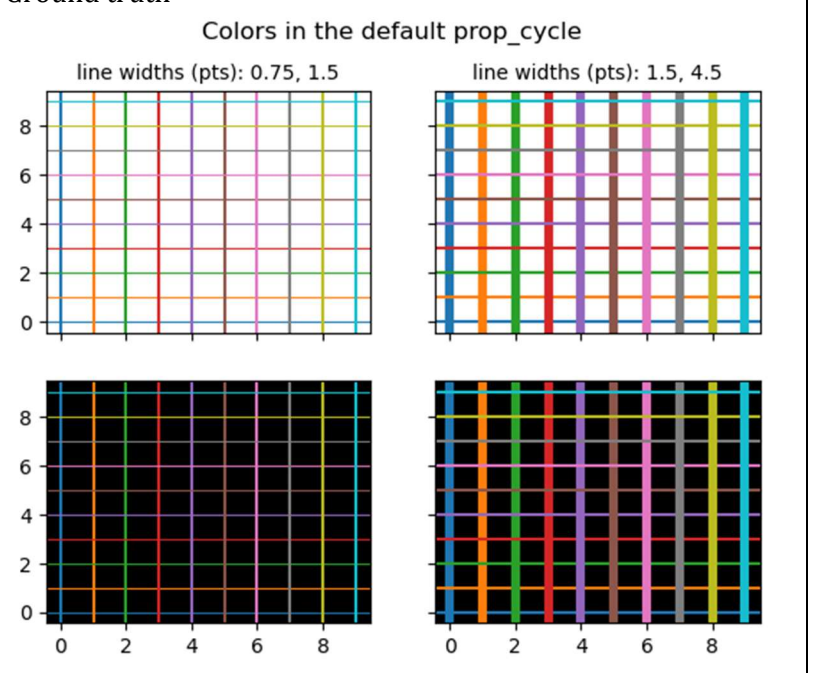
Task: Create a grid of plots (2x2) where each subplot corresponds to combinations of rows and columns indexed by 'irow' and 'icol'. Within each subplot, draw both horizontal and vertical lines at positions specified by the index 'i', with line colors and thickness denoted by 'color', 'lwx', and 'lwy'. Additionally, configure ticks, titles, and subplot background color, ensuring consistency across certain axes.

Style: Configure the subplot backgrounds in the second row to be black. Adjust tick positions on axes selectively to improve readability. The plot titles should indicate the line widths used, formatted to a readable size. The overall plot should have a title that is larger in size, summarizing the theme or category of data visualized.

Generated



Ground truth

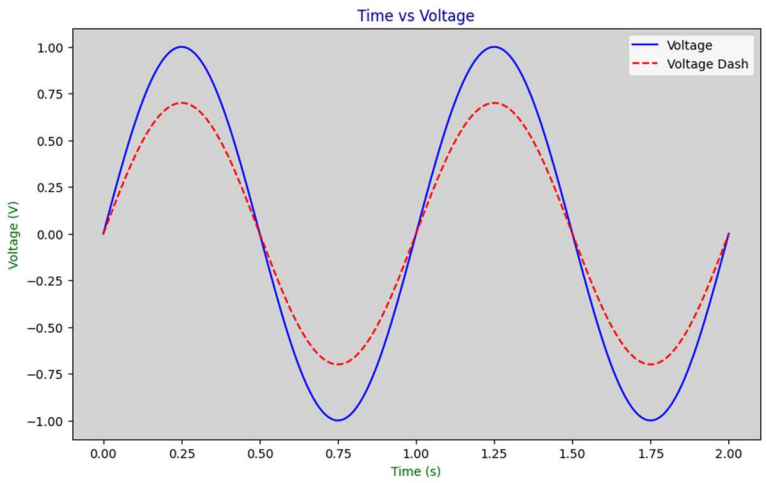


ID = 25
Score vis = 90
Score task = 95
Score human = 100

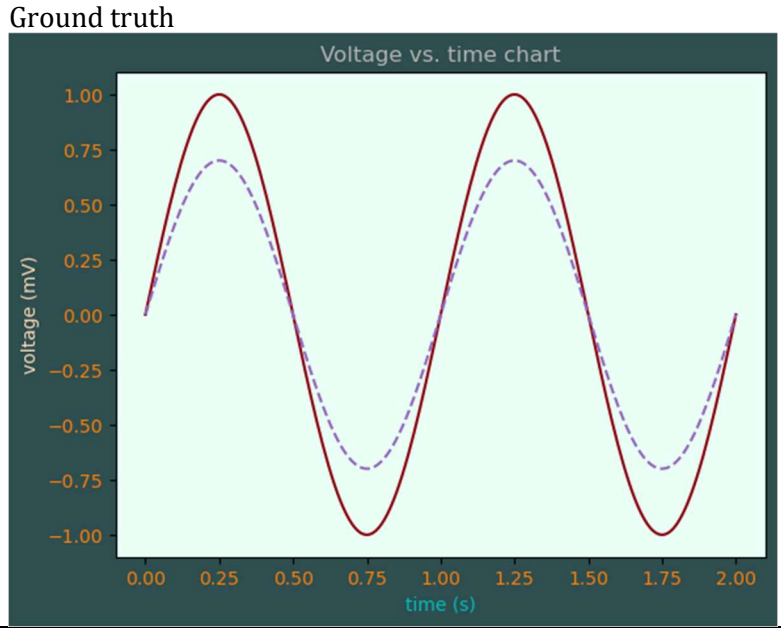
Task: Create a plot displaying the relationship between time and voltage values for both 'voltage' and 'voltage_dash' columns. Display 'voltage' using a solid line and 'voltage_dash' using a dashed line to differentiate the two series clearly. Include appropriate labels for both axes and a title for the chart to effectively communicate the plot's purpose.

Style: he plot with a distinctive background color for readability and aesthetic appeal. Set the color of the plot title and axis labels using specific tones that maintain visibility and contrast against the background. Use distinct color choices for the plot lines to separate the two data series visually. Adjust the color settings for tick labels to ensure they are clear and easily readable against the styled background.

Generated



Ground truth

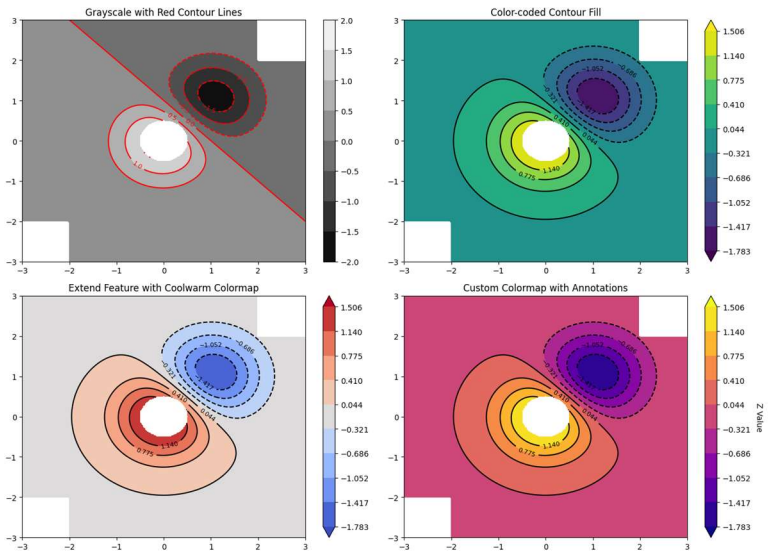


ID = 26
Score vis = 30
Score task = 100
Score human = 100

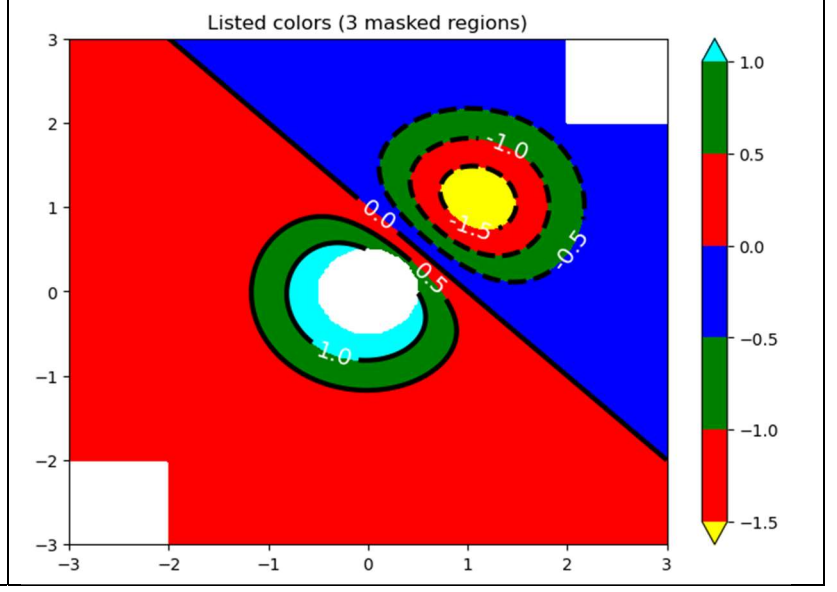
Task: The task involves generating multiple contour and contour-filled plots from the structured grid data defined by the columns X, Y, and Z. These plots should demonstrate different visualization capabilities, including varying color maps, levels of contour, handling of data extremes with color extensions, and configuration of contour lines and labels.

Style: fferent styles and configurations are needed for distinct visual effects. These include:
- Using a grayscale color map and a red contour line for the first set of plots, with masked regions indicated.
- A set of color-coded contour fill plots with explicit color assignments for ranges both within and outside the defined levels (i.e., under and over values).
- For demonstrating the 'extend' feature in contour plots, using a color map that adjusts for values below and above specified thresholds, handling color overflow and underflow visually.
- Proper annotation and color bar adjustments to enhance the interpretability of the plots, including modifying label orientation, font sizes, and adding titles for axes and the overall plots.

Generated



Ground truth

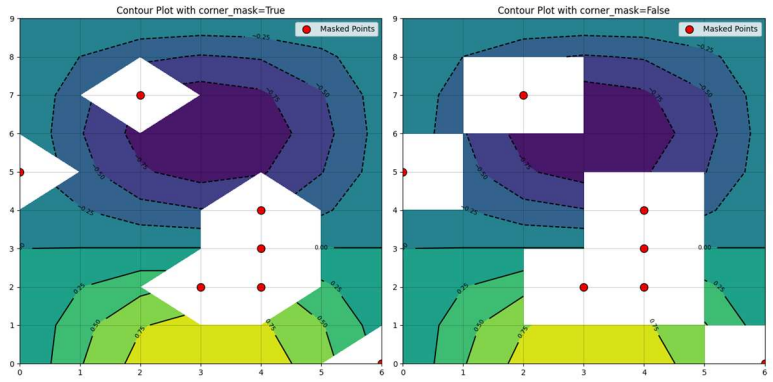


ID = 27
Score vis = 90
Score task = 95
Score human = 100

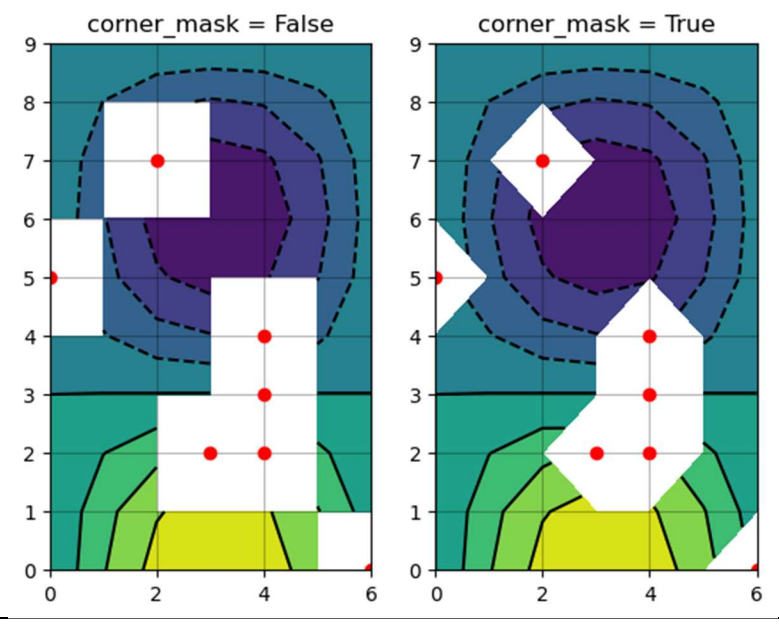
Task: Create two plots side-by-side, each displaying a contour filled plot of Z values on a grid defined by X and Y coordinates. Additionally, overlay a contour line plot with the same Z values. Highlight data points excluded by the Mask using red circles. Each subplot should handle the corner mask parameter differently.

Style: Use black grid lines with slight transparency for better visibility of contours. The title of each subplot should indicate the state of the corner mask parameter. The contour fill should vary in color according to the Z value, with each plot showing contour lines in black for clarity and contrast.

Generated



Ground truth

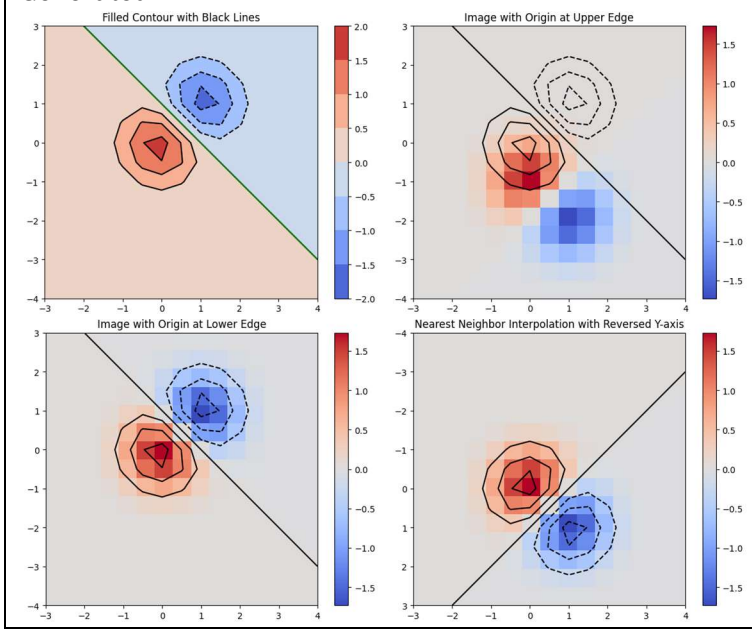


ID = 28
Score vis = 80
Score task = 95
Score human = 100

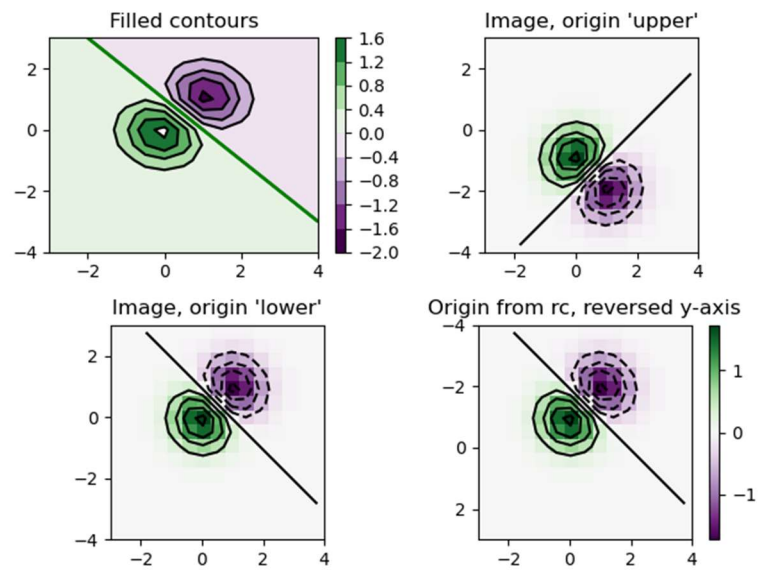
Task: Create a series of plots in a 2x2 grid layout:
- The first plot should display filled contour lines based on the Z values with color differentiation and contour lines highlighted in black. Add a green contour line at zero for prominence.
- The second and third plots should show images derived from the Z values, one with the origin at the upper edge and one at the lower, both overlaid with black contour lines.
- The fourth plot is an image with nearest neighbor interpolation and the y-axis reversed, also overlaid with black contour lines.

Style: Adjust aesthetics and functionality across all plots:
- Employ a dual-ended colormap that highlights variation in Z values symmetrically around zero.
- Use color bars to indicate the value ranges in at least two of the plots.
- Set contour lines and images within specified axis extents to maintain uniform spatial representation.
- Apply clear titles to each subplot for easy identification and adjust plot layouts for optimized spacing and clarity.

Generated



Ground truth

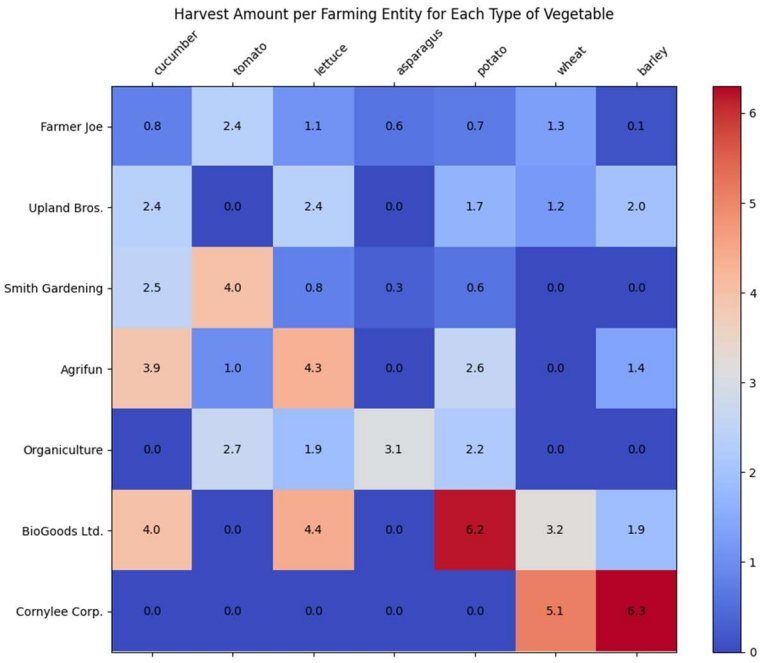


ID = 29
Score vis = 95
Score task = 95
Score human = 100

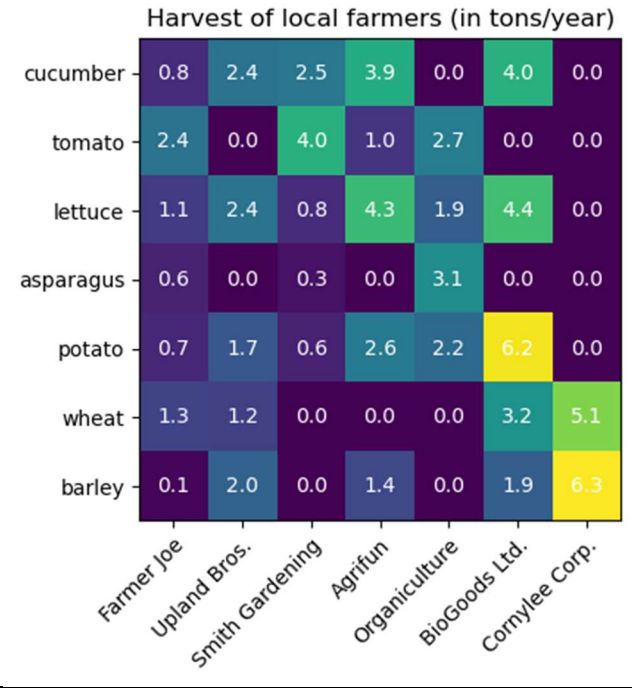
Task: Create a heatmap displaying the harvest amount per farming entity for each type of vegetable. Display axes ticks for readability, customized with labels from the dataframe (vegetables and farmer names). Superimpose numerical harvest values over the heatmap for clarity and add annotation for understanding individual entries.

Style: Configure the tick labels on the x-axis to make them more readable by rotating them. Enhance visual clarity and aesthetics by selecting appropriate colors for the heatmap where lower values might correspond to cooler colors and higher values to warmer colors. Emphasize text within the plot to ensure visibility against varying background colors. Ensure the plot is neatly laid out with an appropriate title and tight layout settings to optimize space usage and presentation.

Generated



Ground truth



ID = 30
Score vis = 70
Score task = 100
Score human = 100

Task:

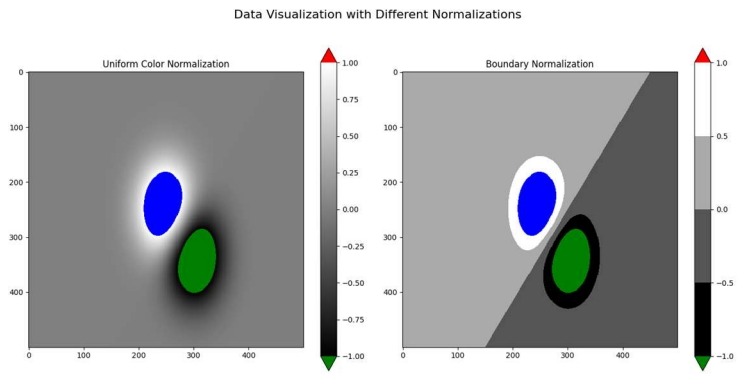
Generate two subplots using data from the DataFrame. Each subplot should display the data using an image visualization method where data values are represented as colored pixels. Mask certain conditions (e.g., values greater than a certain threshold). The first subplot should display the data with a uniform color normalization within the range specified, while the second subplot should utilize boundary normalization with specified thresholds to map colors.

Style:

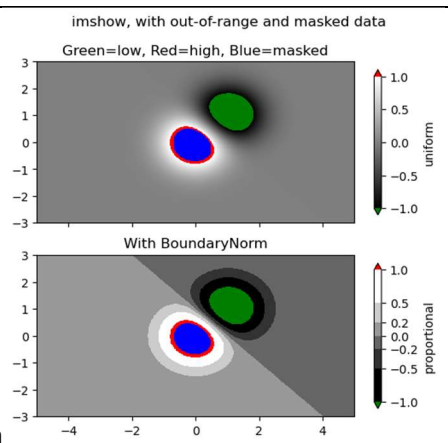
Customize the plot with specific color maps and norms:

- In the first subplot, use a grayscale color map with modifications to show segments of data that are beyond the normal range or masked, by using specific colors (such as red for high values and green for low values).
- In the second subplot, enhance the visualization by employing a boundary normalization method that uses a grayscale color map, again distinguishing out-of-range data with specified colors.
- Both subplots should have annotations and addition of colorbars to indicate the scale and extend the display of out-of-range values.
- Set the overall layout to ensure clarity and visual appeal, with titles for each subplot to explain the color encoding logic. Include an overarching title for the figure.

Generated



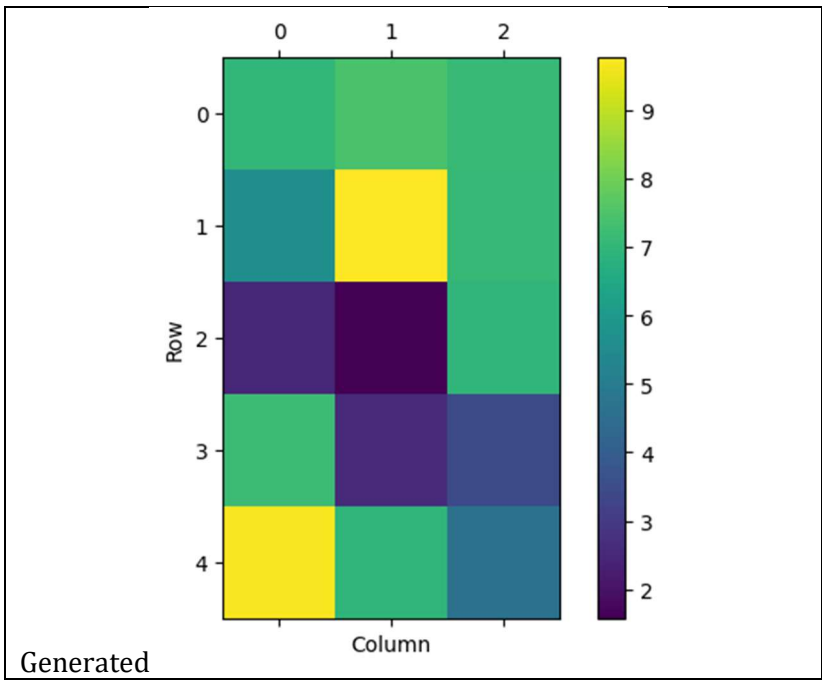
Ground truth



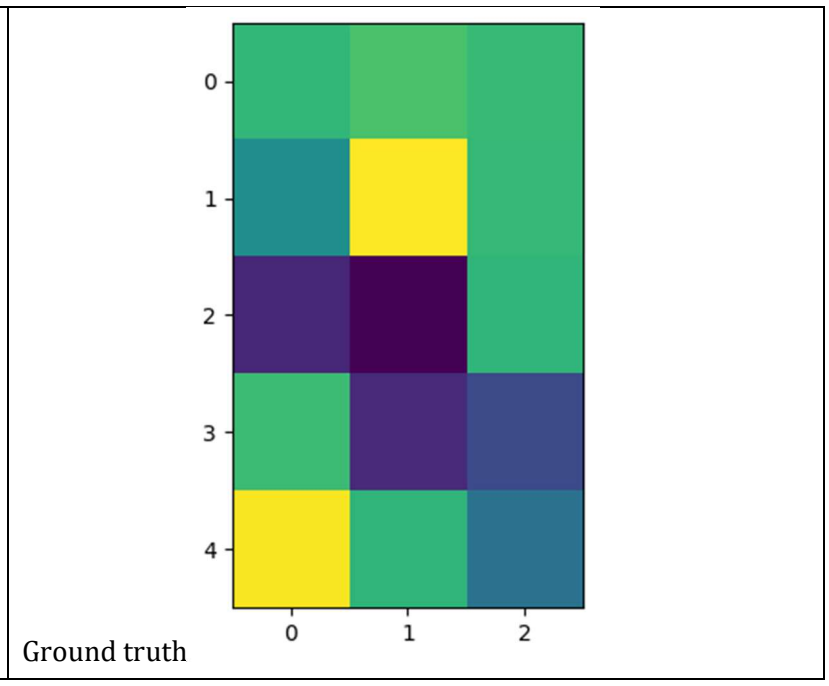
ID = 31
Score vis = 90
Score task = 60
Score human = 100

Task: ay the dataframe as a colored grid where each cell represents an element of the dataframe. The color of each cell should reflect the magnitude of the value relative to the other cells, with a legend or scale possibly indicating value ranges. Provide interactive mouse-over feature to display the exact coordinates of the mouse pointer and the value of the cell at those coordinates on the plot.

Style: The plot should use distinct colors to differentiate between the values, enhancing readability and visual appeal. The simplicity in the color scheme will aid in immediately distinguishing between high and low values. The axes should be clearly labelled to indicate row and column numbers, enhancing understandability of the plot layout. Incorporate a method for displaying value and coordinates dynamically when hovering over different parts of the plot, facilitating deeper analysis through interactivity.



Generated



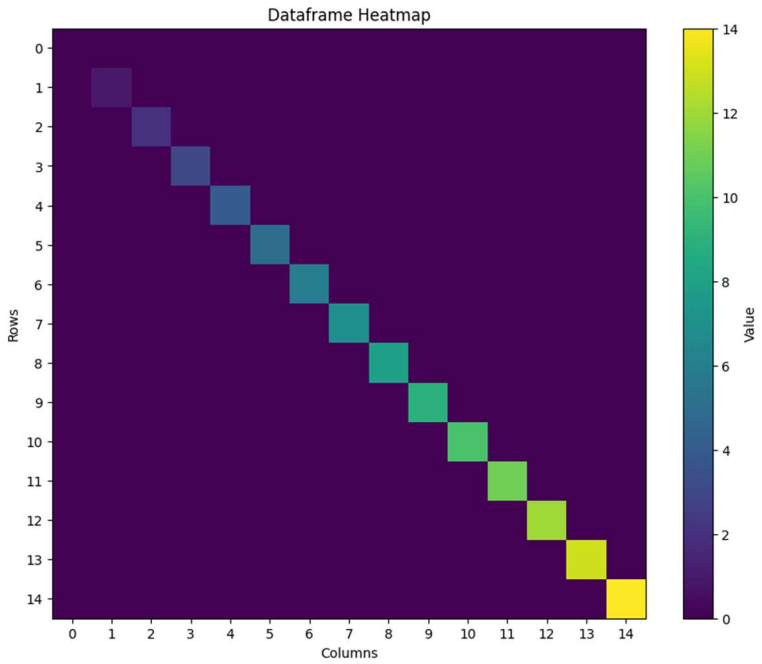
Ground truth

ID = 32
Score vis = 95
Score task = 90
Score human = 100

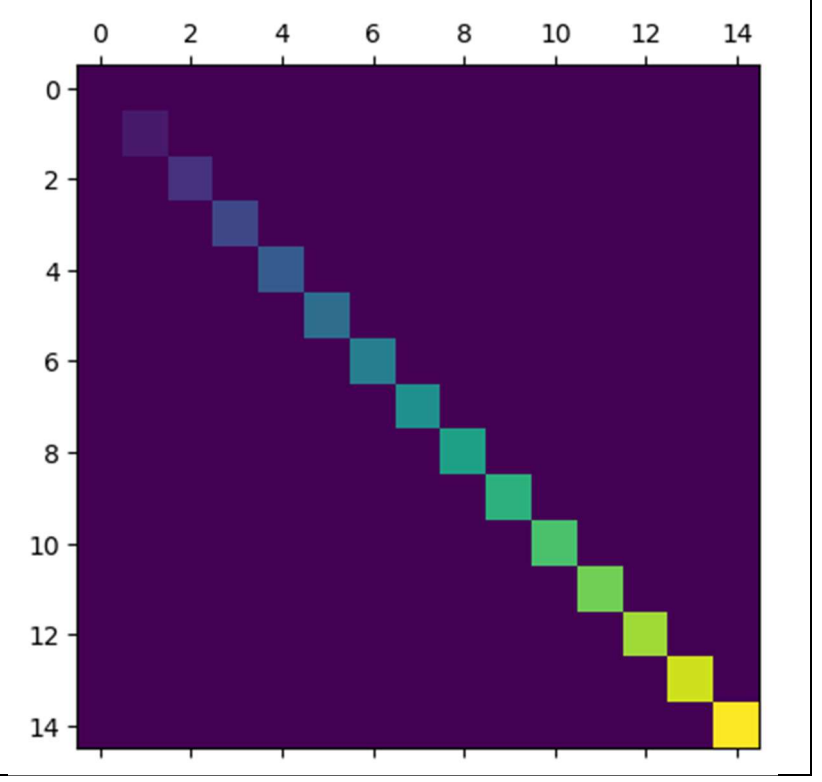
Task: The task involves visualizing the dataframe as a colored matrix where each cell's color intensity relates to its value. Each row and column of the matrix must be labeled clearly to understand the scale of the data. The main aim is to visually represent the variability and distribution of the dataframe's numerical values through hues, providing an immediate visual summary of the data characteristics.

Style: The visualization will use a color scale that changes with the values to make different data magnitudes easily distinguishable. Axis labels should be apparent, showcasing row and column numbers corresponding to the dataframe's structure. The plot should have a cohesive color theme that aids in pattern recognition and highlights progression across the dataframe.

Generated



Ground truth

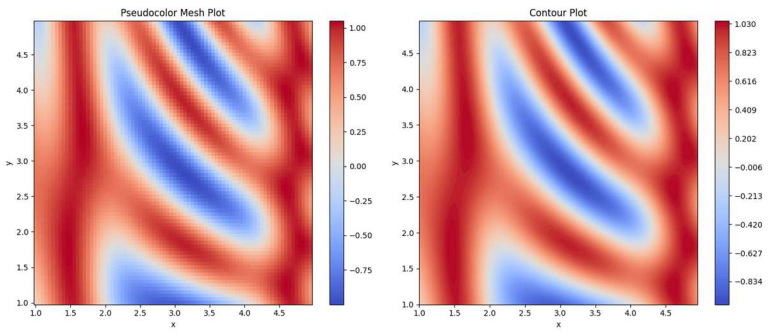


ID = 33
Score vis = 80
Score task = 95
Score human = 75

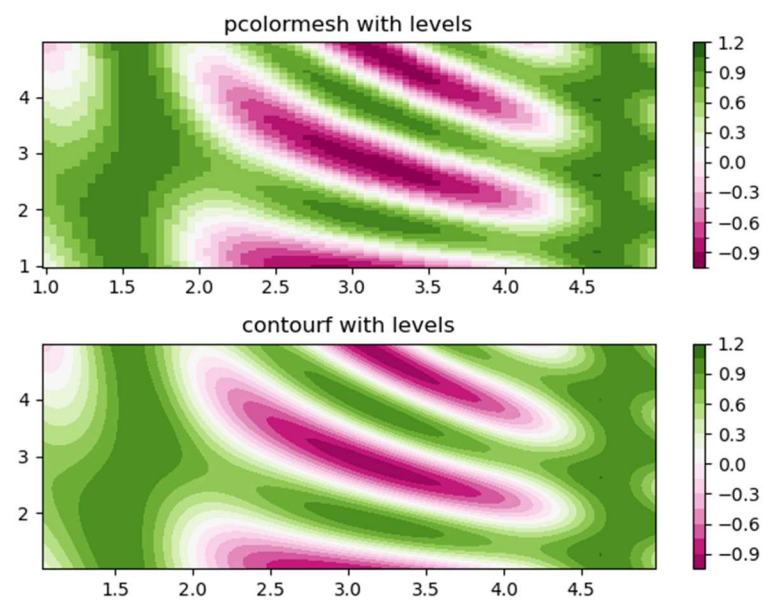
Task: description: Create two types of plots, placed as subplots in a single figure. The first plot should be a pseudocolor mesh plot that visualizes the 'z' values over 'x' and 'y' coordinates. The second plot should depict contour levels built from the 'x', 'y', and 'z' values. These plots should utilize a normalization and coloring scheme based on the provided levels determined from the 'z' data's range.

Style: description: Utilize a diverging color map to differentiate values clearly in both subplots. Define a fixed number of levels to represent the data range effectively through color gradations. Ensure axis titles, color bars for scale reference, and appropriate plot titles are included to enhance readability. Adjust subplot spacing to prevent label overlap, maintaining a clean and organized visual presentation.

Generated



Ground truth

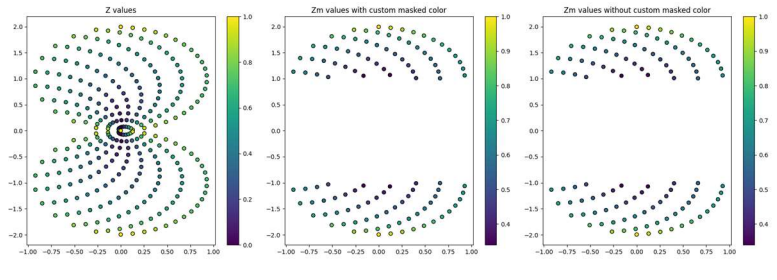


ID = 34
Score vis = 10
Score task = 95
Score human = 90

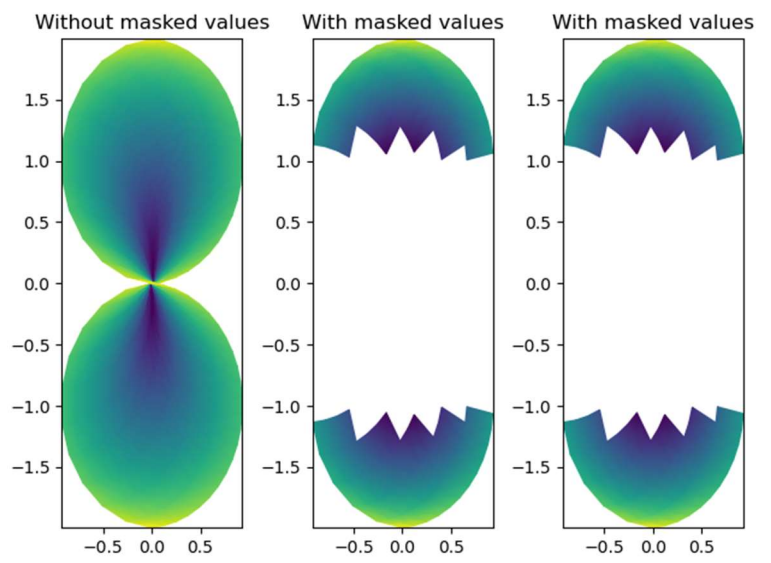
Task: Create a single figure containing three subplots arranged horizontally. Each subplot should display a two-dimensional grid plot of the data:
- The first plot should visualize the 'Z' values against 'Qx' and 'Qz' grids, showing data distribution without masking any values.
- The second plot should also show 'Qx' and 'Qz' grids but plot 'Zm' values where missing data are visualized distinctly using a custom color map.
- The third plot should repeat the second plot layout but without adjusting the color map, directly handling missing data as per default settings. Each subplot should be clearly titled to denote whether they include masked values or not.

Style: Use a smooth shading interpolation for all three plots to enhance visual clarity. The first two plots should have specific styling adjustments: the second plot necessitates a modified color map where masked (missing) values receive a specific color. Ensure the layout is tight to prevent overlap of subplot contents for neat representation. The color palette should be consistent across plots where masked values are treated distinctly for enhanced data interpretation.

Generated



Ground truth

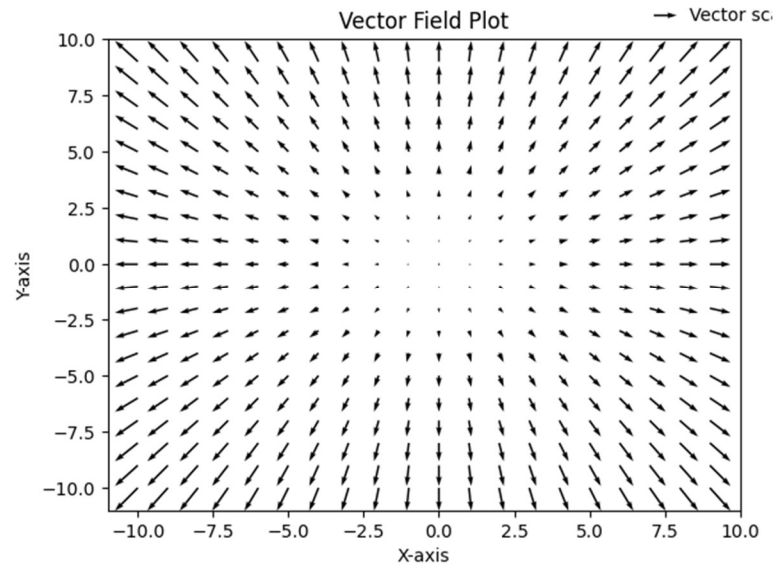


ID = 35
Score vis = 95
Score task = 95
Score human = 100

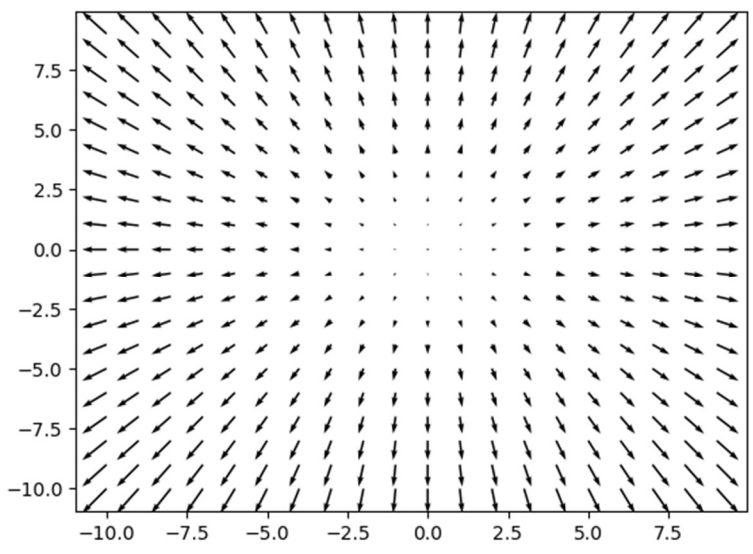
Task: Create a vector field plot using the data from the DataFrame. The plot should graphically represent vectors with components (U, V) at points (X, Y) on a two-dimensional grid. Include a key in the plot to indicate the scaling of the vectors, with specifications on its position relative to the plot area.

Style: The plot should be stylized to ensure legibility and aesthetic appeal. This includes setting an appropriate scale and limits for the axes to encompass all data points neatly. Aesthetic enhancements like labeling the axes and including a descriptive title or legend for the vector scale are also essential. Adjust the plot's general appearance such as colors or line styles to make important features stand out and ensure the plot communicates the data clearly and effectively.

Generated



Ground truth

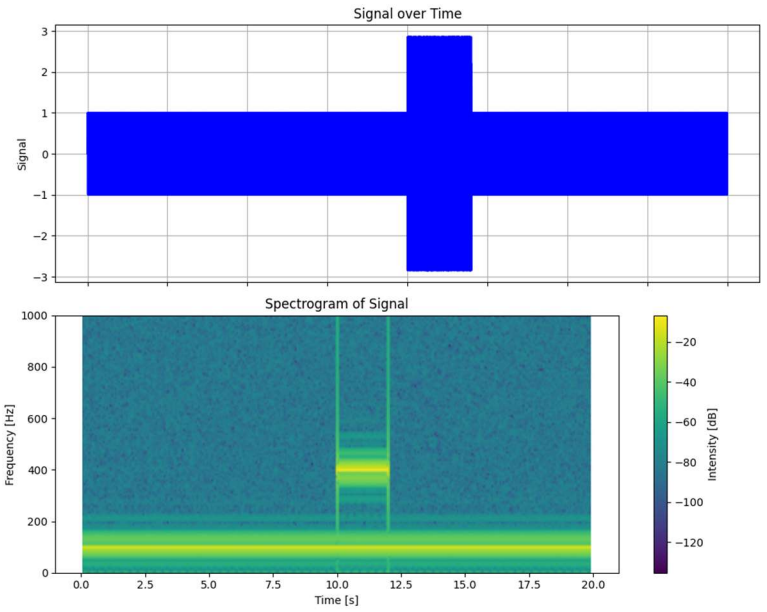


ID = 36
Score vis = 95
Score task = 90
Score human = 100

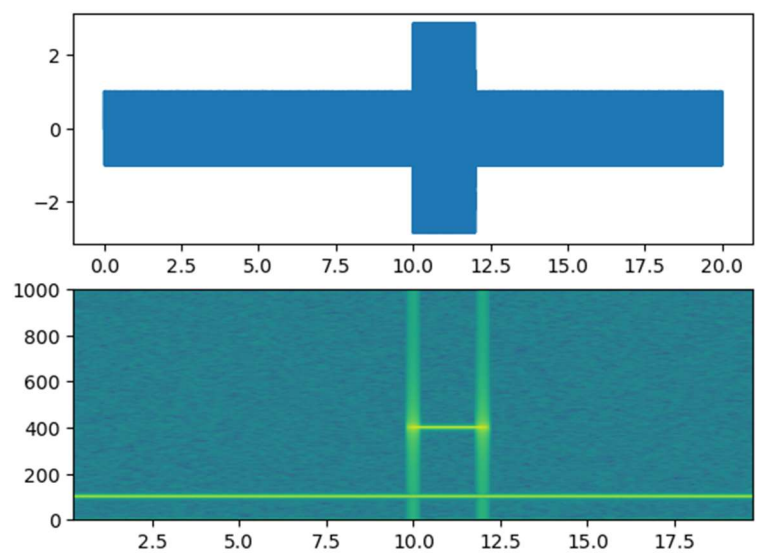
Task: The plot should consist of two subplots arranged vertically. The top plot should be a simple line plot illustrating how the 'Signal' varies over 'Time'. The bottom plot should be a spectrogram representing the frequency content of the 'Signal' over 'Time'. The spectrogram plot should use a specific frequency analysis window and overlap parameter.

Style: The top plot should have clear, labelled axes indicating the units of 'Time' and 'Signal'. It should display the full range of the data. The bottom spectrogram plot should include axes indicating time and frequency, with frequency on the vertical axis extending to the adequate range based on the sampling rate used in spectrogram computation. Both plots should share the same horizontal 'Time' axis for easy comparison. Colormaps, axis labels, and layout configurations should enhance clarity and understanding of data representation.

Generated



Ground truth



ID = 37
Score vis = 20
Score task = 90
Score human = 75

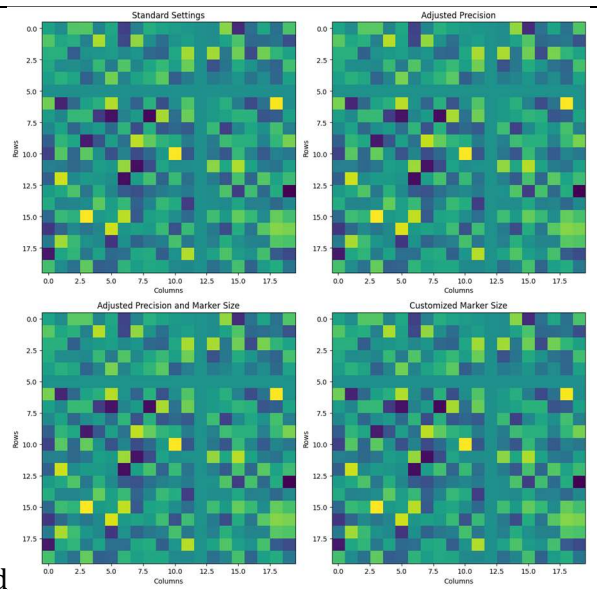
Task:

Create a grid of four separate plots (2x2 layout), where each plot will visualize the matrix from the DataFrame. The visualization should reflect the presence or absence of significant values in the matrix, possibly using differing precision and depiction settings across plots to highlight different aspects. Each matrix plot should be customizable in terms of how the data entries are visualized (e.g., adjusting precision or marker size) to allow a nuanced examination of the numerical data.

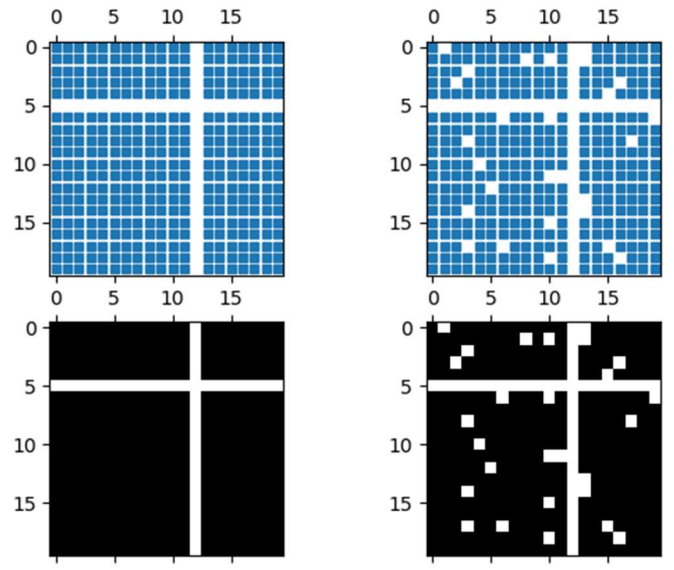
Style:

Apply specific style adjustments to each subplot for clear differentiation. One plot should use standard settings, another with adjusted precision, one with both adjusted precision and marker size, and one more with customized marker size. These variations will aid in better understanding the spread and concentration of significant values in the DataFrame's matrix by altering the granularity and scaling of visual indicators. Adjust axes and tick settings as necessary for optimal clarity and presentation aesthetics. Choose style settings such as color or marker types to enhance legibility and analytical utility.

Generated



Ground truth

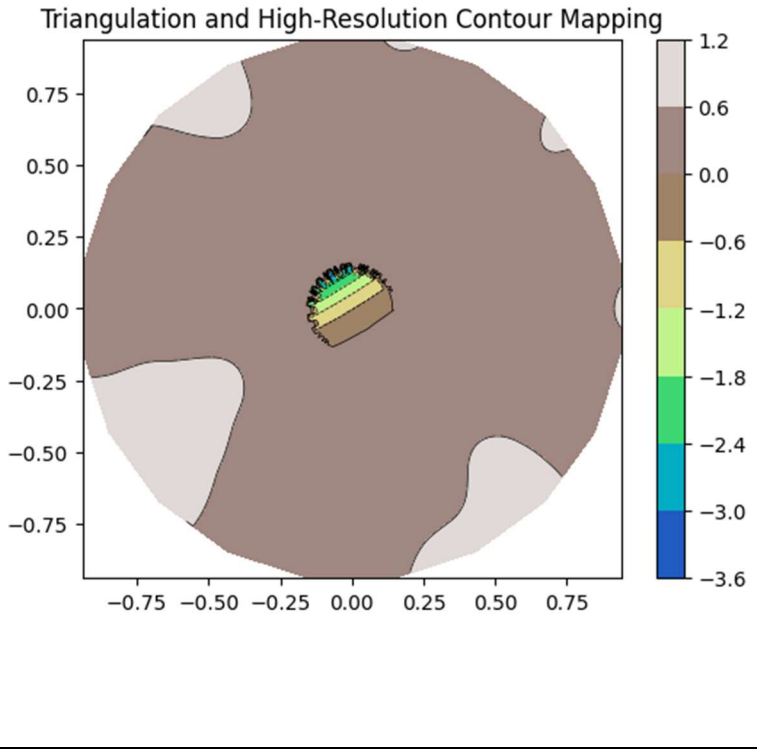


ID = 38
Score vis = 30
Score task = 85
Score human = 50

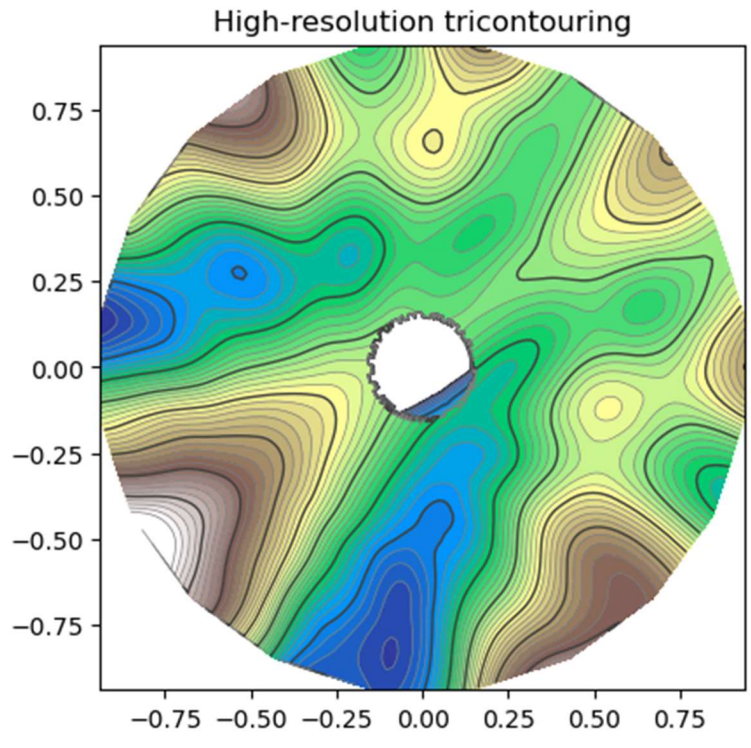
Task: The goal is to create a plot exhibiting a triangulation and high-resolution contour mapping of scalar data. Triangulation should be based on the 'x' and 'y' coordinates, and contouring reflects the 'z' values to visualize data topology with both contour lines and color-filled contour levels. Include the plot title and equal aspect ratio to maintain spatial proportions.

Style: Utilize a terrain color map to fill contours and depict elevation or density with varying hues. Contours should be visually distinguished by varying widths and greyscale colors to indicate different thresholds. The plot should prioritize clarity and aesthetic appeal, integrating fine lines for the triangulated grid and smooth, vibrant color transitions for the contour fills. The background or base of the plot should be styled minimally to ensure focal attention on the contour features.

Generated



Ground truth



ID = 39
Score vis = 100
Score task = 100
Score human = 100

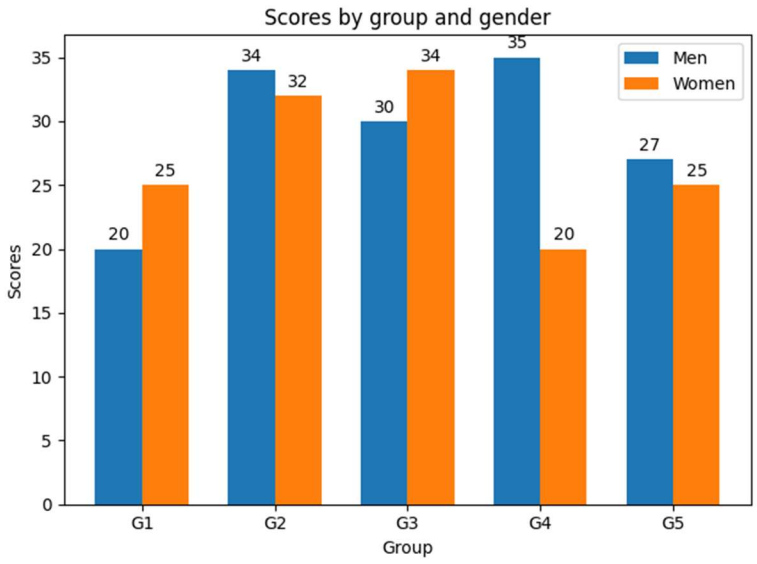
Task:

- Produce a grouped bar chart which shows scores for men and women across different groups.
- Each group identified by 'labels' should have two adjacent bars showing scores for men and women respectively.
- Include annotations displaying the score value on top of each bar.
- Add a legend to differentiate between men's and women's scores.

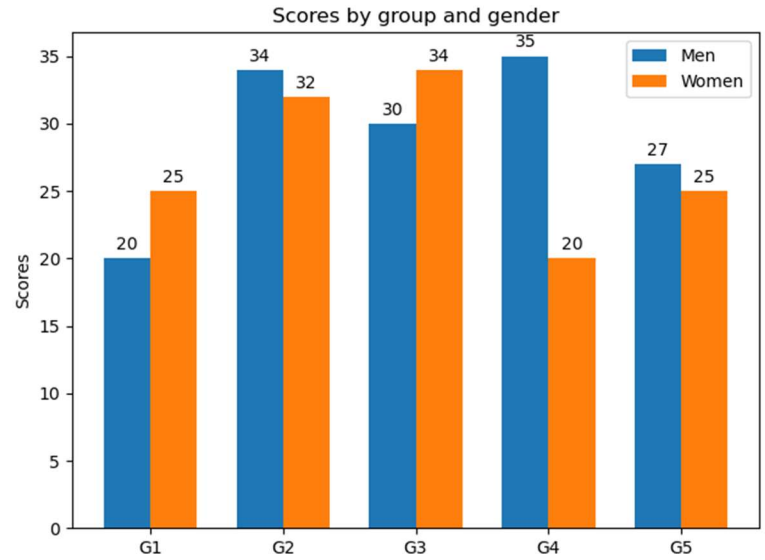
Style:

- Set the y-axis label to "Scores".
- Set the plot title to "Scores by group and gender".
- For each group on the x-axis, label the ticks according to the 'labels' column.
- Format the plot to ensure all elements are displayed correctly and without overlap.

Generated



Ground truth

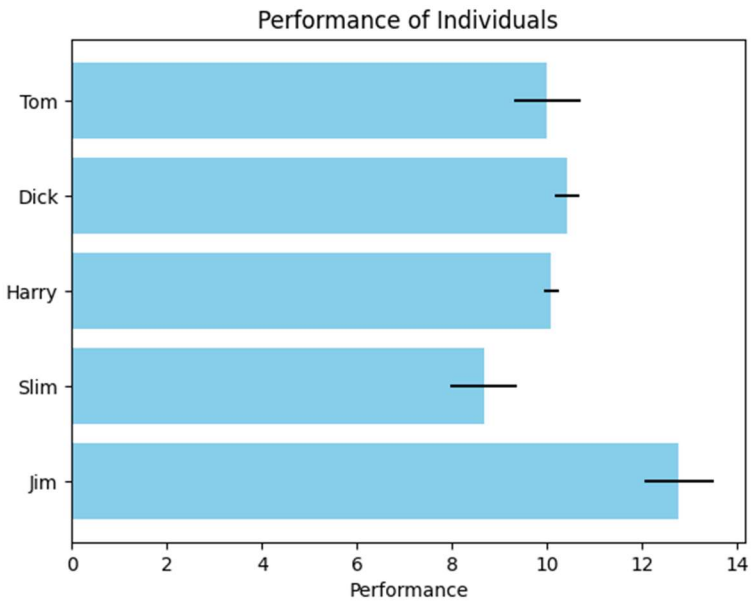


ID = 40
Score vis = 90
Score task = 95
Score human = 100

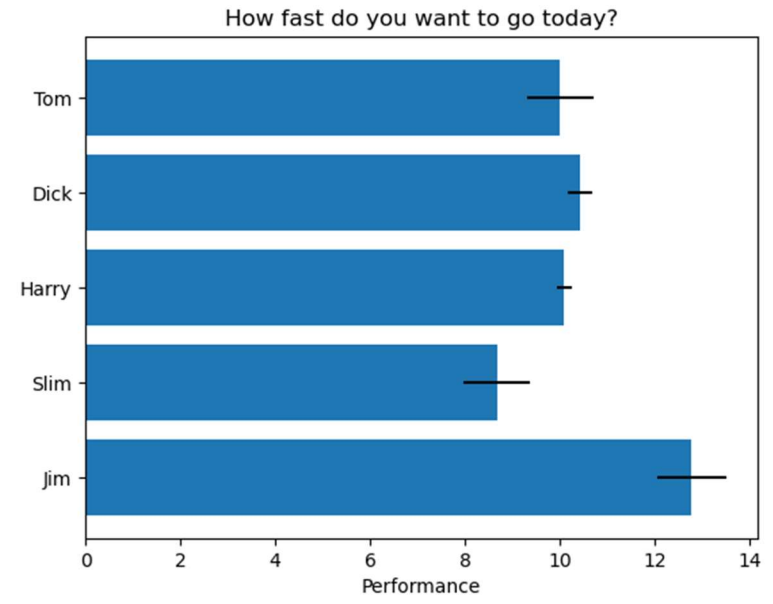
Task: The plot will be a horizontal bar chart displaying individuals' performance on the vertical axis with their respective performance values on the horizontal axis. Error bars will be added to each bar to represent the variability in the performance measurements. The vertical axis will display the names of individuals and should be inverted so that the labels read top-to-bottom. The horizontal axis will be labeled as 'Performance'. A title will be added at the top of the plot.

Style: The chart will use a simple, clear style with bars in a solid color. The plot's layout will be straightforward with clearly labeled axes and bars, along with inclusively displayed error bars for each individual's performance data. The plot will emphasize readability and direct representation of data values, ensuring that the visual presentation matches the data's intent.

Generated



Ground truth

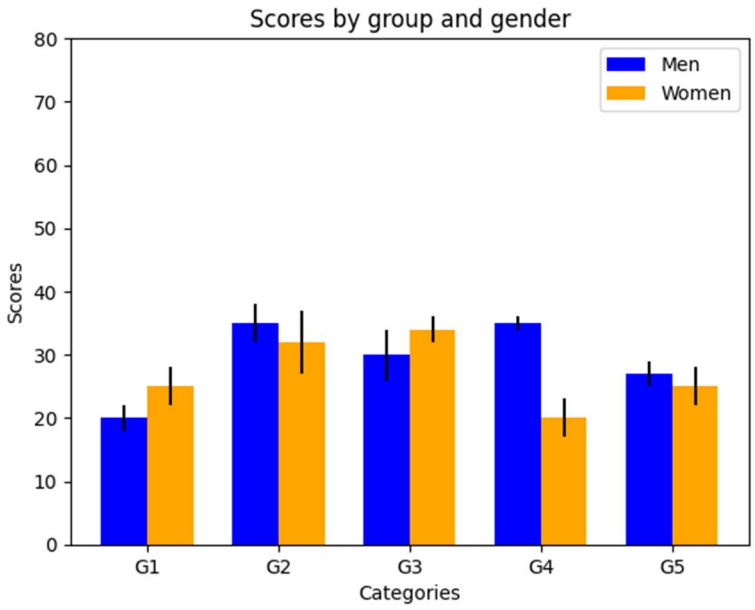


ID = 41
Score vis = 60
Score task = 95
Score human = 100

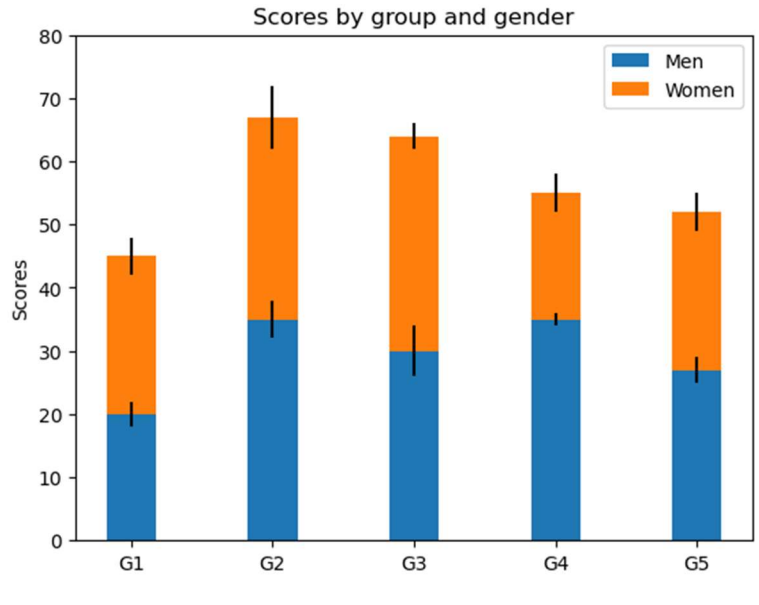
Task: Create a grouped bar chart that visually compares the means and variability (standard deviation) of scores between men and women across different categories specified in the dataframe. Each category should have two bars (one for men and one for women), the bars should be grouped side by side, and error bars should represent the standard deviations.

Style: Customize the plot with a title "Scores by group and gender". Label the y-axis as 'Scores' and configure the x-axis to show category labels from the 'categories' column of the dataframe. Use different colors for men and women to differentiate between the bars. Include a legend to identify color coding. Set the ticks on the y-axis from 0 to 80 with intervals of 10.

Generated



Ground truth

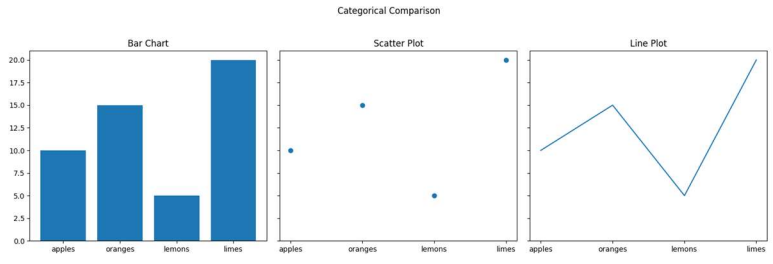


ID = 42
Score vis = 90
Score task = 95
Score human = 100

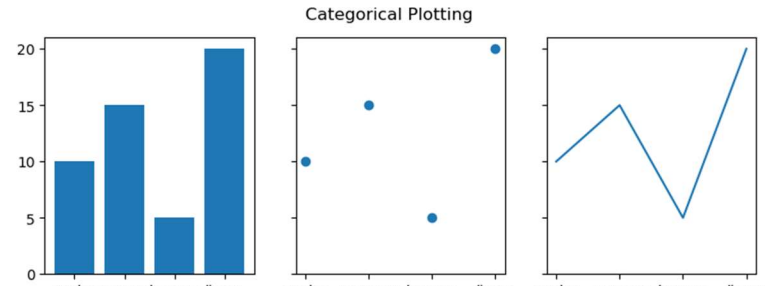
Task: The task involves creating three different plots displayed horizontally in one row: a bar chart, a scatter plot, and a line plot. Each plot should use the 'names' column for the x-axis and the 'values' column for the y-axis. Each subplot should share the same y-axis to maintain uniform scaling across the plots.

Style: Adjust the overall size of the figure to ensure readability and maintain proportional spacing between the plots. Include a super title across the top of the figure to label it as a categorical comparison. The visual style should be clean and clearly distinguish each type of plot for easy comparison.

Generated



Ground truth

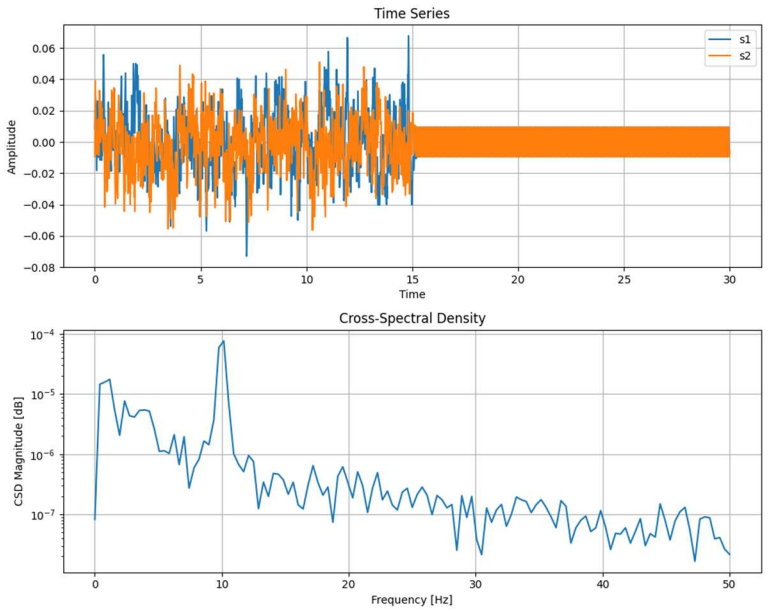


ID = 43
Score vis = 85
Score task = 95
Score human = 85

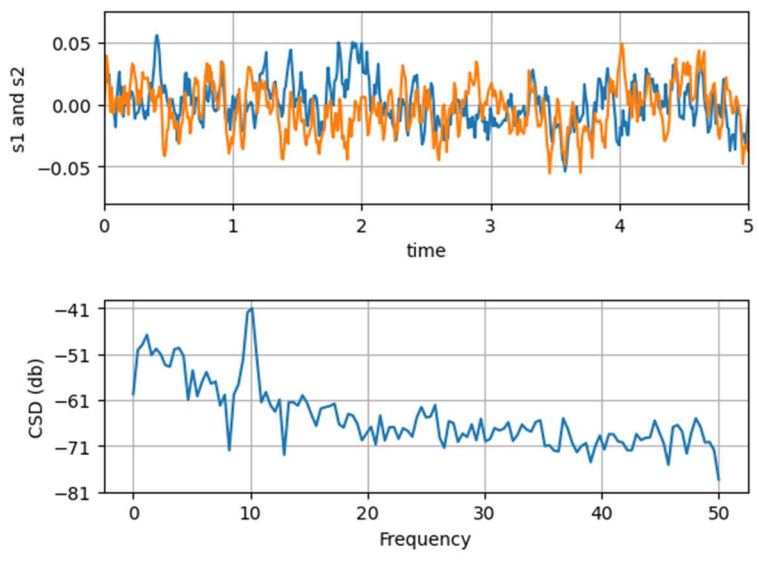
Task: Create a plot with two vertically stacked subplots. The first subplot should display two time series, 's1' and 's2', against 'time'. The second subplot should show the Cross-Spectral Density (CSD) between 's1' and 's2' calculated at specified parameters.

Style: Adjust layout for clear visibility of both subplots. The top subplot should have grid lines and labeled axes. The bottom subplot should show the CSD values in decibels with frequency on the horizontal axis and CSD magnitude on the vertical axis. Set appropriate axis limits and labels for both plots.

Generated



Ground truth

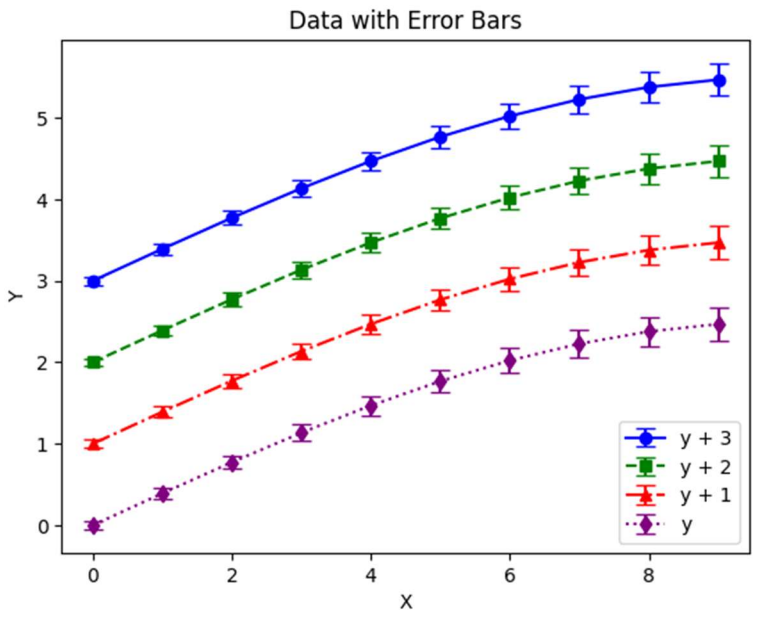


ID = 44
Score vis = 60
Score task = 95
Score human = 75

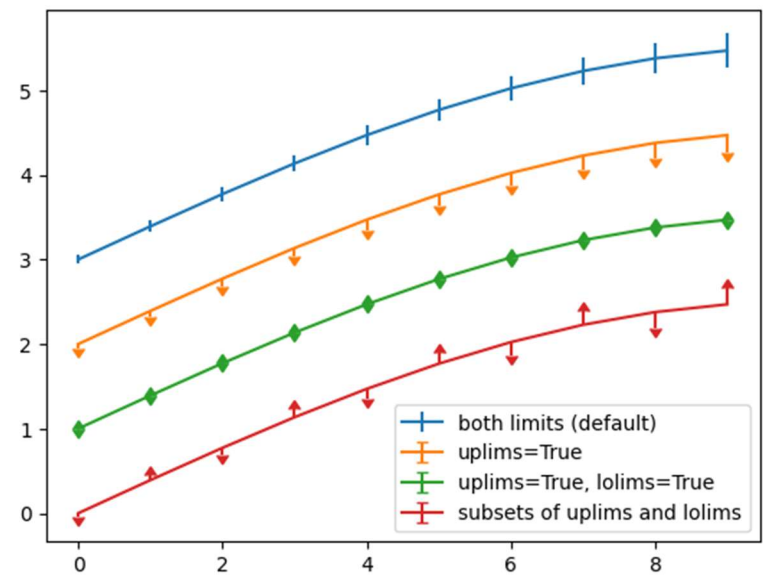
Task: Create a plot that includes multiple lines where each represents data and error limits from the dataframe. Each line corresponds to a different adjustment of error bars (like upper limits, lower limits, or both). Add legends to distinguish between these different error adjustments.

Style: Enhance the plot visually by using different colors for each line. Use markers to indicate the data points, utilize error bars to represent uncertainty, and apply line styles to differentiate between the varying error bar settings. Place a legend in the lower right corner to guide the viewer in interpreting the lines with respect to their associated dataset and error bar settings.

Generated



Ground truth

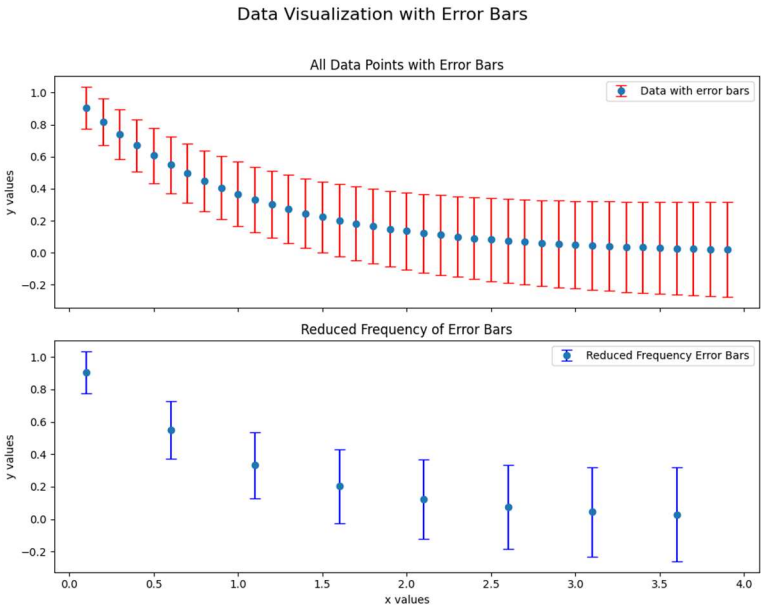


ID = 45
Score vis = 70
Score task = 95
Score human = 100

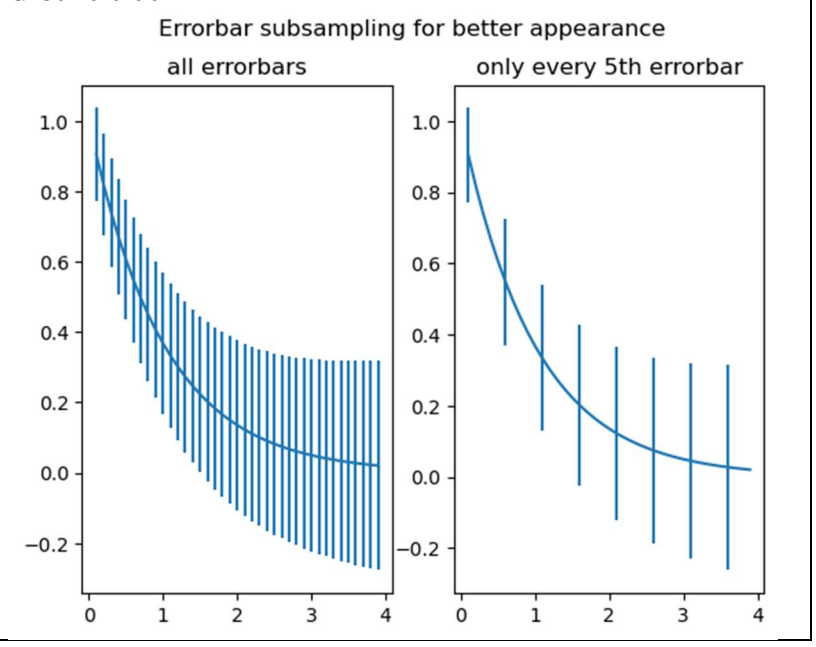
Task: Create a set of subplots where the first plot visualizes all data points with error bars, and the second plot displays error bars at a reduced frequency for clarity. Each plot should share the same x-axis to maintain consistency in comparison.

Style: gn the plots ensuring readability and clarity. Include error bars for visualization of uncertainties. Set titles for individual subplots to describe the data representation specifically, and use an overarching title for the combined figure to unify the theme of the analysis. Adjust visual elements such as color and style to enhance the presentation and interpretability of the data.

Generated



Ground truth

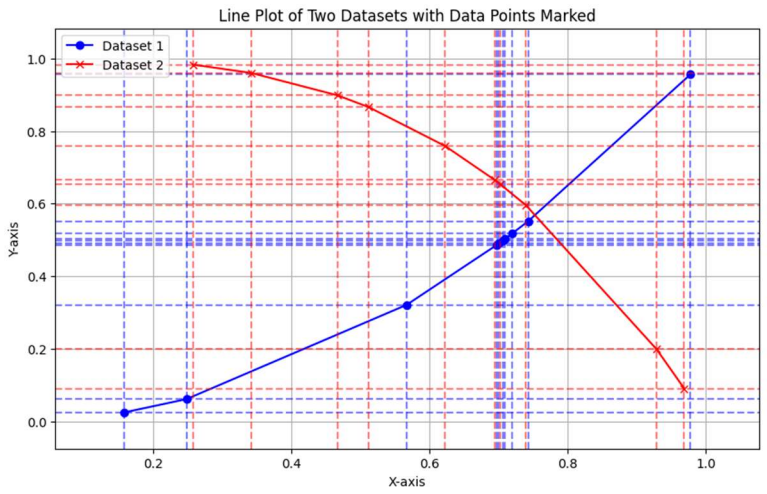


ID = 46
Score vis = 30
Score task = 90
Score human = 100

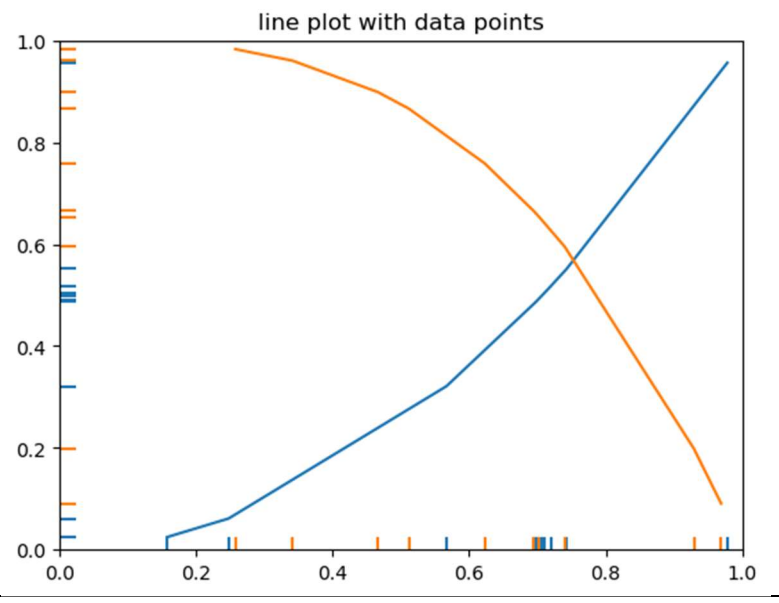
Task: Create a line plot for each x-y data pair. The plot should include distinct lines for each dataset to visually separate them. After plotting lines, enhance the graph by marking individual data points for x and y coordinates that correspond to each dataset using vertical and horizontal lines.

Style: The plot should be clear and well-designed for readability. Use contrasting colors to differentiate between the two datasets. Additional marks for data points should be subtle yet visible, extending across a small fraction of the plot area without overpowering the line plots. Set plot limits to encapsulate all data points comfortably and add a title for context.

Generated



Ground truth

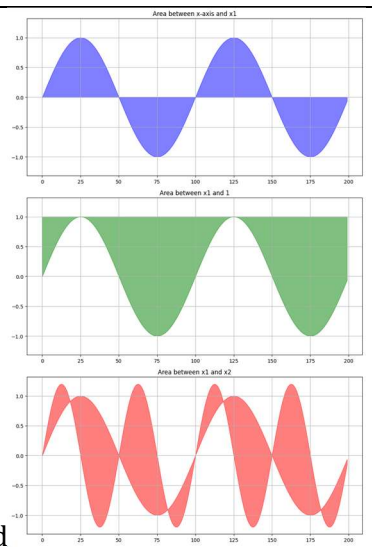


ID = 47
Score vis = 10
Score task = 100
Score human = 100

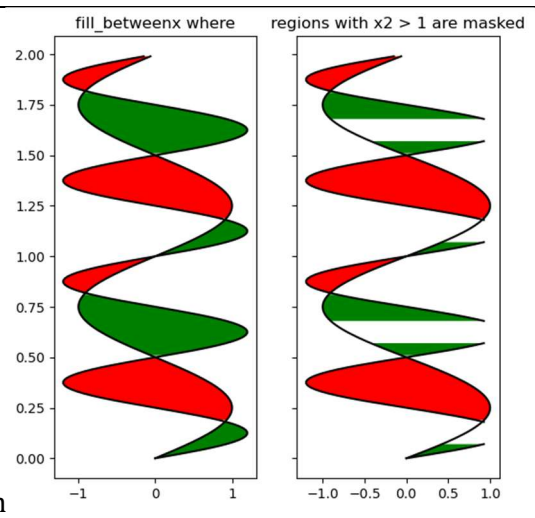
Task: Create multiple line and area plots over two figures with shared y-axes:
- In the first figure, plot three subplots:
  1. Area between x-axis values 0 and values from `x1` column.
  2. Area between x-axis values from `x1` column and 1.
  3. Area between x-axis values from `x1` column and values from `x2` column.
- In the second figure, plot two subplots:
  1. Overlaid line plots of y vs. `x1` and y vs. `x2` with areas filled based on conditions comparing `x1` and `x2`.
  2. Overlaid line plots of y vs. `x1` and y vs. a masked version of `x2` (`x2_masked`) with areas filled under the same conditions as above but depicting only regions where `x2` is greater than 1 as masked.

Style: Each subplot should distinctly color regions based on the specific condition being visualized. Employ colors to effectively highlight differences, such as red and green to differentiate areas where conditions on `x1` and `x2` vary, and blue tones for simple area fills. Titles should clearly indicate what each subplot represents. All x-axis labels, grid lines, and other stylistic elements should be properly aligned to enhance readability and visual appeal.

Generated



Ground truth

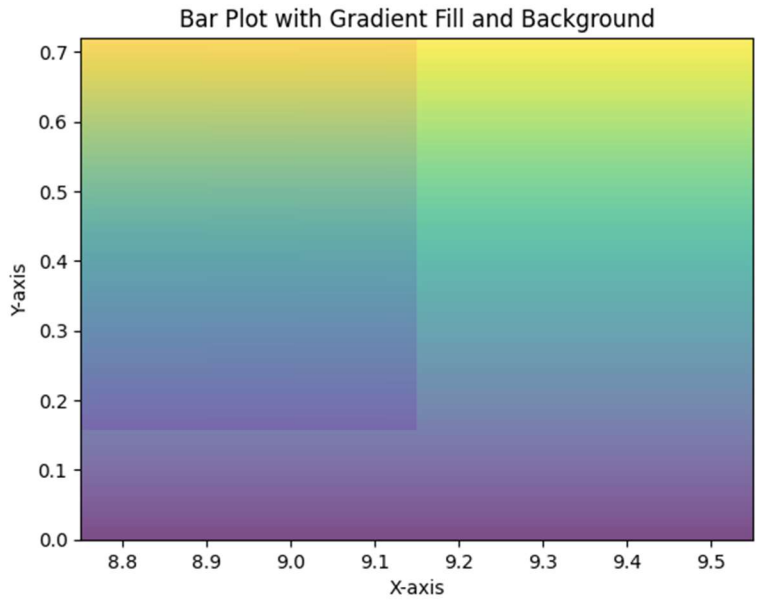


ID = 48
Score vis = 10
Score task = 20
Score human = 15

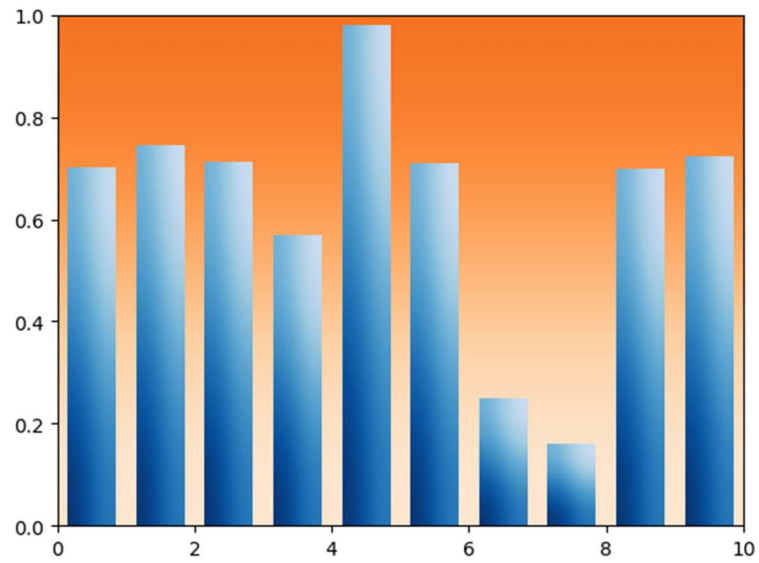
Task: The task is to create a bar plot where the bar heights are determined by the 'y' values and the position of each bar corresponds to the 'x' values of the DataFrame. Additionally, each bar should have a gradient fill which reflects a visual gradient effect. Furthermore, an overall background gradient image needs to be added to the plot area.

Style: Each bar in the plot should exhibit a vertical gradient using a specified color map (from full color at the top to faded at the bottom of the bar). The entire background of the plot should also display a horizontal gradient with a different specified color map. The plot should have clearly marked axes and the aspect ratio should be adjusted for better visual representation.

Generated



Ground truth

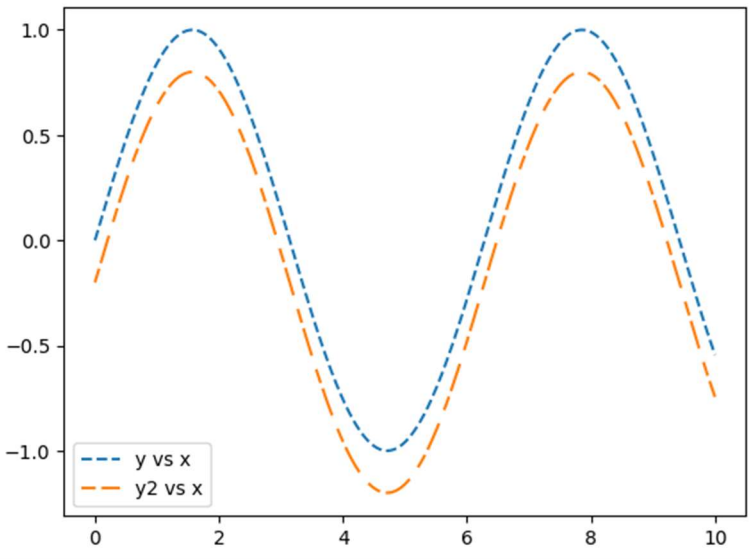


ID = 49
Score vis = 95
Score task = 95
Score human = 100

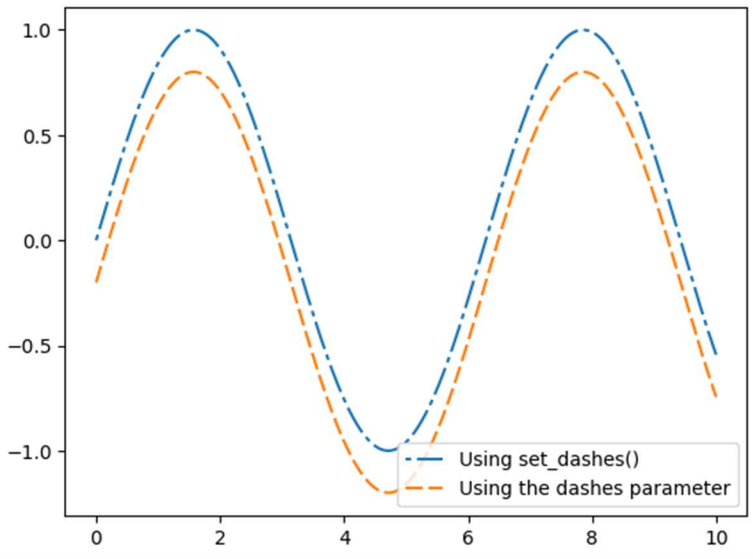
Task: description: Generate a line plot using the data from the dataframe. Plot two lines; one line will be for the 'y' values against 'x', and the other line for 'y2' values against 'x'. Incorporate a mechanism to style each line differently in terms of dashing patterns to distinguish between the two.

Style: description: Customize the appearance of the two lines to enhance their distinction visually. Apply a specific dashing pattern to one line directly during its creation, while adjusting the other line's pattern post-creation using a method that modifies its dash-sequence properties. Include a legend to label each line distinctly according to its dashing style.

Generated



Ground truth

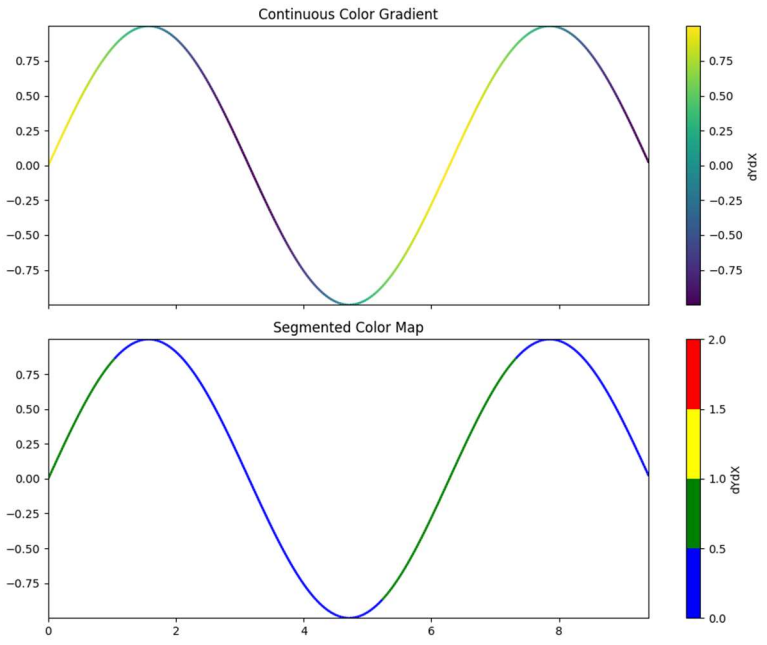


ID = 50
Score vis = 90
Score task = 100
Score human = 90

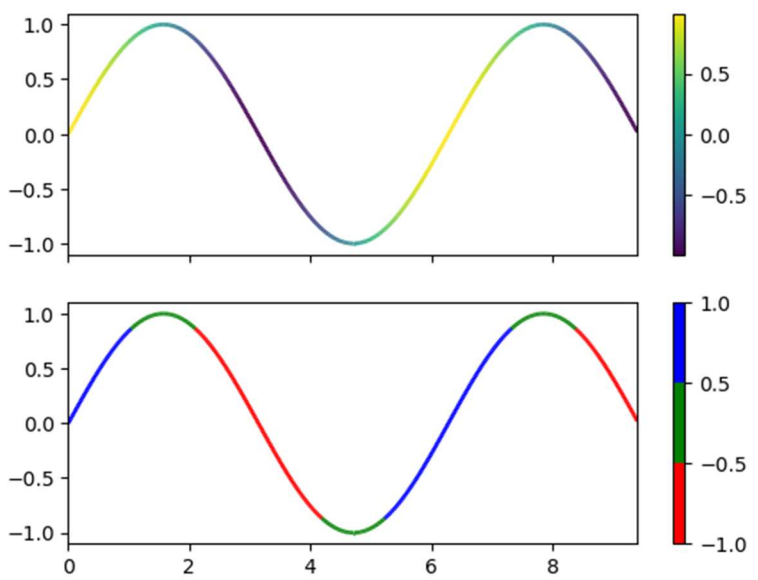
Task: The goal is to create a plot consisting of two subplots sharing the same x and y axes. Each plot should display a line graph where segments between points are individually colored based on the derivative values ('dYdX'). The first plot will use a continuous color gradient to reflect the range of derivative values, and the second plot will use distinct color segments to categorize these values into intervals.

Style: For the first subplot, apply a continuous color map to visualize the smooth transition across a spectrum reflecting the range of derivative values. In the second subplot, use a segmented color map with defined boundaries to demarcate specific ranges of the derivative with distinct colors. Both plots should include color bars indicating the mapping of the derivative values to the respective colors used in the plot. Additionally, set the width of the line segments and adjust the axes limits to appropriately encompass all data points with a slight margin.

Generated



Ground truth

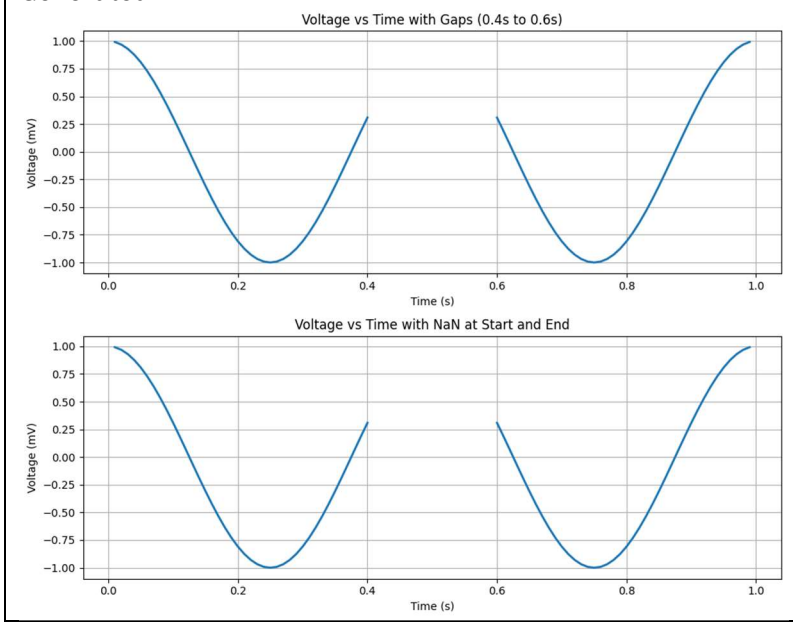


ID = 51
Score vis = 90
Score task = 95
Score human = 100

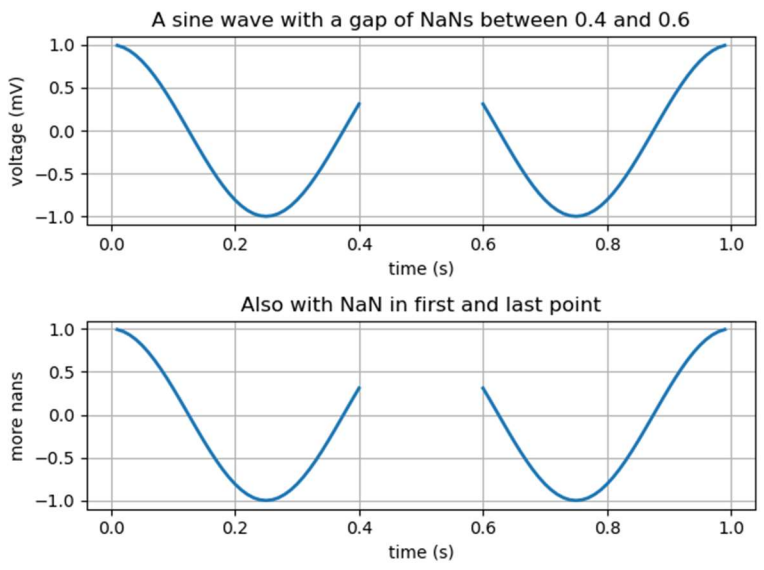
Task: Create two subplots arranged vertically. In the first subplot, graph the 'voltage(mV)' against 'time(s)', clearly indicating any gaps due to NaN values between 0.4 and 0.6 seconds. In the second subplot, graph the same 'voltage(mV)' against 'time(s)' but include NaN values at the start and end of the dataset to showcase differing data handling. Both graphs should have axis labels and titles denoting the peculiarities of the data representation. Add a grid for better readability of the plots.

Style: Use a line plot with a line width of 2 for visibility. The plots should include grid lines for easier interpretation of the data points. Set specific titles for each subplot to reflect the unique aspects of the data shown in each plot. Ensure the layout is tight so that labels and titles do not overlap. Display both plots in a clean and orderly manner. Adjust the layout to prevent any overlap or congestion in the visual presentation.

Generated



Ground truth

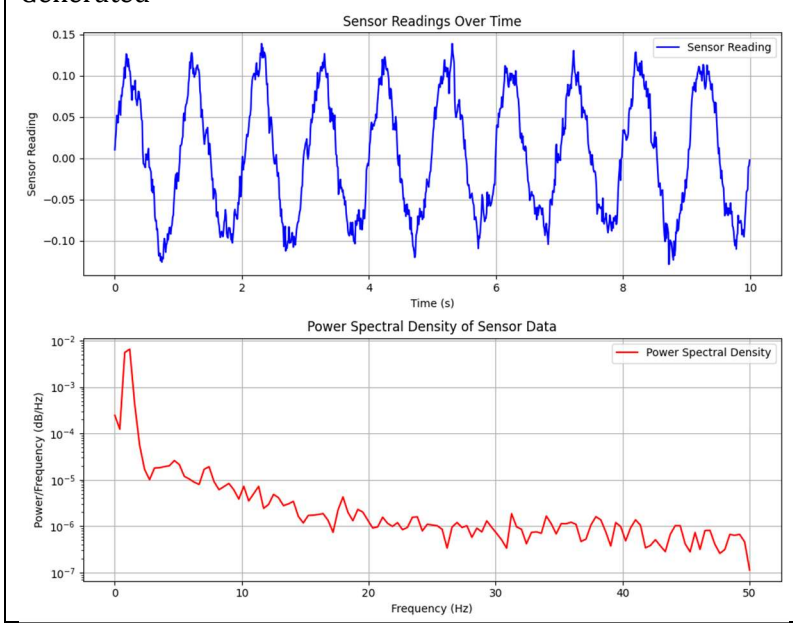


ID = 52
Score vis = 90
Score task = 95
Score human = 100

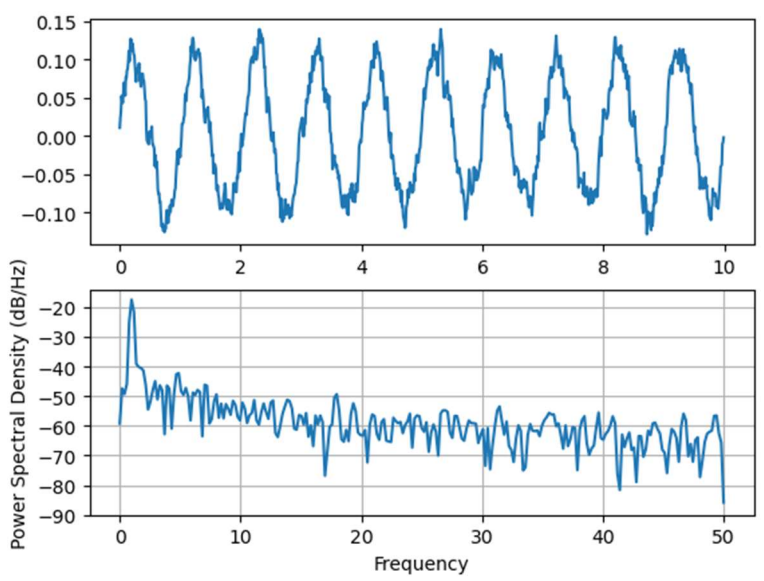
Task: Generate a plot containing two subplots. The first subplot should display a line chart of sensor readings over time. The second subplot should analyze the frequency components of the sensor data using a power spectral density (PSD) plot, considering a specific frequency rate.

Style: he plots with clearly defined axes and appropriate labeling. Ensure the line chart reflects changes over time with smooth curves or lines. The PSD plot should visibly outline the power distribution across different frequencies in a descriptively manner, using contrasting colors or styles for better visibility.

Generated



Ground truth

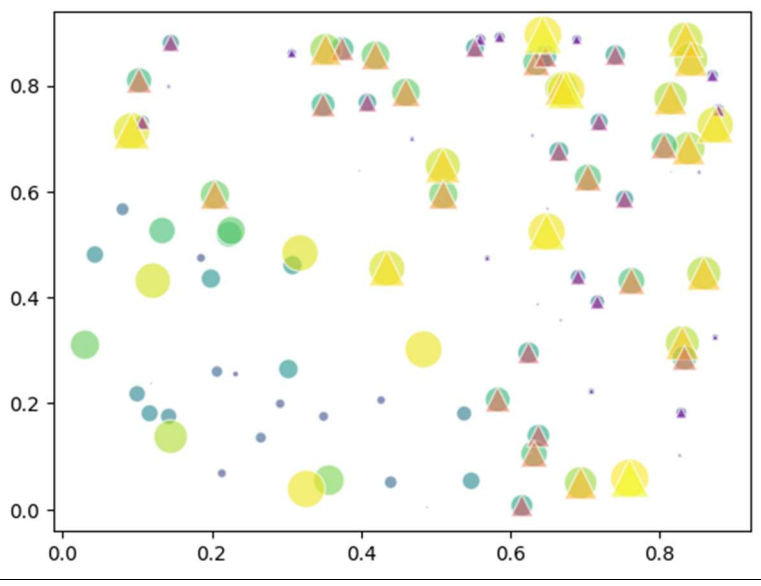


ID = 53
Score vis = 30
Score task = 70
Score human = 70

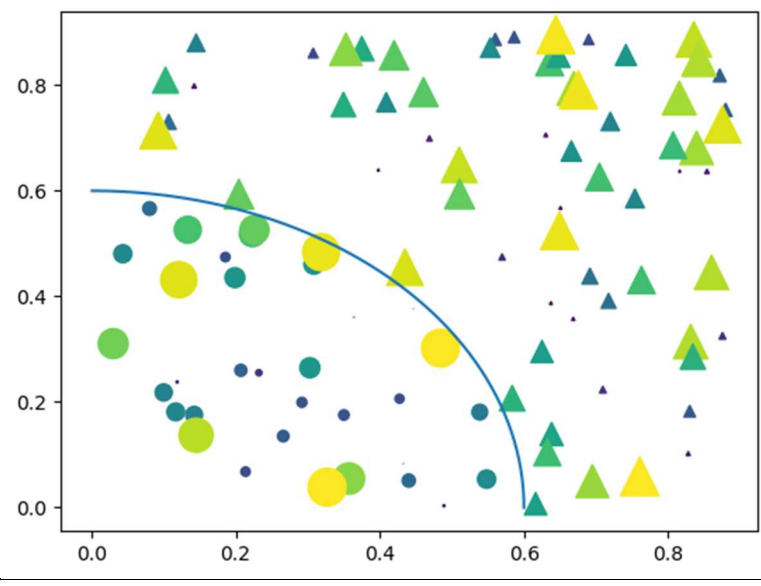
Task: The plot should include two scatter plots overlaid on each other, each representing sets of points using different areas for sizing and different markers. Additionally, a curved line should be plotted that indicates a boundary or threshold within the plot space.

Style: For the scatter plots, use triangular and circular markers to differentiate between two series in the dataset. Size these markers based on related area columns. Apply color scaling to these points based on another column to represent a third variable visually. The line should be smooth and curved, visually defining a boundary between different regions on the plot.

Generated



Ground truth

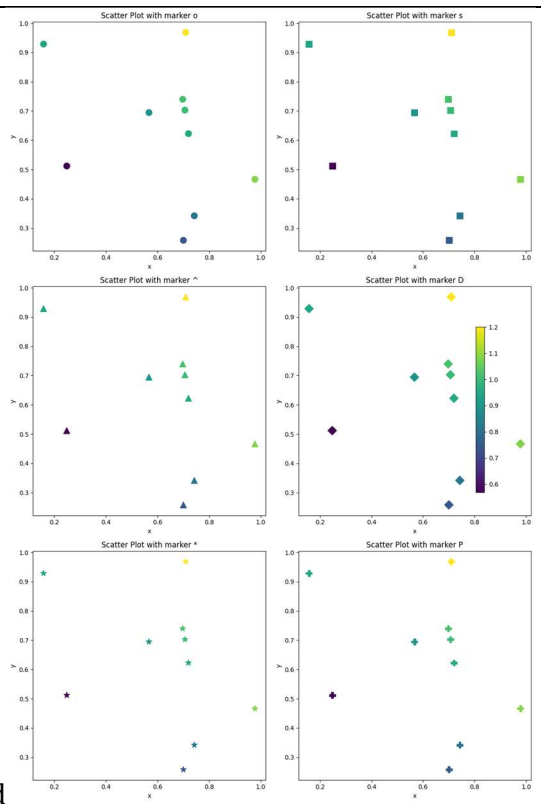


ID = 54
Score vis = 80
Score task = 95
Score human = 100

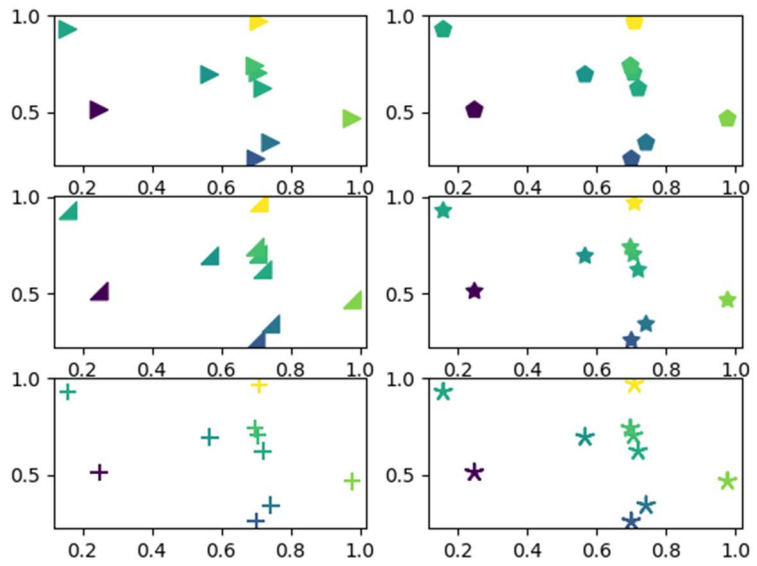
Task: Construct a multi-plot figure with six subplots arranged in a 3x2 grid. Each subplot will display a scatter plot where 'x' and 'y' columns provide the data points' positions, and the 'z' column determines the color of each point. Use different marker styles for each subplot to demonstrate various scatter plot customizations.

Style: he scatter plots consistently regarding point size and color mapping across all subplots, but vary each subplot's marker shape. This variation will include predefined shapes such as triangles and stars, as well as custom shapes defined by vertices coordinates and tuples. Ensure each subplot has clearly distinguishable markers while maintaining overall harmonious appearance.

Generated



Ground truth

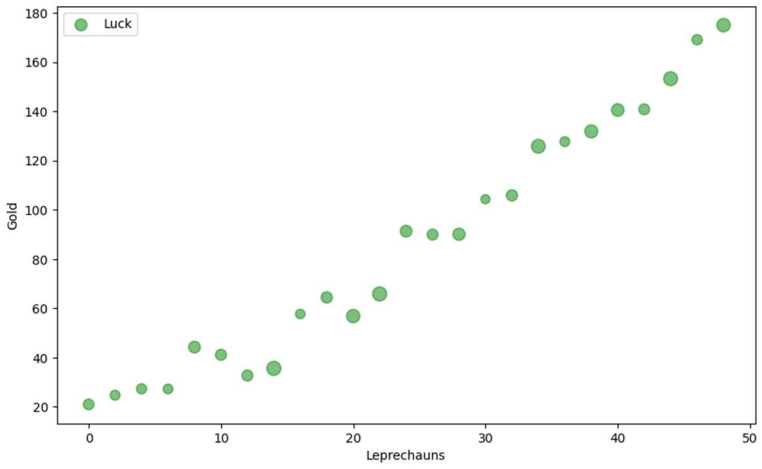


ID = 55
Score vis = 90
Score task = 90
Score human = 90

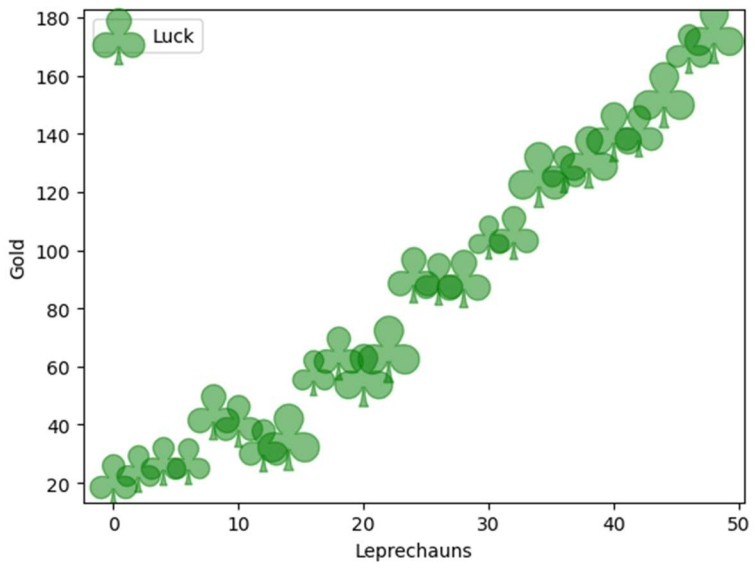
Task: Generate a scatter plot representing the relationship between 'Leprechauns' and 'Gold'. Use the 'Luck' values to adjust the size of the scatter points dynamically. The color of the scatter points should be consistently green, and each point should be represented with a specific marker symbolizing a suit, indicative of the 'Luck' attribute.

Style: Configure the graph with properly labeled axes for 'Leprechauns' and 'Gold'. The sizes of the scatter marks (reflecting 'Luck') should have varying transparency to better visualize density and overlap. Include a legend in the upper-left corner to identify the 'Luck' attribute visually keying to the marker shape and color. Set a green color theme for markers to maintain visual appeal and coherence with the theme of luck.

Generated



Ground truth

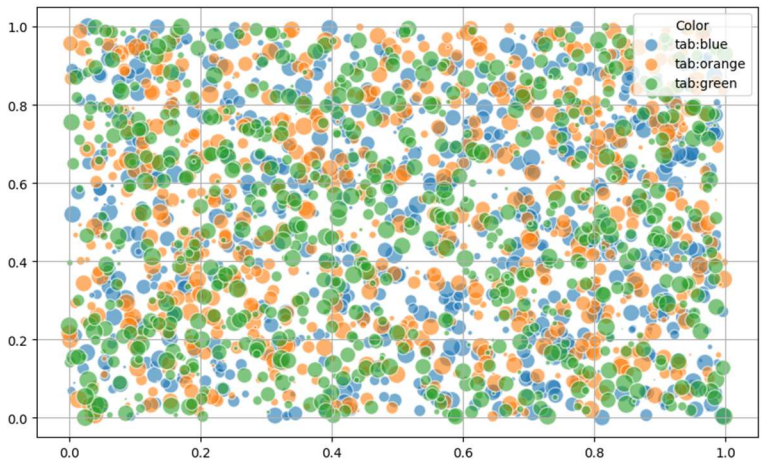


ID = 56
Score vis = 95
Score task = 95
Score human = 100

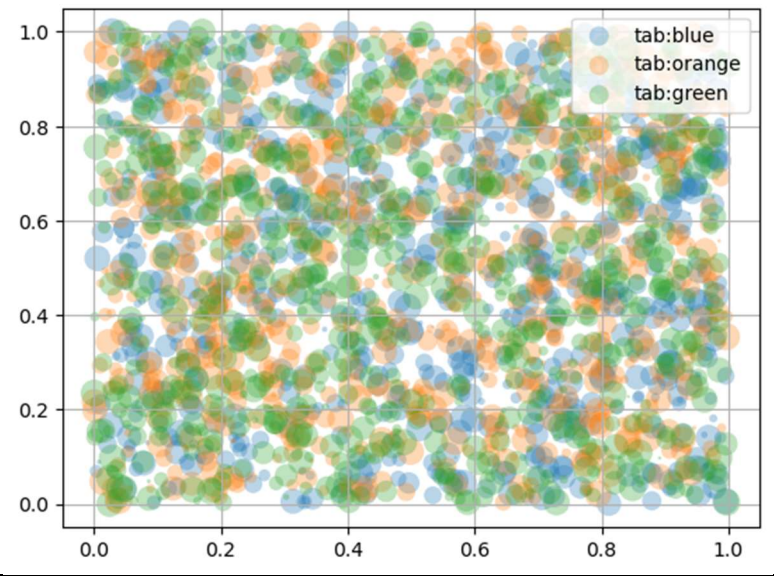
Task: duce a scatter plot for the provided DataFrame where each data point's X and Y coordinates are plotted, categorized by color, and with varying sizes according to 'scale'. Each unique color in the 'color' column defines a different group, and points should be plotted with an associated legend identifying each color. Transparency should be adjusted for better visualization.

Style: Apply a grid to the plot background for better readability of data points. Use a non-bordered style for each data point to maintain the plot's clarity. Legends should be included to map colors to their corresponding data categories. Adjust the opacity to ensure that overlapping points are visually distinguishable.

Generated



Ground truth

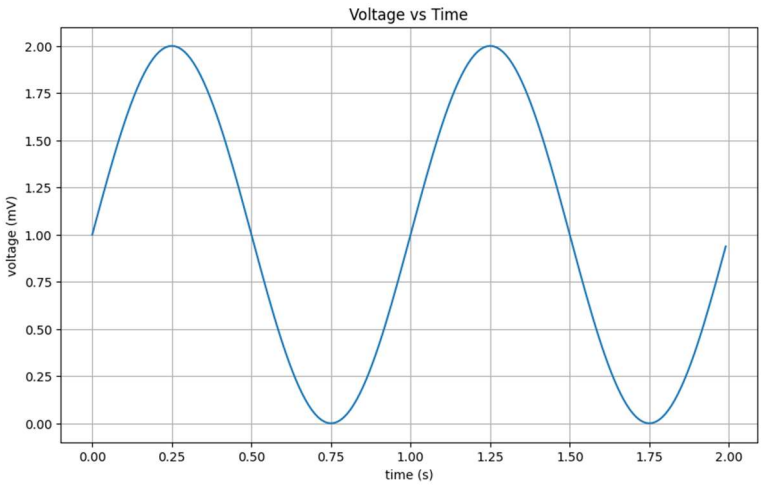


ID = 57
Score vis = 95
Score task = 100
Score human = 100

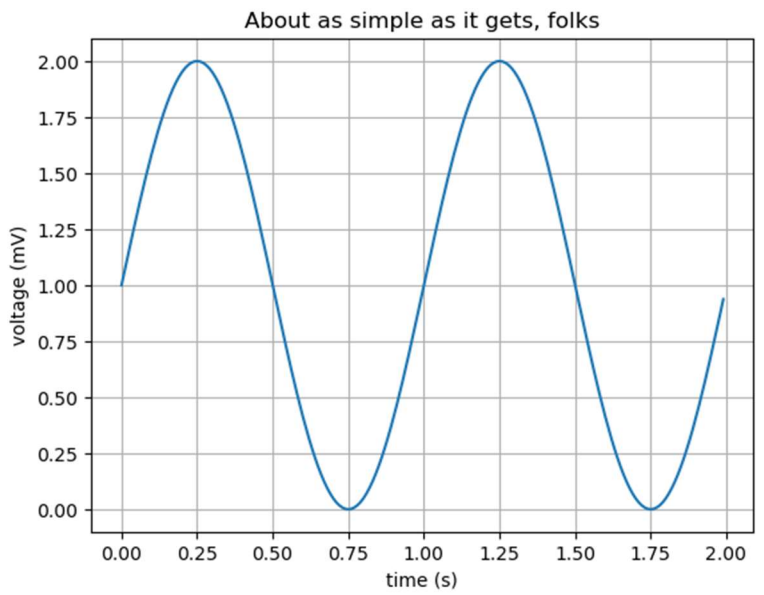
Task: Create a line plot using the provided DataFrame. The plot should use the 'time' column for the x-axis and 'voltage' column for the y-axis. After plotting, label the x-axis as 'time (s)' and y-axis as 'voltage (mV)'. The plot should also have an appropriate title.

Style: Make sure the plot includes a grid for easier readability of the values. The lines on the plot should clearly represent the trend of voltage over time. Also ensure that the plot is saved as an image file for further usage.

Generated



Ground truth

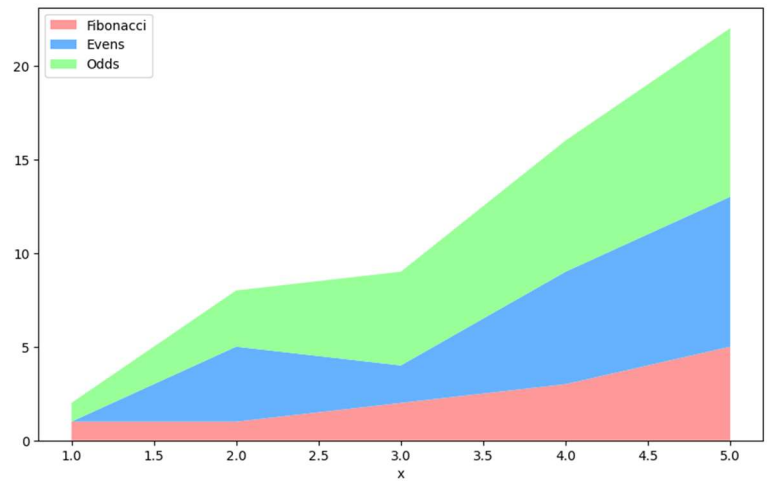


ID = 58
Score vis = 95
Score task = 95
Score human = 100

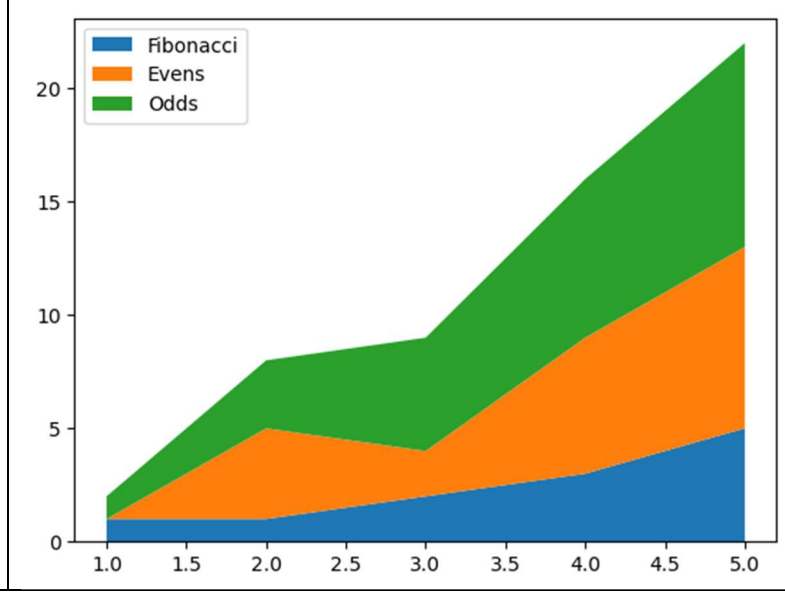
Task: Create an area plot where values of specific series 'Fibonacci', 'Evens', and 'Odds' are stacked upon one another across the x-axis values. Include a legend in the plot, describing each data series for better readability.

Style: The plot should be styled to clearly differentiate between the stacked areas with distinct colors for each data series. Position the legend in the upper left corner to enhance plot clarity and aesthetics. The x-axis should be properly labeled and visible, providing a clear index of progression for the data values. The overall appearance should be tidy and engaging, ensuring legibility all elements, especially in distinguishing between the different series stacked in the plot.

Generated



Ground truth

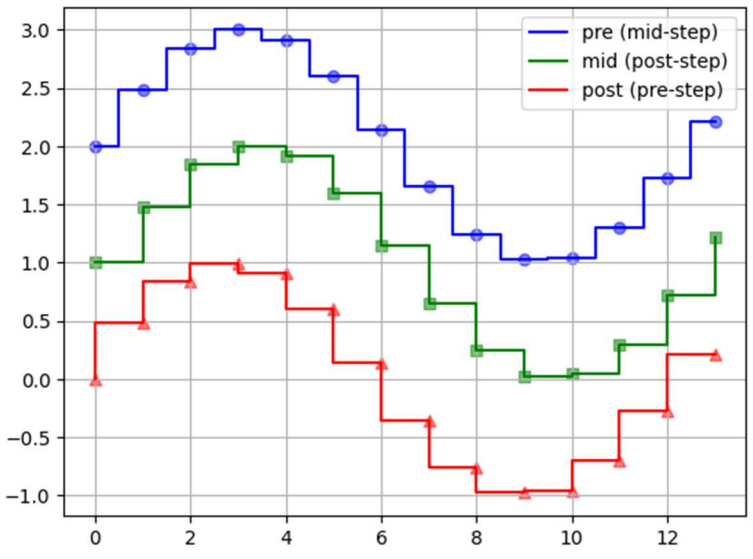


ID = 59
Score vis = 90
Score task = 95
Score human = 100

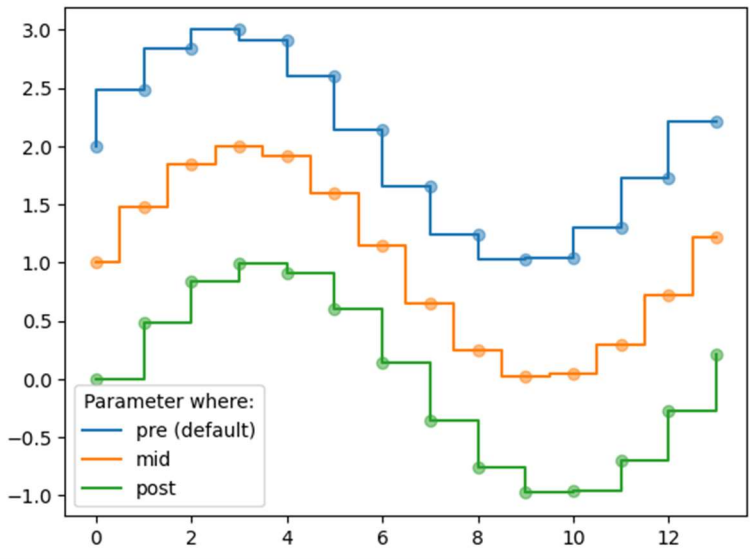
Task: hree sets of data ('pre', 'mid', 'post') from the dataframe against 'x'. Each dataset should be represented as both a step plot and a scatter plot on the same graph. Each type of data should be plotted distinctly, with variations in how the steps transition between data points.

Style: Use different colors and markers for each of the data series to enhance visual distinction. Add transparency to the scatter plots to differentiate them from step plots. Include a plot legend to identify each series and reference the type of step transition used for clarity, positioning this legend appropriately. Display axes and grid lines for better readability of plotted data points.

Generated



Ground truth

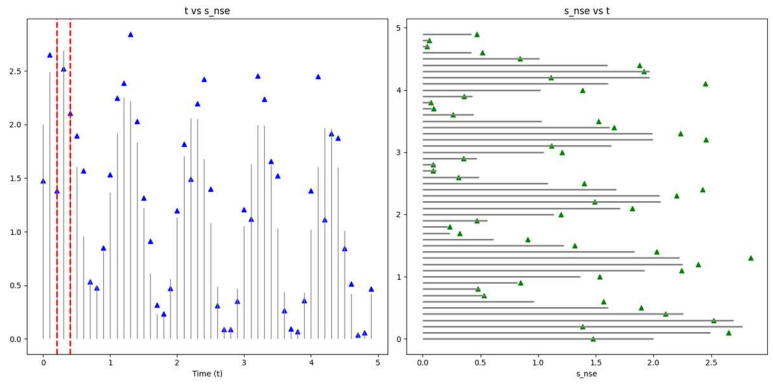


ID = 60
Score vis = 90
Score task = 95
Score human = 100

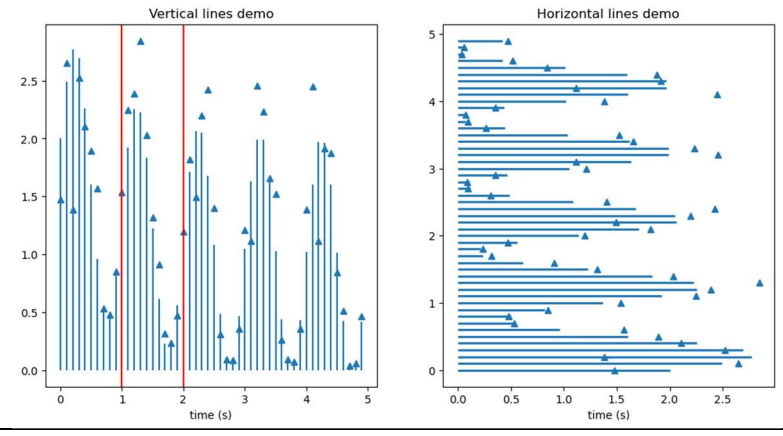
Task: description:
- Create two subplots side by side;
- First subplot (left side): Plot 't' vs 's_nse' using a scatter plot marked with triangles. Add vertical lines at every 't' value, spanning from 0 to corresponding 's' values. Include additional static vertical lines at specific 't' values colored in red.
- Second subplot (right side): Plot 's_nse' vs 't' using scatter plot with triangles pointed upwards. Draw horizontal lines at each 't' value, running from 0 to corresponding 's' values with increased line width.
- Include x-labels for time and titles for each subplot.

Style: description:
- Adjust the plot size appropriately for clarity and visual appeal.
- Style the scatter plots with triangle markers to differentiate the points clearly.
- Use distinct colors for static lines (like red) to highlight specific features within the plots.
- Customize line widths to enhance the visibility of lines in the plot.

Generated



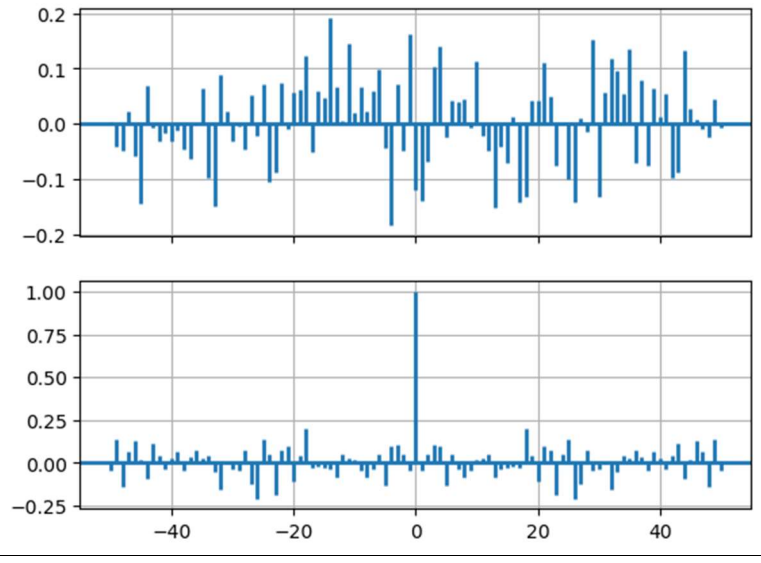
Ground truth



ID = 61
Score vis = 0
Score task = 0
Score human = 0

Task: description: The plot consists of two subplots arranged vertically and sharing the x-axis. The first subplot displays the cross-correlation of columns 'x' and 'y', indicating the degree to which they shift relative to each other. The second subplot shows the autocorrelation of column 'x', exploring how this column correlates with itself at different lags.

Style: description: Both subplots use vertical lines to mark significant relationships and are normalized. Line widths are consistent across plots, enhancing visual clarity. Grids are added to both plots to improve readability and data interpretation. The line width is set to a moderate thickness to ensure clear visibility of trends and correlations.

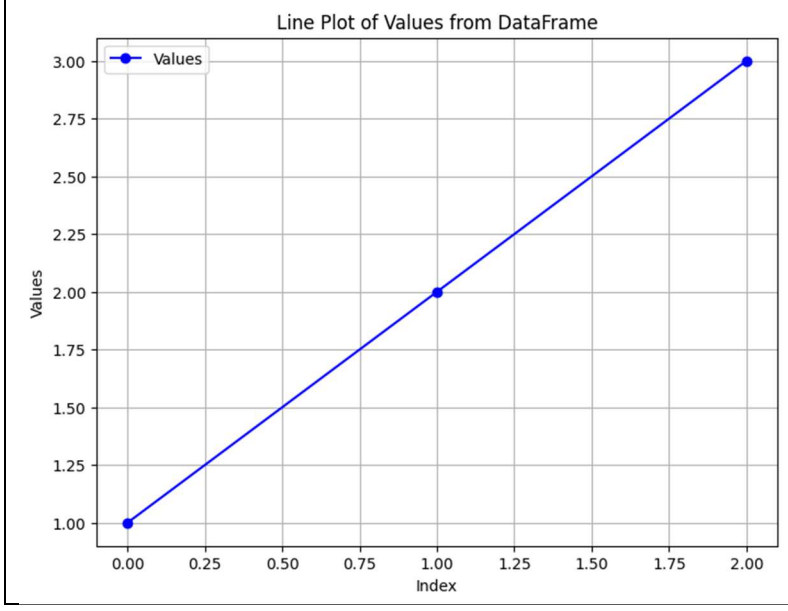
Generated	Ground truth
	

ID = 62
Score vis = 90
Score task = 90
Score human = 100

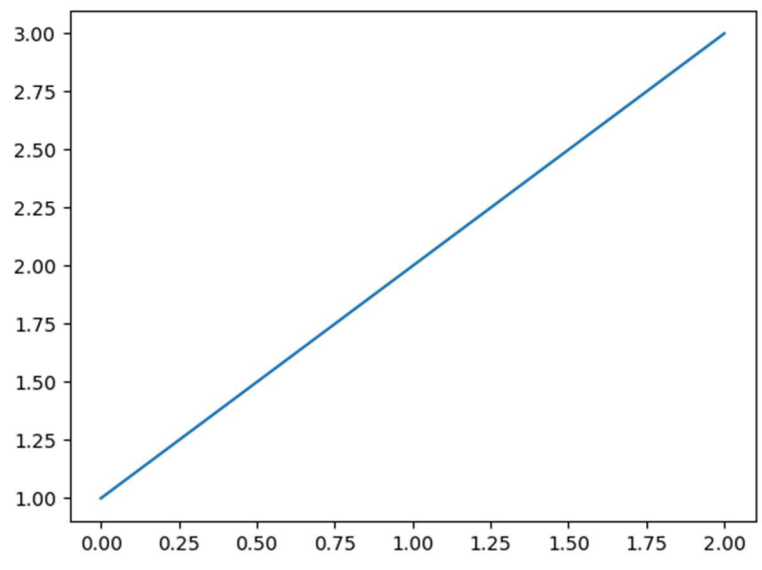
Task: Create a line plot based on the values from the dataframe. The X-axis should reflect the index of the dataframe, while the Y-axis should represent the values from the dataframe's sole column. Ensure that the plot provides a clear visual representation of the data progression or trend.

Style: The visual styling of the plot should include axes that are labeled clearly, and the line should be distinct and easily identifiable. Opt for a simple and clean background with grid lines to facilitate easier reading of the plotted values. The output format should support both display within a GUI and saving to an image file format. The resulting image should preserve the plot's resolution and colors correctly.

Generated



Ground truth

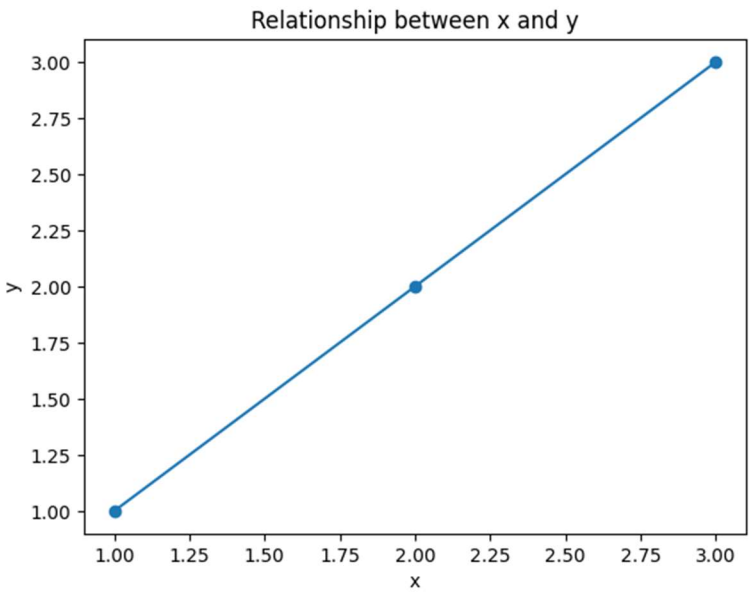


ID = 63
Score vis = 80
Score task = 90
Score human = 100

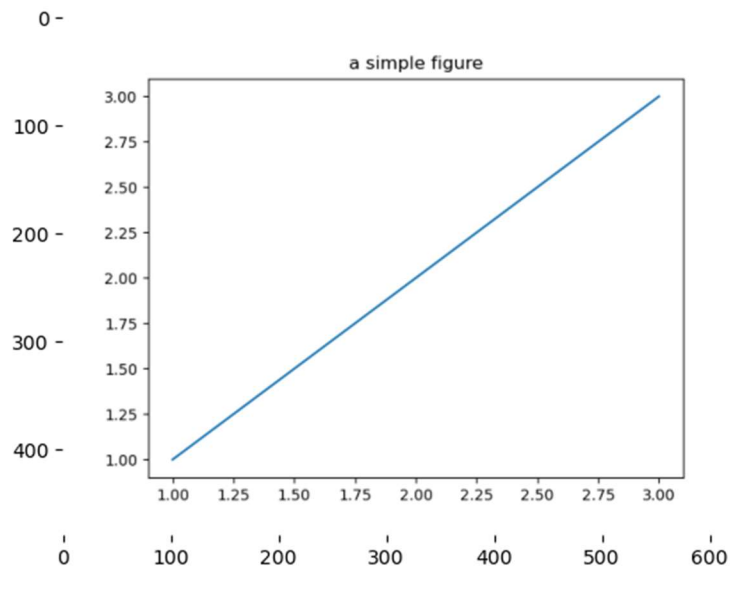
Task: The task involves creating two plots. The first plot graphically represents the relationship between the two dataframe columns by plotting one against the other. A title should be added to this plot. The second part of the task includes taking the rendered image of this plot (captured directly from its pixel buffer), and then displaying this image as a new plot itself.

Style: The first plot should be styled with clear labels, a title, and an appropriate scale on the axes to accurately represent the data points. The second plot involves displaying the pixel buffer as an image on a new plot frame without any frame or axes, focusing solely on the graphical representation of the first plot. This plot should retain clarity and accurately reflect the visual information from the first plot.

Generated



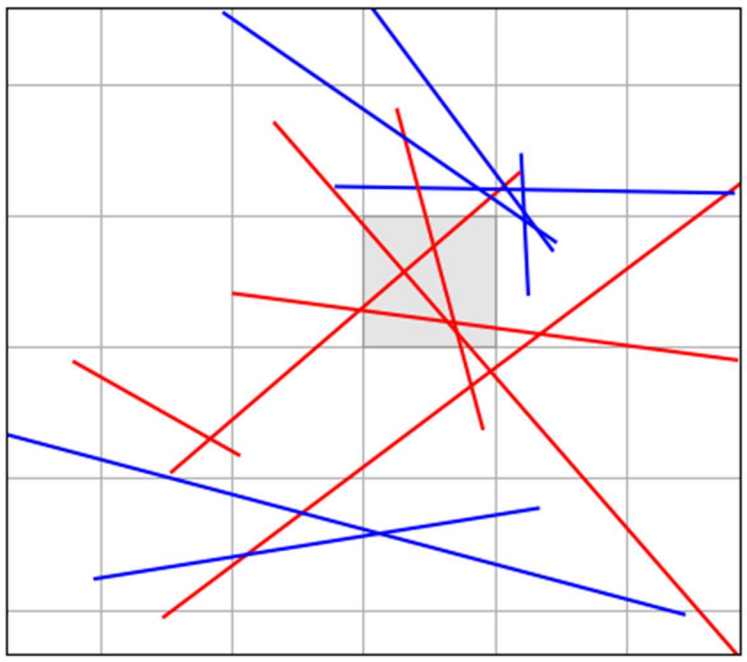
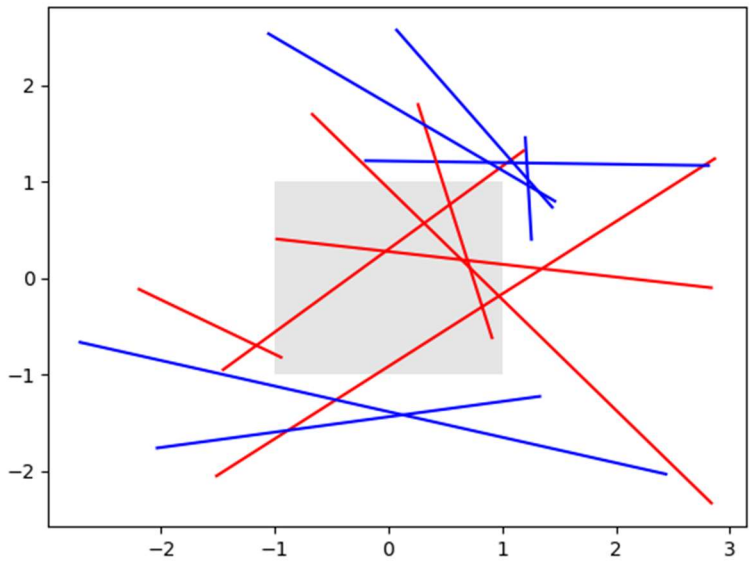
Ground truth



ID = 64
Score vis = 90
Score task = 90
Score human = 100

Task: aw a 2-dimensional plot where each row from the DataFrame will be used to create a line segment on the plot. Each line segment's start and end coordinates are dictated by 'x1', 'y1' for the start, and 'x2', 'y2' for the end of the line. The color of each line is specified in the 'color' column of the DataFrame. Additionally, include a partially transparent rectangle as a background feature on the plot.

Style: Each line should inherit the color specified in the associated DataFrame row. The rectangle background should be black with low opacity. Set appropriate axis limits and aspect ratios to ensure all line segments and the rectangle are comfortably visible within the plot frame. Adjust other aesthetic elements like the plot's gridlines, axes visibility, and tick marks to enhance readability and visual appeal.

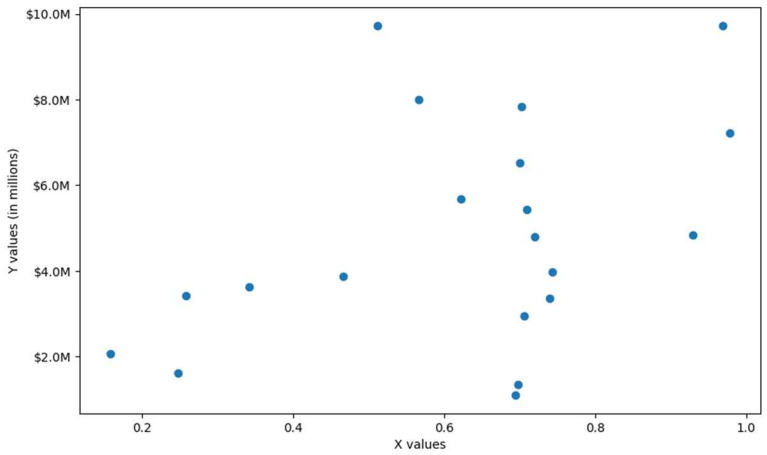
Generated	Ground truth
	

ID = 65
Score vis = 90
Score task = 95
Score human = 100

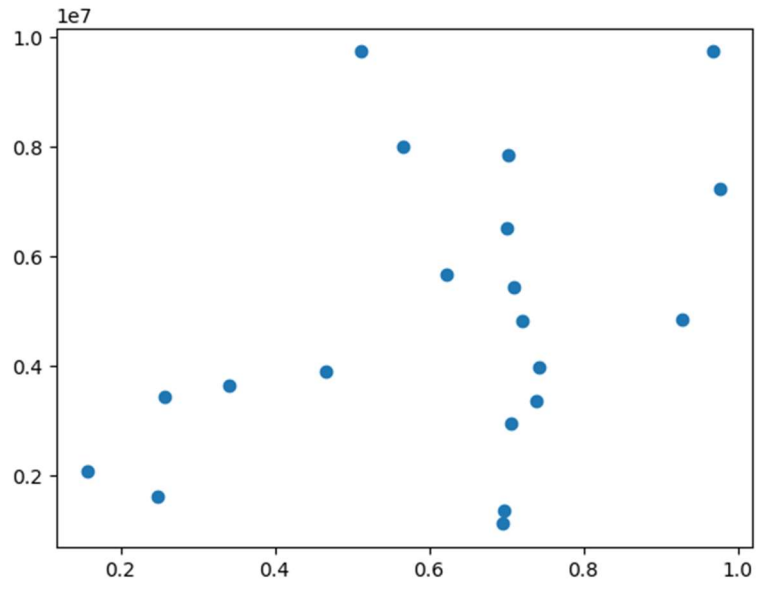
Task: Create a scatter plot using the data from the DataFrame. The 'x' column values should be plotted on the horizontal axis, and the 'y' column values should be plotted on the vertical axis. The vertical axis should display values formatted as millions for easier readability.

Style: The plot should use point markers represented by circles. Ensure that the plot size, axes labels, and other styles are clearly configured to enhance visual appeal and readability. The formatting function for the vertical axis should convert raw numbers into a more readable format, displaying each value as a number in millions with a dollar sign prefix.

Generated



Ground truth

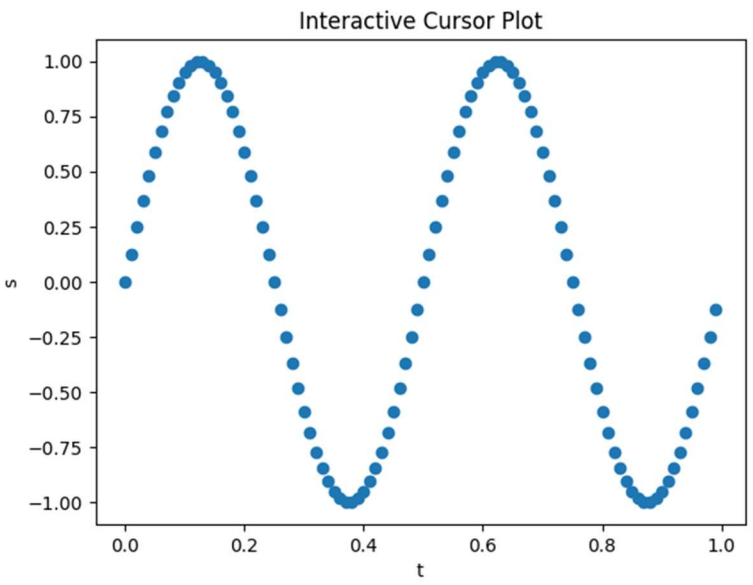


ID = 66
Score vis = 90
Score task = 90
Score human = 90

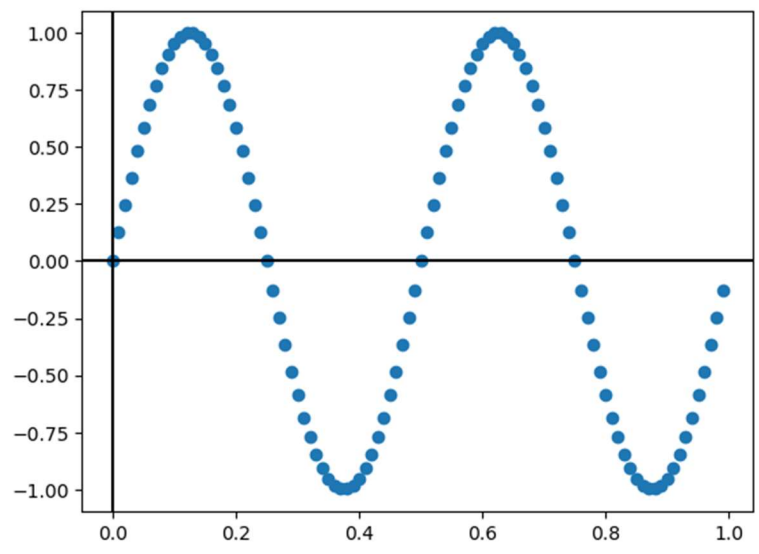
Task: The task involves creating two interactive plots that visualize data points from the 't' and 's' columns of the DataFrame. Each plot should display points as markers. The first plot should have an interactive cursor that displays the precise x and y coordinates of the cursor position. The second plot should include a snapping cursor that jumps to the nearest data point, and updates the displayed x and y values with precise data from the DataFrame.

Style: Each plot should use circular markers for data points. The interactive cursor and snapping cursor should display as crossed lines (horizontal and vertical) with the current coordinates shown in text on the plot area. The design should ensure legible and clear visualization of data points and cursor positions, maintaining a consistent style between plots.

Generated



Ground truth

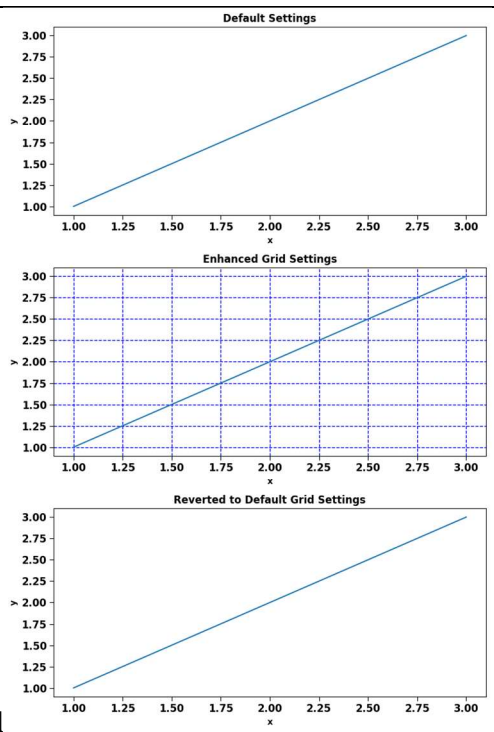


ID = 67
Score vis = 80
Score task = 100
Score human = 100

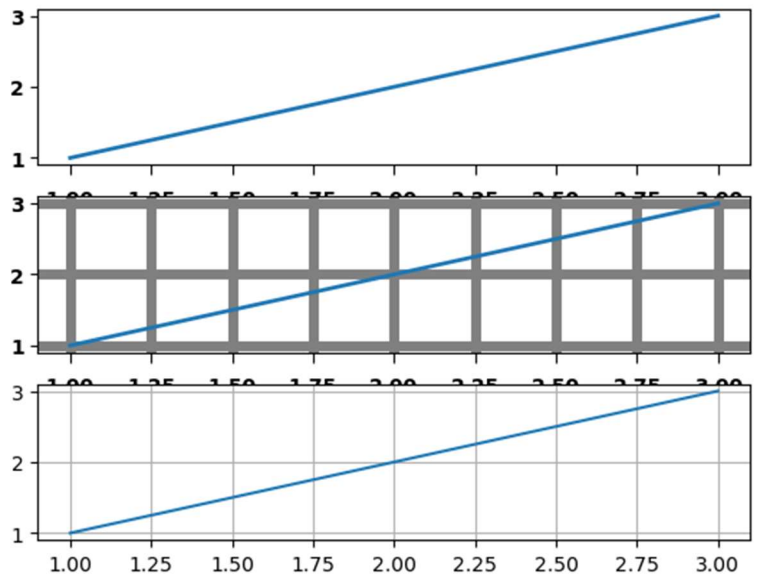
Task: Construct a series of three subplots vertically stacked. Each subplot should plot the same 'x' versus 'y' data. The default settings apply to the first plot, while specific enhancements and grid settings apply to the subsequent plots to differentiate them visually and highlight various features of the data.

Style: Apply global style changes at the beginning, setting font attributes to be bold and adjusting tick properties for clarity and aesthetics. Employ a specific styling for the grid in the second and third subplots, altering color, linestyle, and linewidth to make the grid prominent. These style settings should revert to defaults before the final subplot to demonstrate the effect of global style settings versus localized, temporary adjustments. Each subplot visually confirms the effect of these styling choices on the data presentation.

Generated



Ground truth

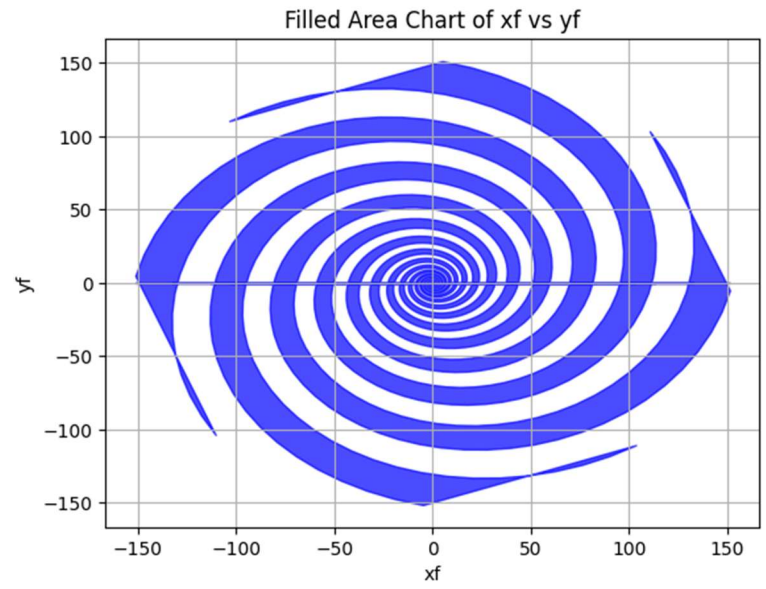


ID = 68
Score vis = 95
Score task = 100
Score human = 100

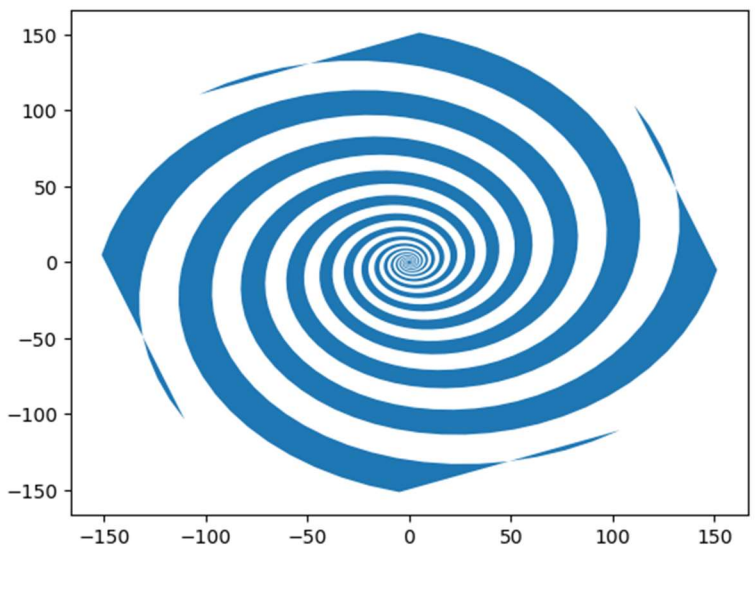
Task: Create a plot using the values from the dataframe with 'xf' as horizontal coordinates and 'yf' as vertical coordinates. The data points should not be plotted as individual points but should instead be represented in a manner that emphasizes the continuity and connectivity between points, implying a filled area chart.

Style: The plot should fill the area under a curve that is formed by connecting the data points sequentially from the dataframe. Emphasize a clear, boundary-defined filled style that highlights the shape formed by the data points. Use a single, bold color to fill the area for visual clarity and impact.

Generated



Ground truth



ID = 69
Score vis = 60
Score task = 95
Score human = 100

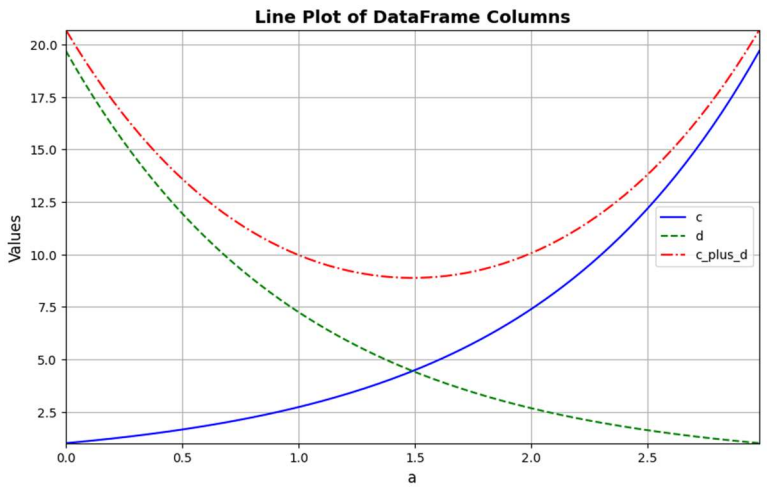
Task:

Create a line plot visualizing three columns from the DataFrame. Each line should represent a different data series with unique line styles. Additionally, customize the plot with titles, labels for the x and y axes, a legend to distinguish between the series, and specific location settings for elements like the legend to enhance plot readability.

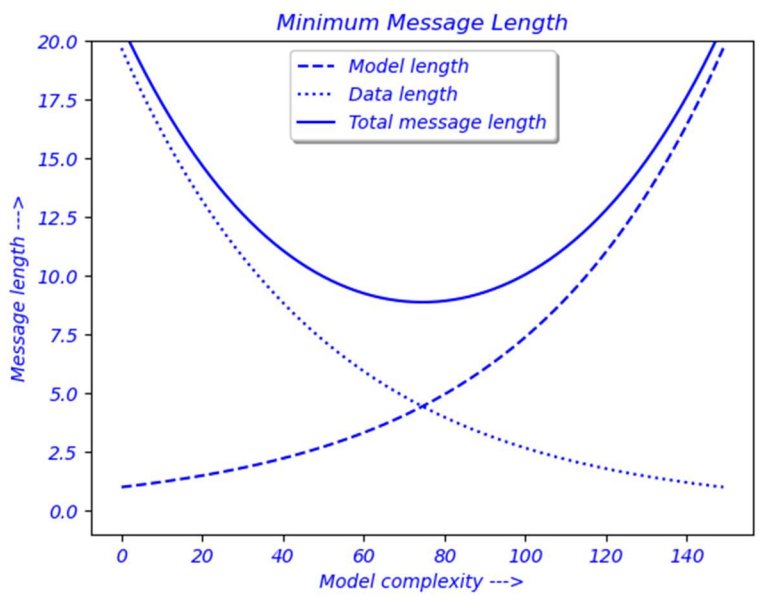
Style:

Define distinct line styles for each data series to ensure visual distinction. Apply axes limits to focus the view on relevant data ranges. Enhance the readability by modifying text elements such as axis labels and titles to follow specific stylistic attributes like font styles. Additional graphical elements such as colors and legend configurations should be customized to improve visual clarity and aesthetics.

Generated



Ground truth

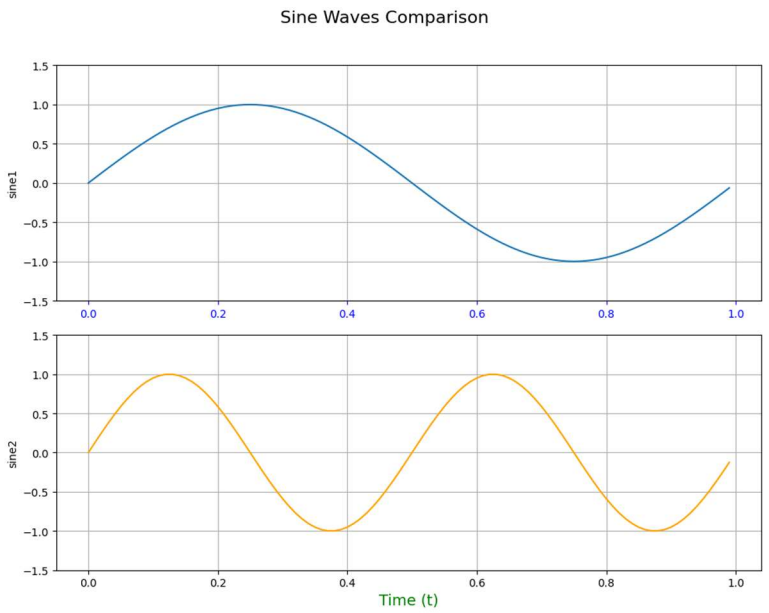


ID = 70
Score vis = 90
Score task = 95
Score human = 100

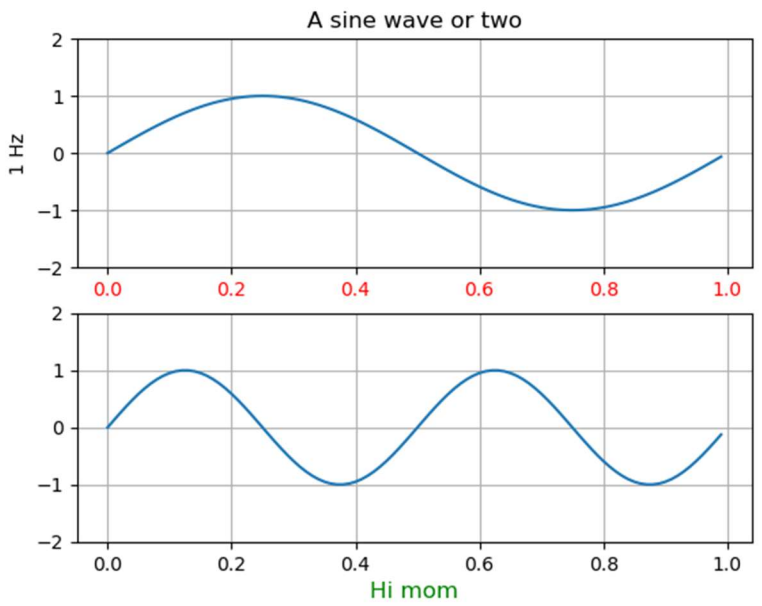
Task: Generate a vertical stack of two plots. The upper subplot should display the 'sine1' values against 't', including grid lines for better visualization and appropriate y-axis limits and labels to constrain and label the sine wave data. The lower subplot should perform a similar rendering for 'sine2' values versus 't', also with grid lines, y-axis limits, and detailed axis labeling including customized text properties.

Style: Enhance the plots with grid lines for easier data reading. Customize the x-axis tick labels of the upper plot with a specific color, and modify the x-axis label of the lower plot using another distinct color and adjusted font size for emphasis and clarity. Both plots should maintain the same amplitude limits for direct comparison. Ensure the entire figure has a coherent title that ties the subplots together.

Generated



Ground truth

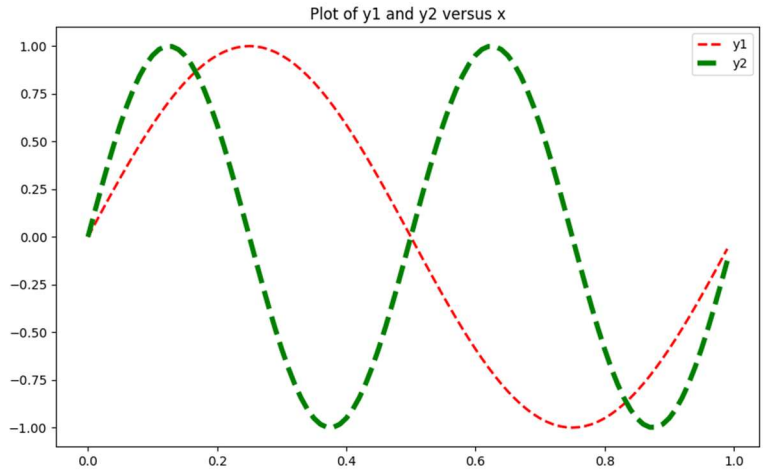


ID = 71
Score vis = 90
Score task = 100
Score human = 100

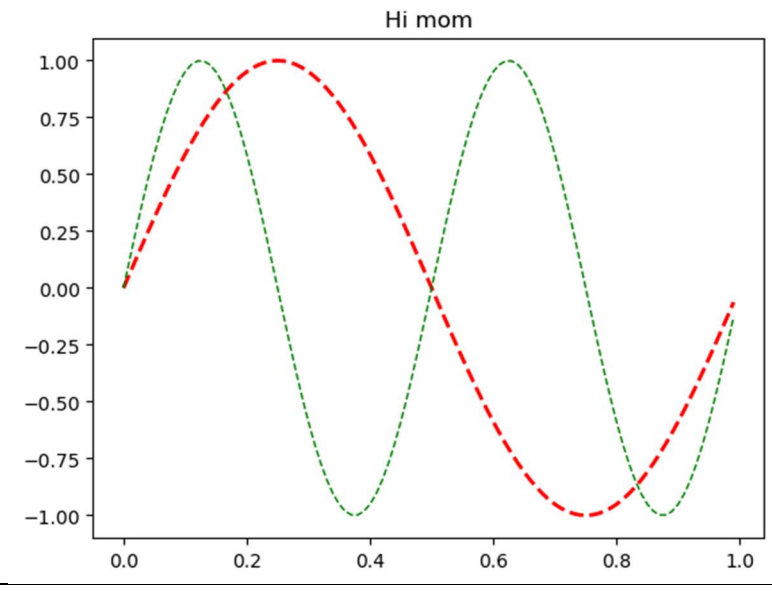
Task: Create a plot that includes two lines representing 'y1' and 'y2' versus 'x'. The two lines should be distinctly styled to differentiate them easily. Attach a title to the graph for identification or descriptive purposes.

Style: he lines with different colors and line patterns to visually segregate them - one line should be dashed and the other should also be dashed but with a different thickness. Color one line red and the other green, adjust line widths to enhance visibility. Set a clear and readable title on the top of the plot.

Generated



Ground truth

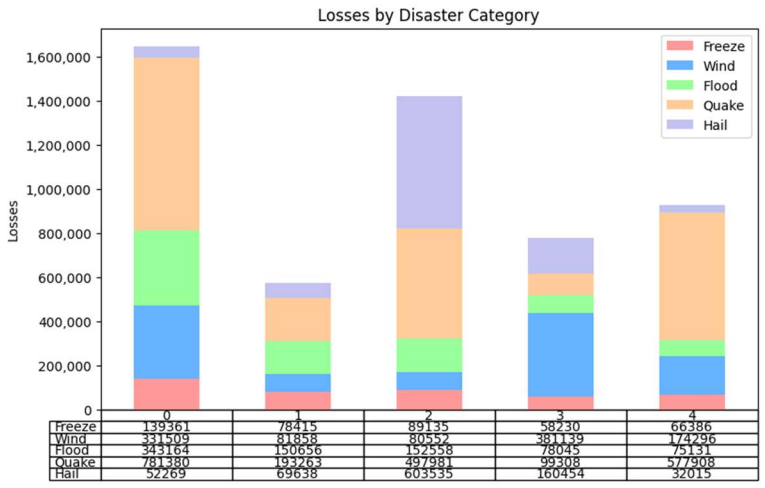


ID = 72
Score vis = 60
Score task = 95
Score human = 100

Task: The task is to create a stacked bar chart to represent the losses by disaster category. Each bar in the chart will represent a category with different segments corresponding to the types of disasters. Additionally, a table will be included below the bar chart displaying the numeric data for each category and disaster type for clearer reference.

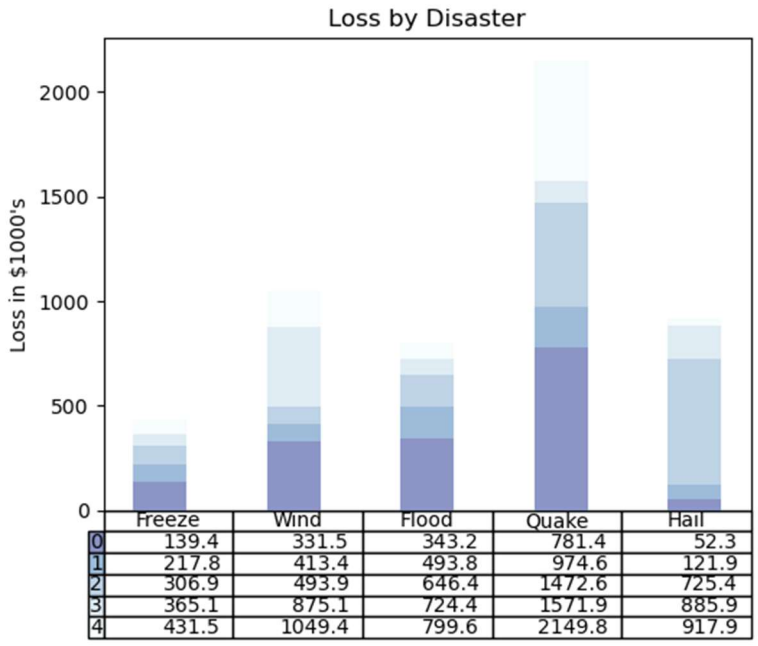
Style: For visual clarity, the bars will be colored using a graduated color palette. The chart will include a title, labels on the y-axis with formatted values, and an axis adjustment to accommodate the table layout at the bottom. The figure should avoid displaying x-tick marks to maintain focus on the table and bar visualization.

Generated



	0	1	2	3	4
Freeze	139361	78415	89135	58230	66386
Wind	331509	81858	80552	381139	174296
Flood	343164	150656	152558	78045	75131
Quake	781380	193263	497981	99308	577908
Hail	52269	69638	603535	160454	32015

Ground truth



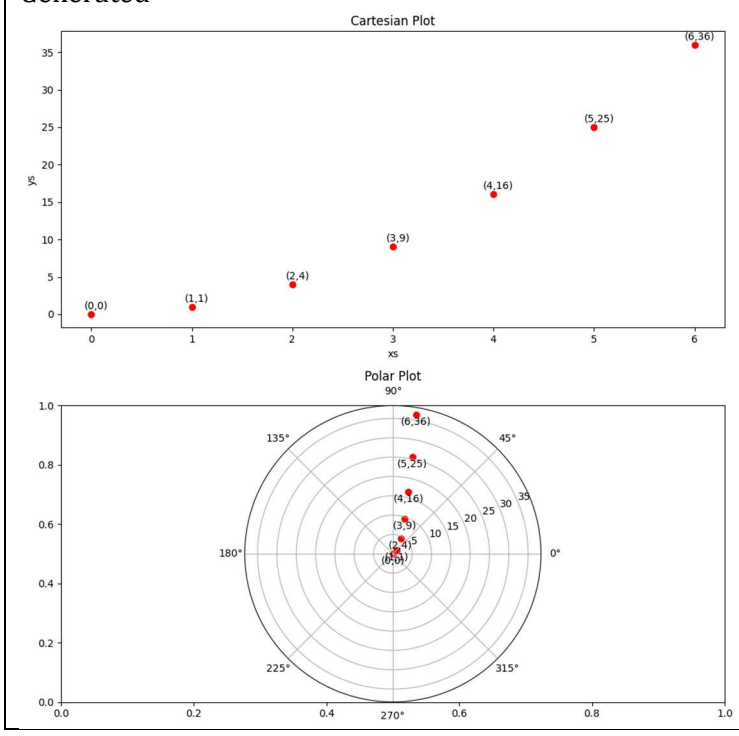
	Freeze	Wind	Flood	Quake	Hail
0	139.4	331.5	343.2	781.4	52.3
1	217.8	413.4	493.8	974.6	121.9
2	306.9	493.9	646.4	1472.6	725.4
3	365.1	875.1	724.4	1571.9	885.9
4	431.5	1049.4	799.6	2149.8	917.9

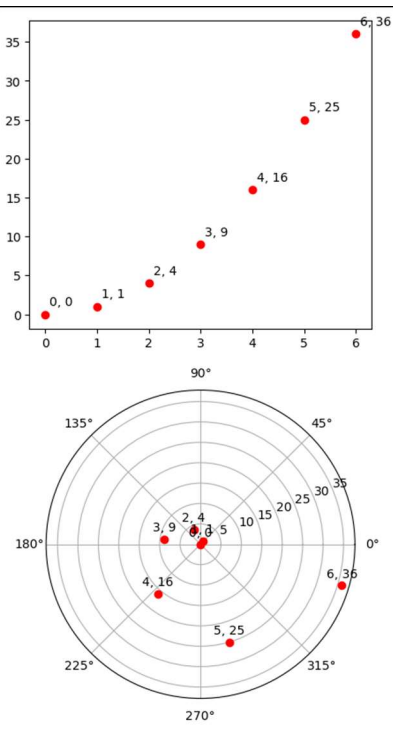
ID = 73
Score vis = 95
Score task = 95
Score human = 100

Task: The task will involve creating two types of plots in a single figure, stacked vertically. The first plot should be a standard Cartesian coordinate system plot displaying points defined by 'xs' and 'ys' values with annotations displaying these points' coordinates. The second plot should transform these points into a polar coordinate system, again plotting and annotating each point directly on the graph.

Style: For both plots, points should be marked in red. Annotations should be offset slightly from each point to avoid overlapping with the point itself, ensuring the annotations are clearly readable. In the Cartesian plot, this offset can be to the right and slightly above each point. In the polar plot, annotations should be adjusted to be centered horizontally and aligned just below each point, using a different offset method tailored to polar coordinates.

Generated



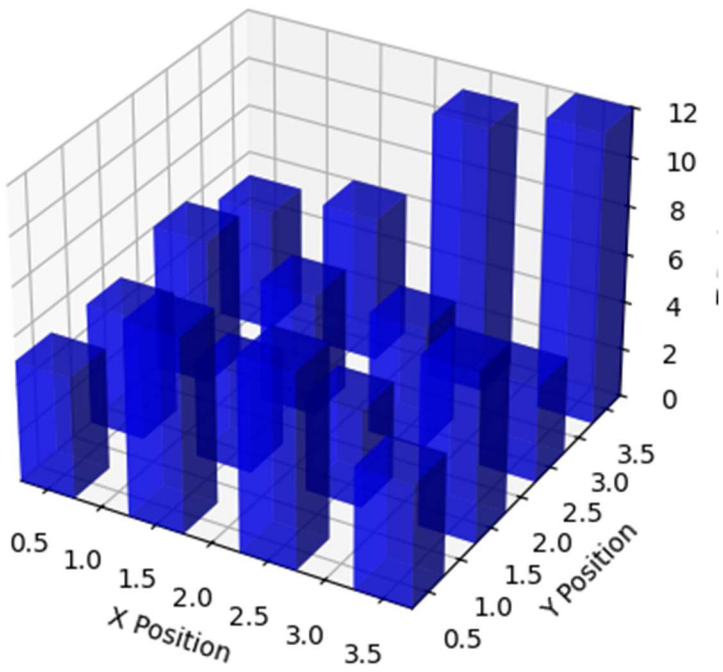
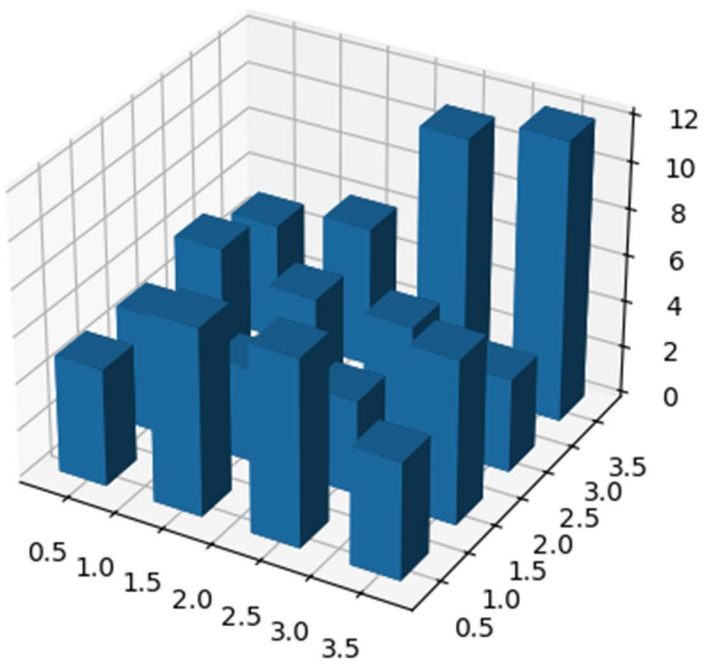


Ground truth

ID = 74
Score vis = 95
Score task = 85
Score human = 85

Task: Create a three-dimensional bar chart using the dataframe columns. Define a 3D subplot in a figure window. Plot bars at positional columns (xpos, ypos, zpos) with specified dimensions (dx, dy, dz), and sort them to optimize visibility.

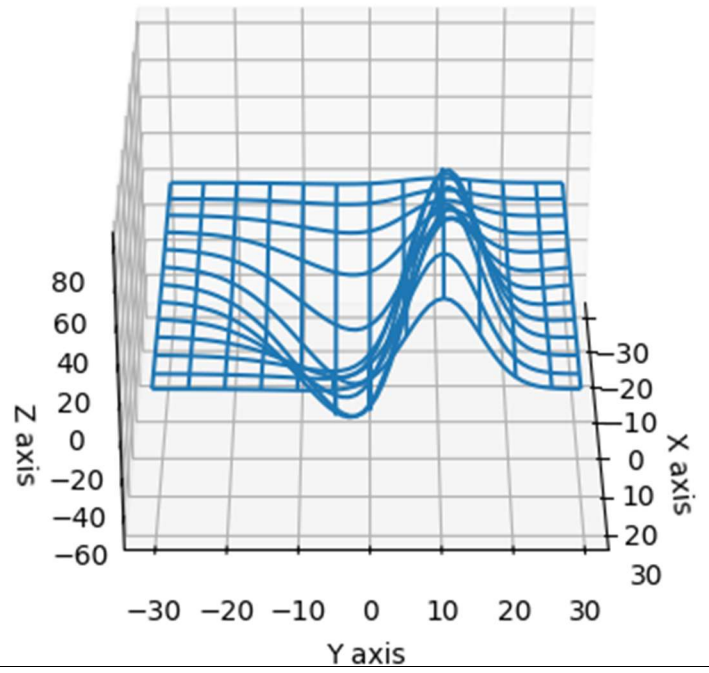
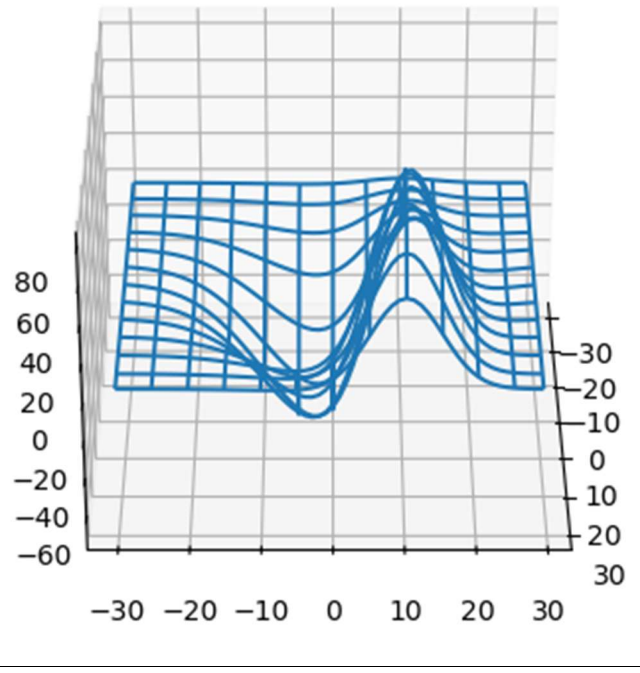
Style: he 3D bar chart to be visually appealing and engaging. Choose color shades and transparency levels to enhance clarity and depth perception, ensuring that the settings harmonize with the overall graphic presentation. Adjust labels, axes limits, and the perspective for a clear visualization of all dimensions.

Generated	Ground truth
	

ID = 75
Score vis = 100
Score task = 90
Score human = 100

Task: Construct a 3D wireframe plot from the reshaped arrays derived from the DataFrame's columns, representing spatial coordinates. The plot must clearly represent the variations and structure of the data in three dimensions. Include dynamic features in the plotting such as rotating the view angle during display for better interpretation of the structure in 3D space.

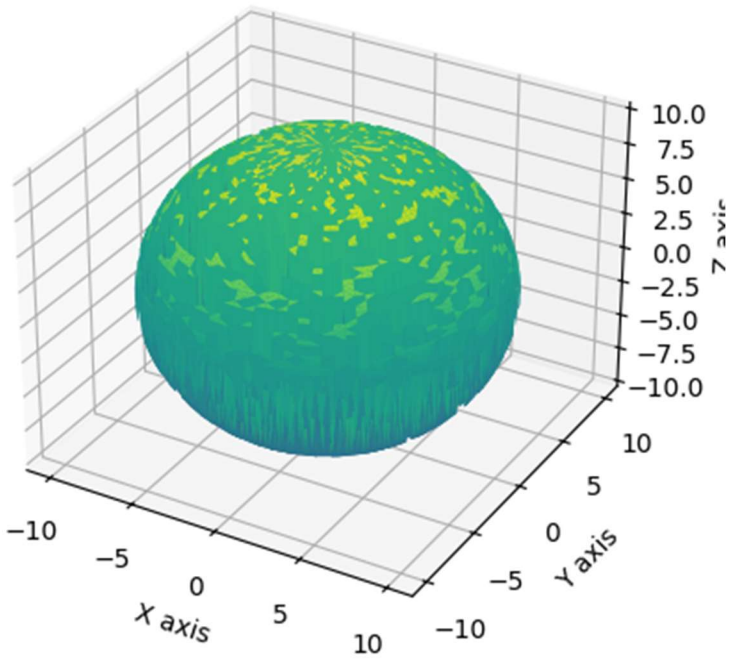
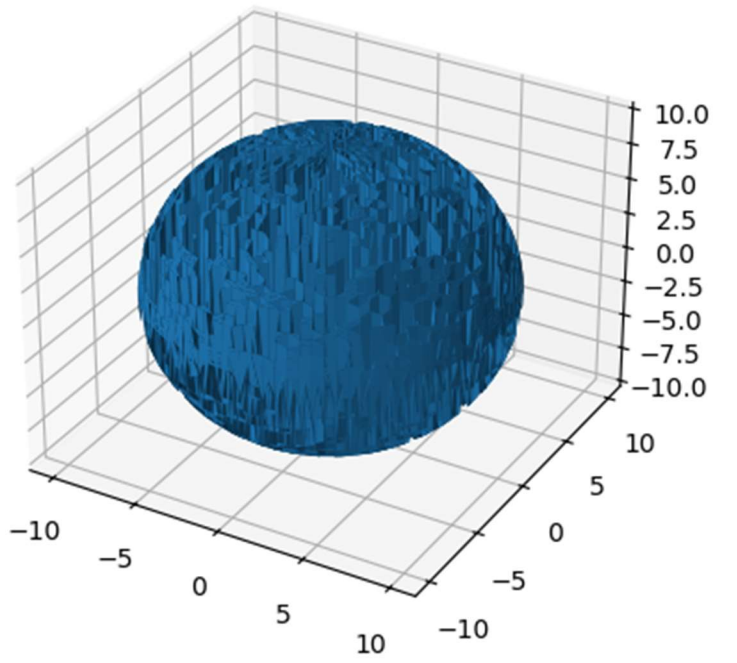
Style: The 3D plot should use a wireframe format to distinctly showcase the grid lines illustrating the coordinates' intersections. Enhance visual clarity by adjusting the grid stride and maintaining a clean, straightforward style, favoring readability and comprehension of complex 3D structures. The axes should be properly labeled to indicate the range of each dimension.

Generated	Ground truth
	

ID = 76
Score vis = 90
Score task = 90
Score human = 100

Task: The plot to be generated should be a three-dimensional visualization capturing the spatial distribution of data points described in the dataframe. Implement a 3D triangular surface plot where the plot surface will represent variations across three axes ('x', 'y', 'z') derived from the dataframe values, showing the topology or the geometrical shape implied by the data points' relations.

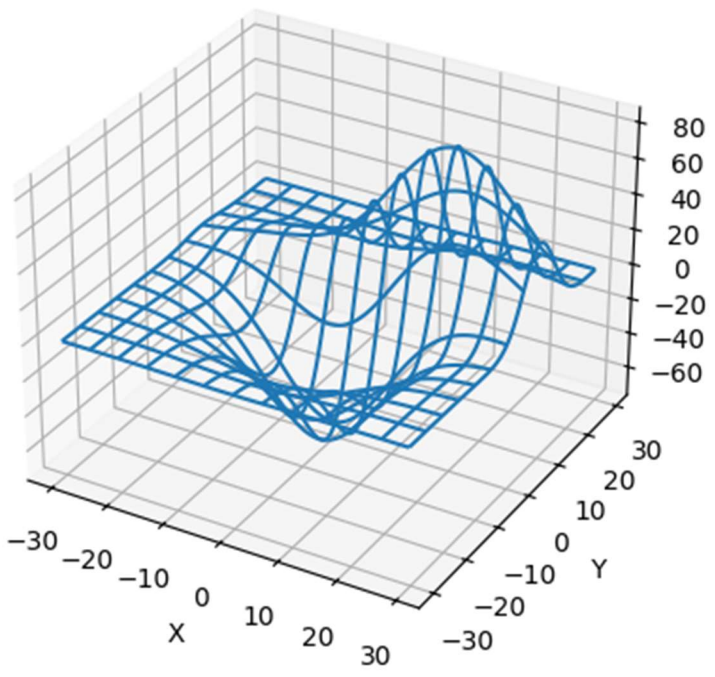
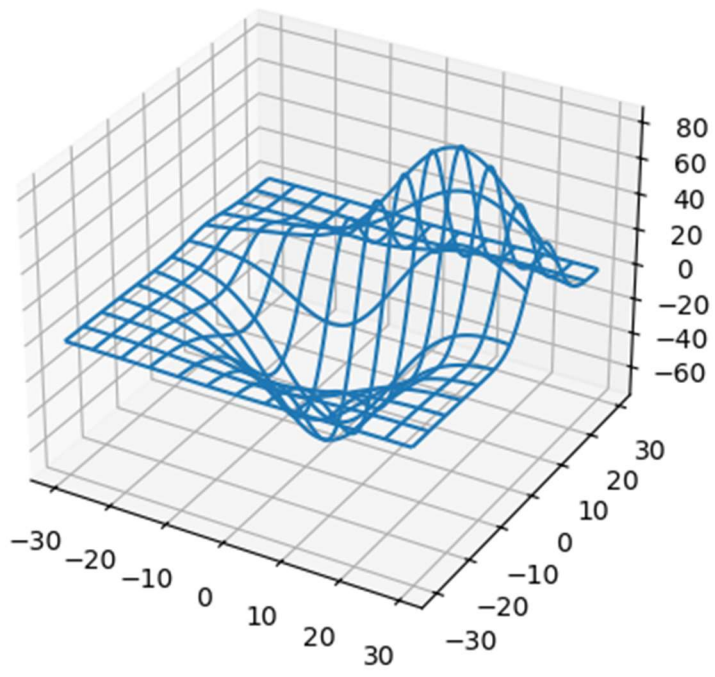
Style: The plot should be styled to highlight its three-dimensional aspects effectively. Use appropriate grid-lines to enhance visibility of depth and dimensionality. The plotting style should utilize a solid fill color to better represent the surface, and lighting or shading techniques to enhance the visualization of the surface's topology or contours.

Generated	Ground truth
	

ID = 77
Score vis = 100
Score task = 90
Score human = 100

Task: Construct a three-dimensional wireframe plot using the reshaped data matrices. The plotted object will depict relationships in spatial data across the X, Y, and Z dimensions. The plot should portray the geometric surface or shape represented by the matrix-valued coordinates, and the graphic should provide a clear and detailed view of the underlying structure.

Style: The graphical representation should be styled as a wireframe, with specified intervals (strides) between the lines to avoid an overcrowded visual. This refined spacing allows for an intuitive, aesthetic, and clear visualization of the three-dimensional surface contours and gradients, while ensuring the plot remains structured and easy to interpret.

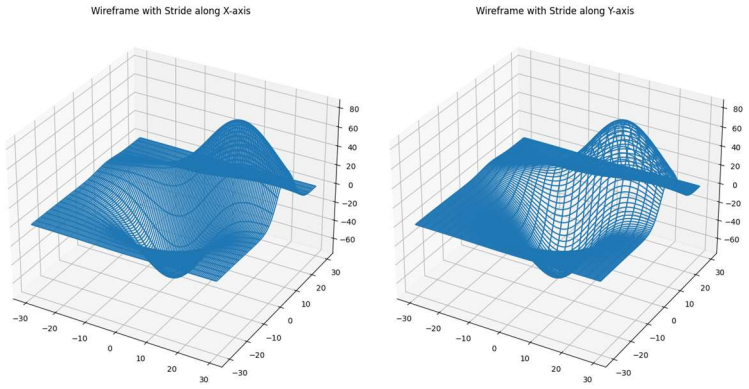
Generated	Ground truth
	

ID = 78
Score vis = 20
Score task = 95
Score human = 70

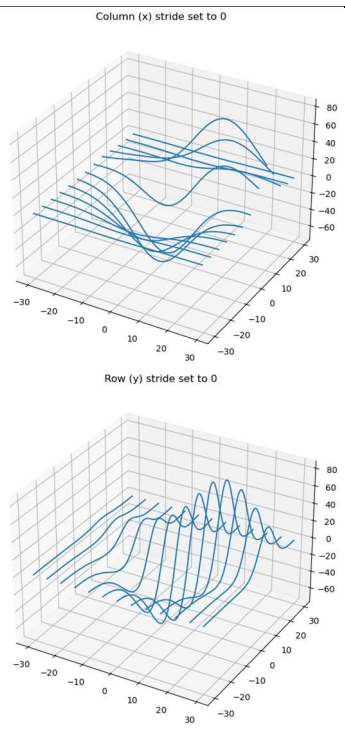
Task: Create two separate 3D plots within a single figure. Each plot visualizes data from the dataframe in a wireframe format. The first plot will emphasize variations by fixing the stride along the x-axis and freeing up the y-axis. The second plot inverts this approach by fixing the stride along the y-axis and freeing up the x-axis. Each subplot should have a title reflecting their respective striding technique.

Style: Configure the plots to have clearly readable titles that describe the striding configuration. Choose a layout that ensures neither plot overlaps the other, maintaining good visibility and differentiation between the two plots. Adjust aesthetic elements such as figure size and layout to enhance the visual appeal and clarity of the plot presentation.

Generated



Ground truth

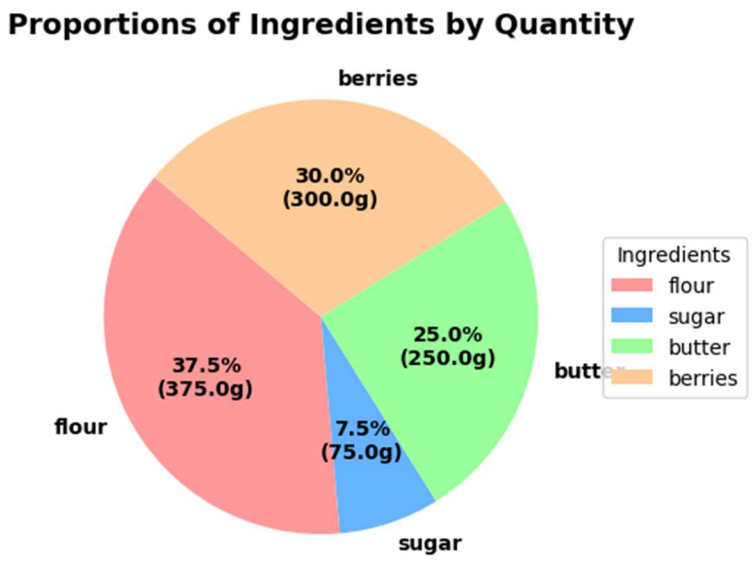


ID = 79
Score vis = 90
Score task = 95
Score human = 100

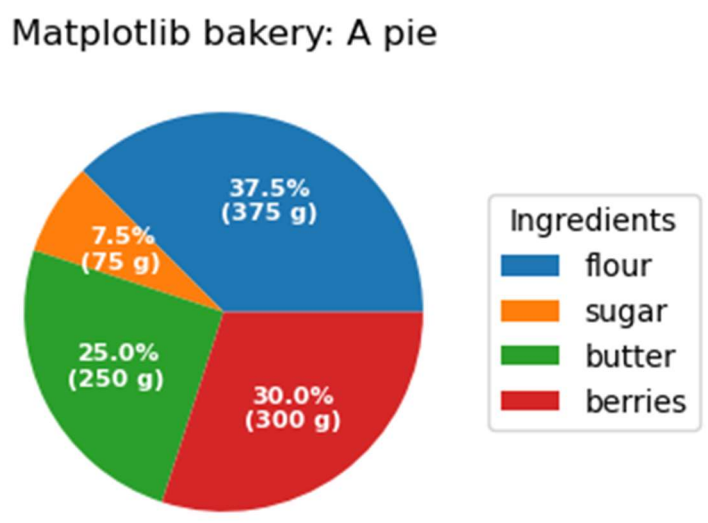
Task: Create a pie chart representing the proportions of various ingredients based on their quantities. Each slice of the pie should be labeled with both the percentage and the gram amount. Additionally, set up annotations so that each slice's data is easily readable for clarity.

Style: Customize the plot with a title and ensure readability by adjusting text properties such as size and weight. Place a legend outside the pie chart, aligning it to the left but centered vertically. Use different colors for each ingredient to differentiate them clearly in the chart.

Generated



Ground truth

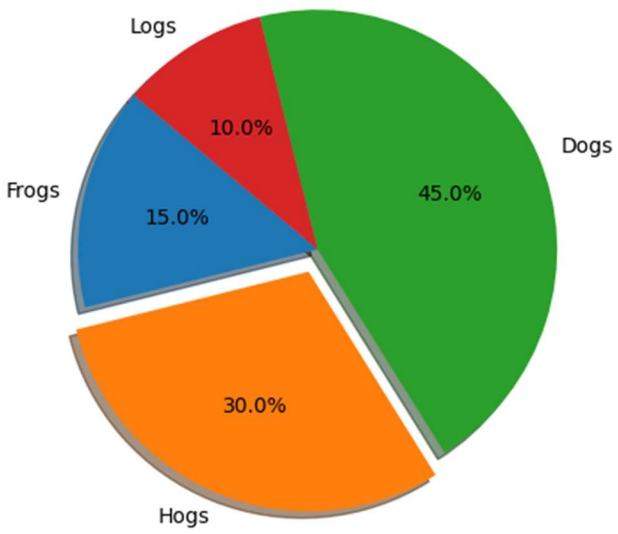


ID = 80
Score vis = 100
Score task = 90
Score human = 100

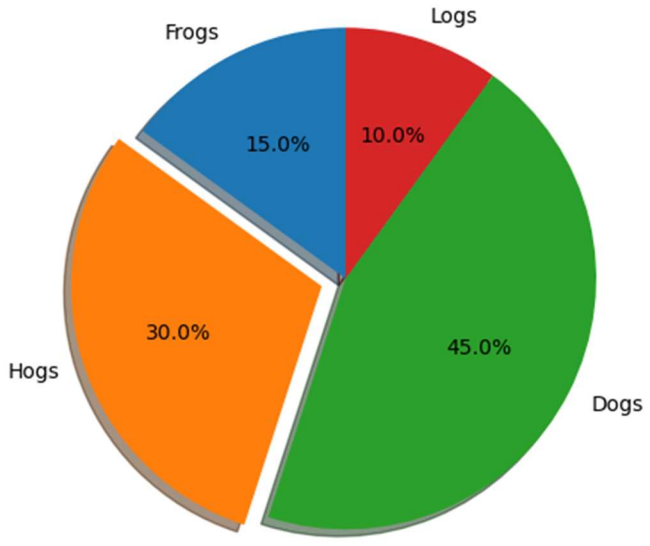
Task: Create a pie chart that visually represents the proportions of different categories provided in the DataFrame. Each slice should be labeled and its size should reflect the corresponding value in the 'sizes' column. Use the 'explode' values to slightly detach the slices for better visibility. Display percentage values within each slice of the pie for a clear quantitative understanding.

Style: The pie chart should be displayed as a perfect circle, ensuring equal aspect ratio. Additional enhancements include a shadow effect for depth perception and starting at a specific angle to orient the segments effectively. Labels and percentages should be clearly visible against a colorful background representing each category distinctively.

Generated



Ground truth

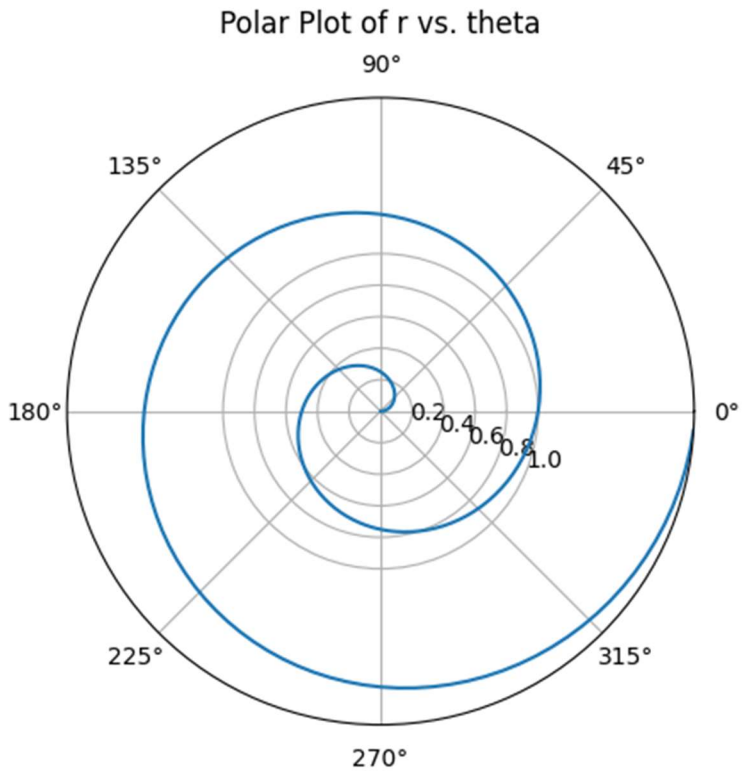


ID = 81
Score vis = 90
Score task = 90
Score human = 90

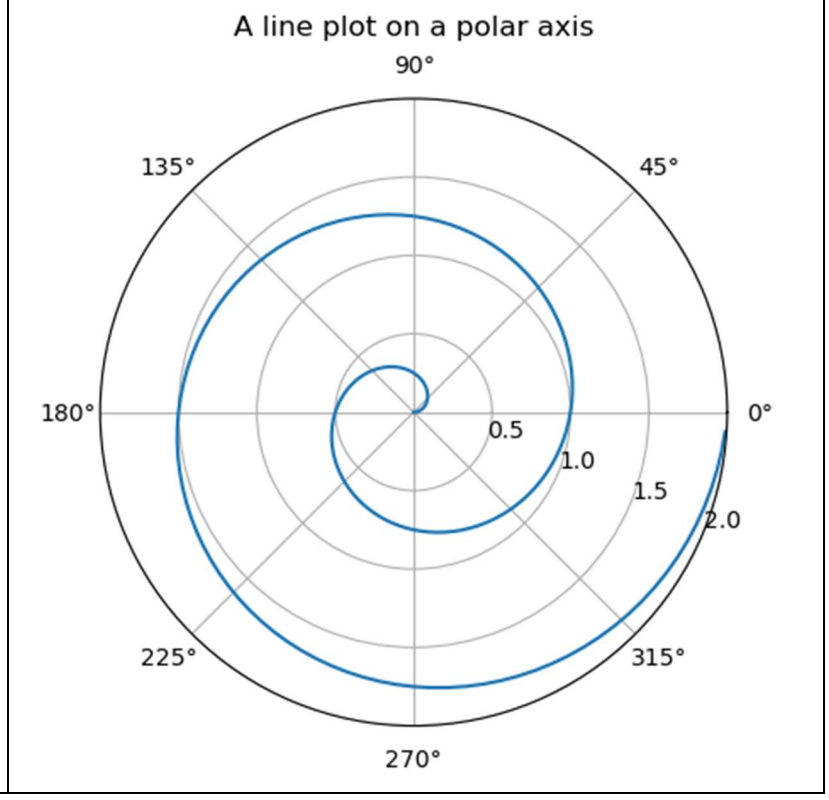
Task: Construct a polar plot where the line connects points defined by the 'theta' and 'r' data from the dataframe. Configure radial limits and ticks to enhance the readability and presentation of the plot. Set the position of radial labels to improve clarity and incorporate grid lines to aid in data interpretation.

Style: Customize the plot's title positioning to optimize space usage and aesthetic appeal. Employ a clean and structured layout for the visualization, ensuring that numeric labels are well-positioned and the overall appearance is informative and visually pleasing.

Generated



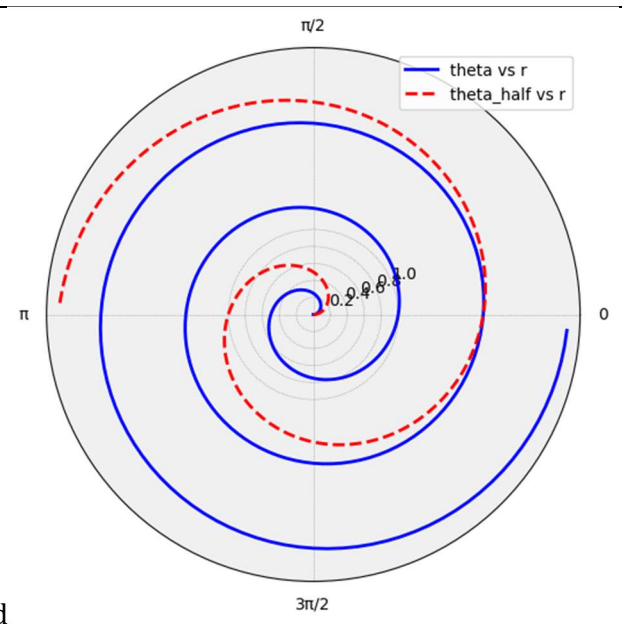
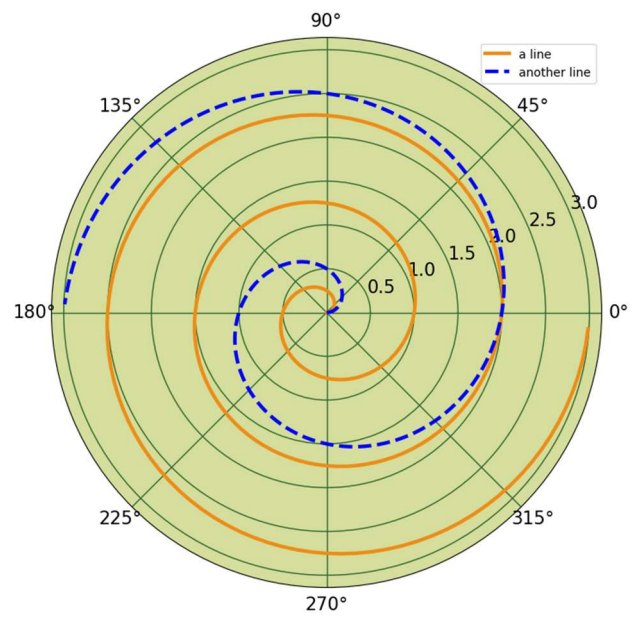
Ground truth



ID = 82
Score vis = 90
Score task = 95
Score human = 85

Task:
The task involves creating a polar plot from the DataFrame. The plot should incorporate two lines:
- One line representing the relationship between 'theta' and 'r'.
- Another line representing the relationship between 'theta_half' and 'r', to visually compare how 'r' changes with 'theta' and its half value.
Both lines should be distinctly styled and labeled appropriately within the plot.

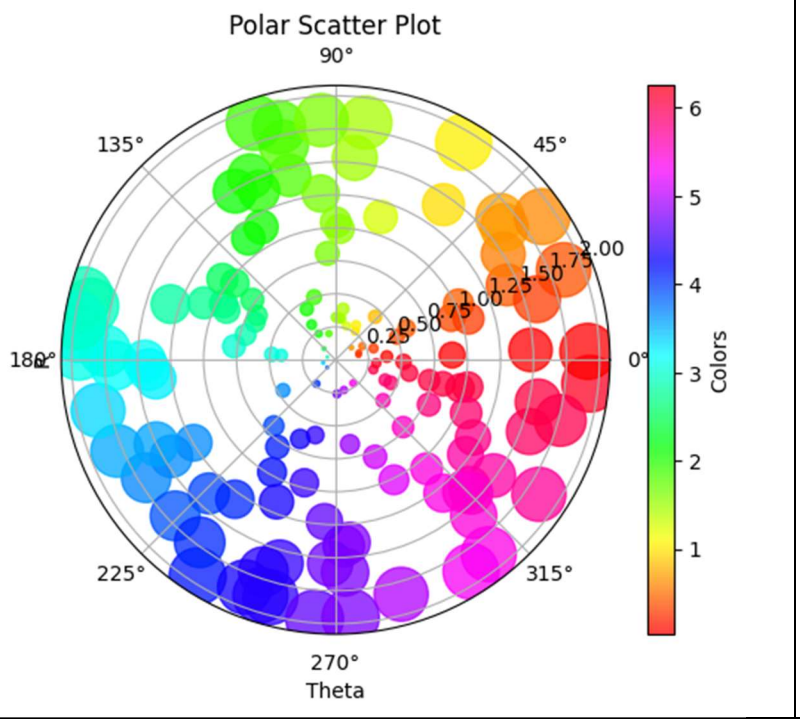
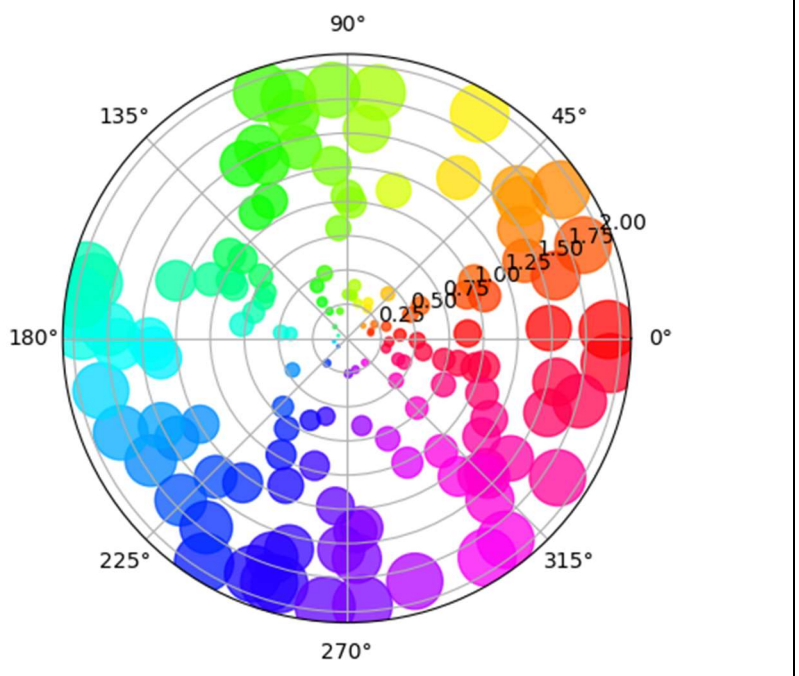
Style:
The plot should have a styled background specific for polar graphs, grid lines with a distinct color and style, and customized tick labels to enhance readability. Line styles should differ between the two series plotted (e.g., solid vs dashed lines), with colors and line widths that ensure they are distinguishable from each other. A legend is necessary to identify the plotted series. The overall size and aspect ratio of the plot should be set to clearly display all data without clutter.

<table>
<tr>
<td>Generated
 
</td>
<td>Ground truth
 
</td>
</tr>
</table>

ID = 83
Score vis = 95
Score task = 95
Score human = 100

Task: Create a polar scatter plot. Use the 'theta' values for the angular coordinates and 'r' values for the radial distance. The size of each scatter point is determined by the ‘area’ values. The color of each point is based on the ‘colors’ column, using a predetermined colormap to encode numeric values into colors. Adjust the transparency of the points to ensure readability where points overlap.

Style: Apply a colormap with a hue-saturation-value (HSV) color model to represent different values in a visually distinct manner. Set the transparency of each scatter point to 75% to balance visibility through overlapping points. Customize the plot to include gridlines and labels for various angles and radii to enhance readability and interpretability of the plot.

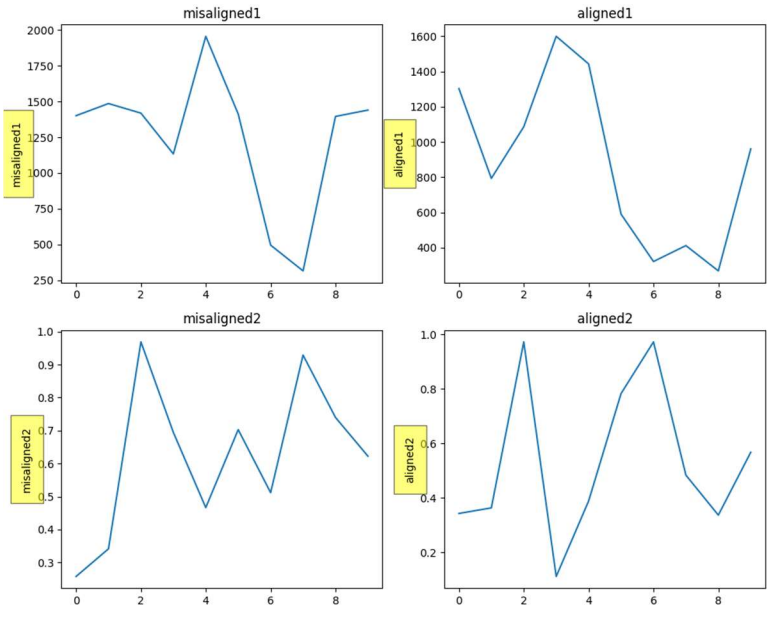
Generated	Ground truth
	

ID = 84
Score vis = 90
Score task = 95
Score human = 100

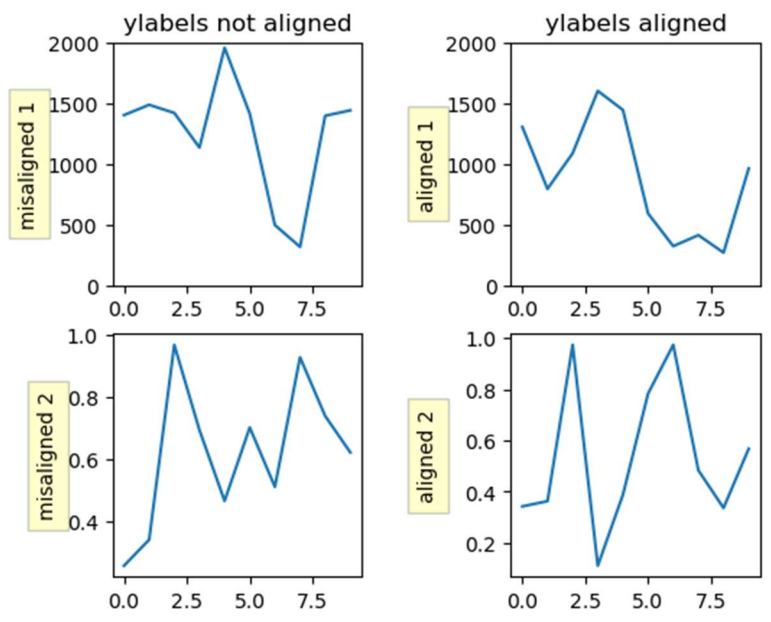
Task: The code should generate a grid of four plots (2 by 2). The top row should contain the plots 'misaligned1' and 'aligned1', each with their own x and y-axes. The bottom row should contain 'misaligned2' and 'aligned2'. The 'misaligned' plots will have y-labels added without alignment adjustments, whereas 'aligned' plots will have y-labels that are aligned across the second column.

Style: Each ylabel should be enclosed in a yellow box with a specified padding and partial transparency. Both columns should have individual settings for y-axis limits where applicable and overall improvement in spacing and alignment of y-labels on the second column against misaligned labels on the first column.

Generated



Ground truth

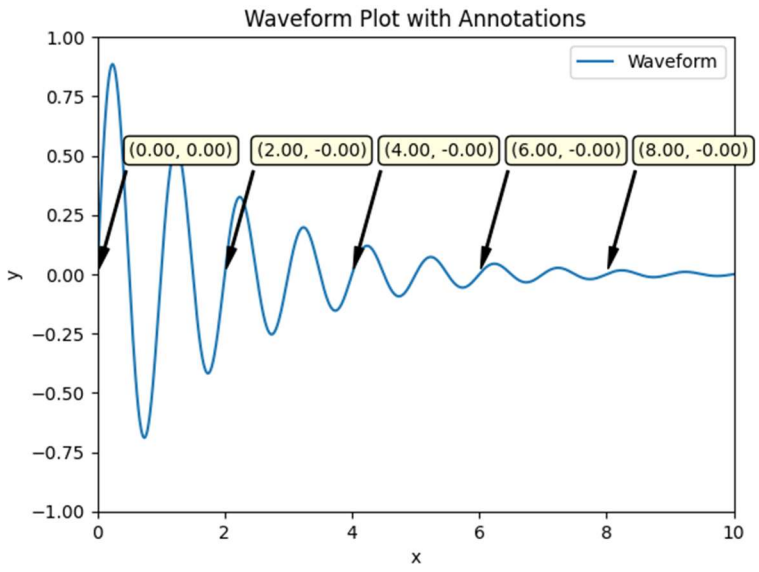


ID = 85
Score vis = 30
Score task = 95
Score human = 100

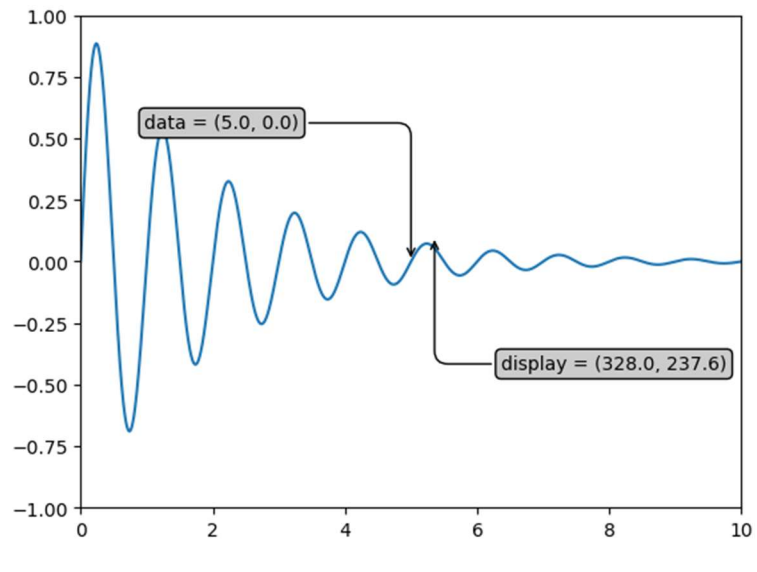
Task: description: Create a line plot using the 'x' column for the x-axis and the 'y' column for the y-axis, displaying a fluctuating waveform. The x-axis limits should be set from 0 to 10, and the y-axis limits should be set from -1 to 1. Include annotations on the plot to denote specific data points and their corresponding display pixel values on the plot. Use transformations to accurately position these annotations based on both data points and derived display coordinates.

Style: description: Customize the annotations with boxes and arrows for clear visibility. Set a rounded box style for textual annotations and configure arrow styles to clearly point from the text to the corresponding data point. Adjust text placement relative to the data points using offset coordinates. Utilize style settings to enhance readability and interpretability of the annotations, such as adjusting the background color of the annotation boxes and setting specific connection styles for arrows.

Generated



Ground truth

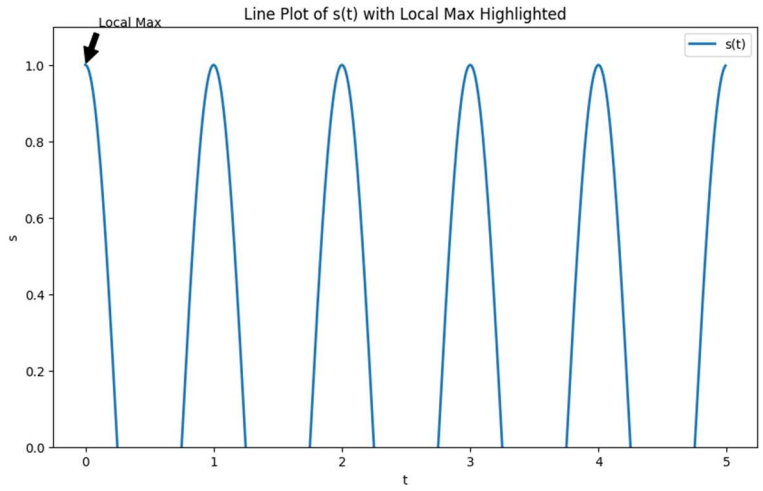


ID = 86
Score vis = 60
Score task = 95
Score human = 80

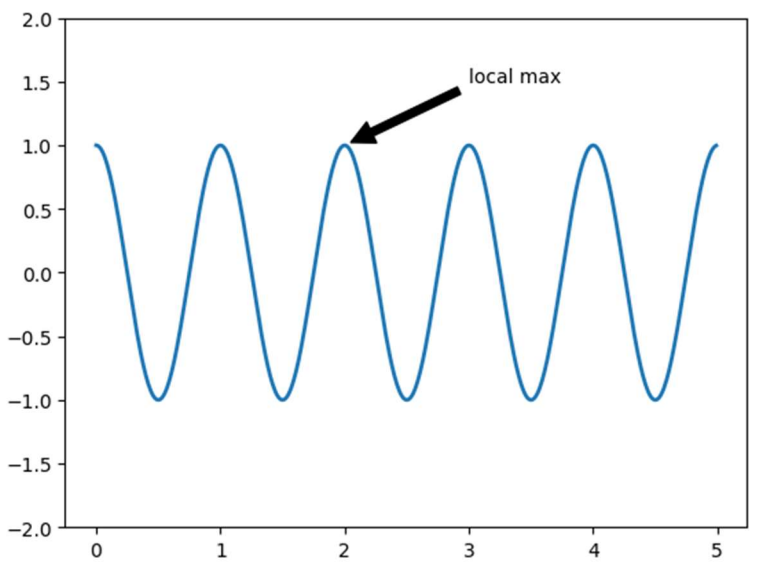
Task: Create a line plot using the two columns from the DataFrame. Column 't' will be set as the x-axis, and column 's' will be set as the y-axis. Highlight a specific point on this plot, referred to as 'local max', using an annotation with an arrow pointing to it.

Style: he line plot with a specific line width. Set the y-axis limits to enhance the visibility of the plot's features. The plot should contain labels for readability and employ arrows for annotations effectively to clarify key observations in the data.

Generated



Ground truth

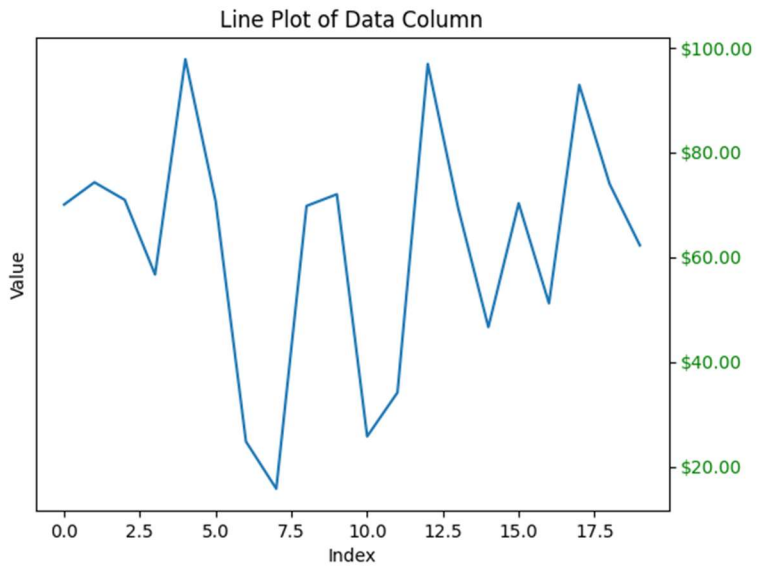


ID = 87
Score vis = 95
Score task = 90
Score human = 100

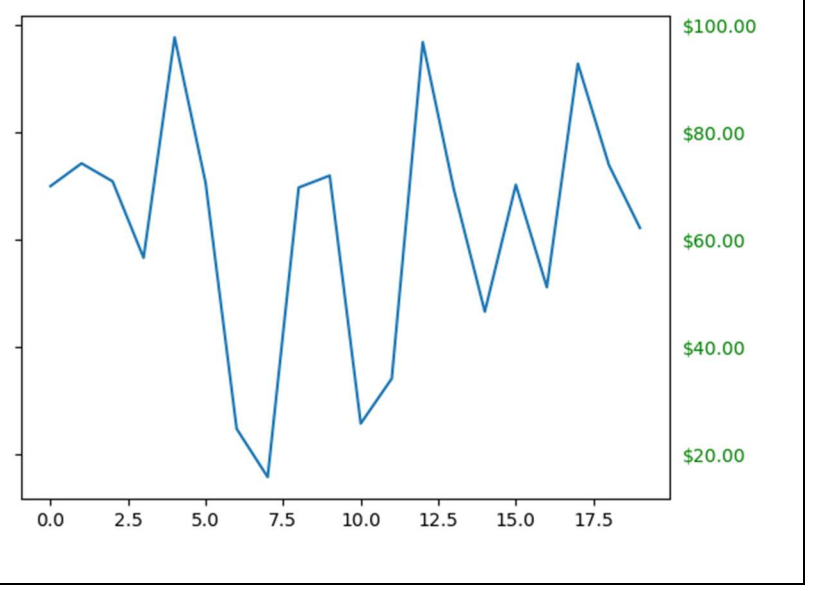
Task: Create a line plot using the data from the DataFrame's 'Data' column. Plotting should result in a visualization that reflects the trends and fluctuations in the data over its index.

Style: Format the y-axis to display values as monetary amounts in US dollars, ensuring each tick label on the y-axis shows values with two decimal places preceded by a dollar sign. Customize the y-axis tick labels to hide labels on the left and display labels on the right in green color. Adjust visualization settings for clearer understanding and display of data trends.

Generated



Ground truth

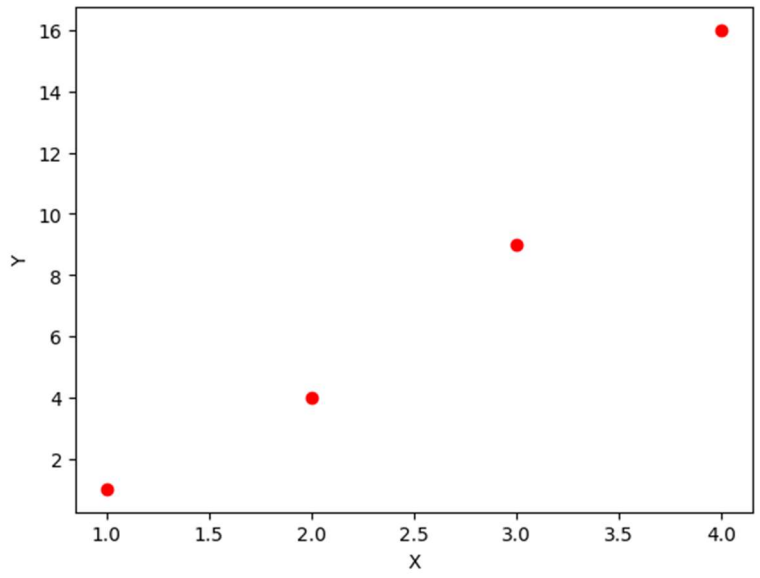


ID = 88
Score vis = 100
Score task = 100
Score human = 100

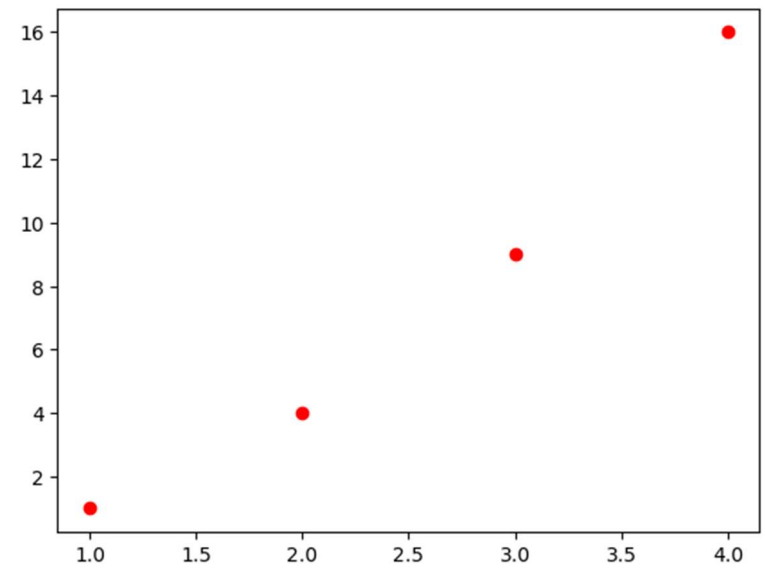
Task: Create a scatter plot of the dataframe using the two columns. Each column represents one axis: one for the horizontal axis (x-axis) and the other for the vertical axis (y-axis). The plot should depict individual data points.

Style: The data points should be represented as red circles on the plot. Emphasize the visibility and distinction of each point against the plot grid. The axes should be clearly labeled based on the dataframe's column names, and the range of each axis should automatically adjust to accommodate all data points.

Generated



Ground truth

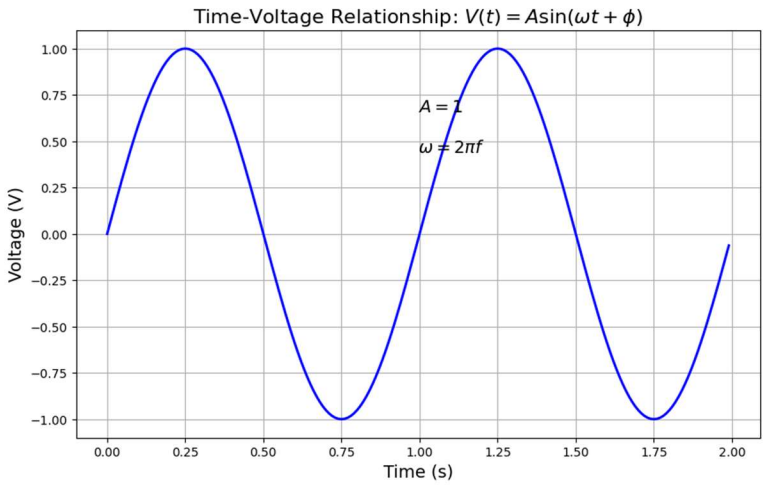


ID = 89
Score vis = 80
Score task = 95
Score human = 100

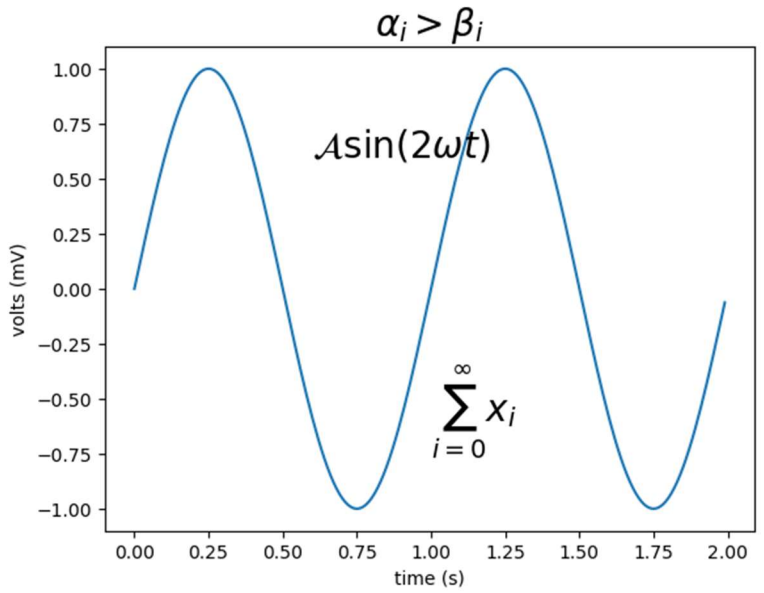
Task: Generate a plot using data from the DataFrame with time on the horizontal axis and voltage on the vertical axis. Include a title using mathematical notation, and superimpose two additional text elements containing mathematical expressions at specified positions on the plot. Label the axes to indicate the units of measurement for each.

Style: he plot by adjusting the font size of the title and text elements to enhance readability and aesthetic appeal. Choose a line style and width that clearly depicts the time-voltage relationship. Ensure that axis labels are appropriately detailed to reflect the units of measurement. Add grid marks or background style if necessary to improve the visual distinction of data points.

Generated



Ground truth

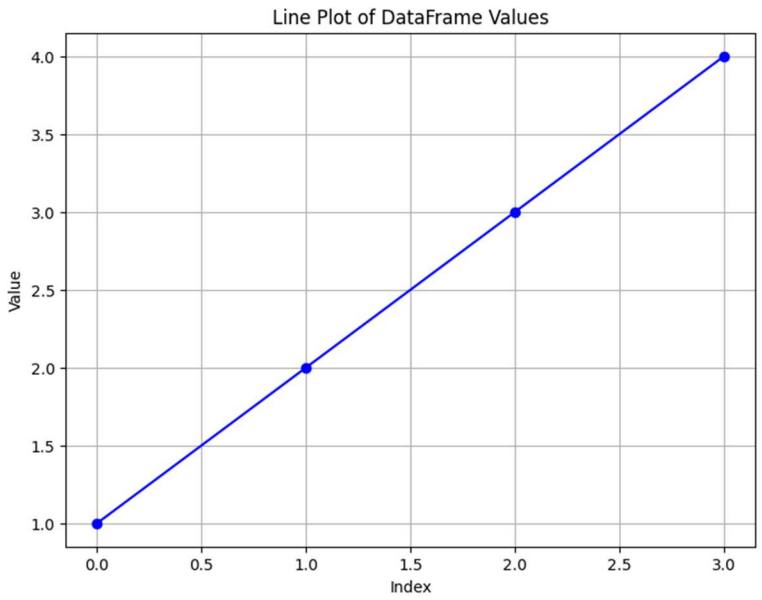


ID = 90
Score vis = 90
Score task = 100
Score human = 100

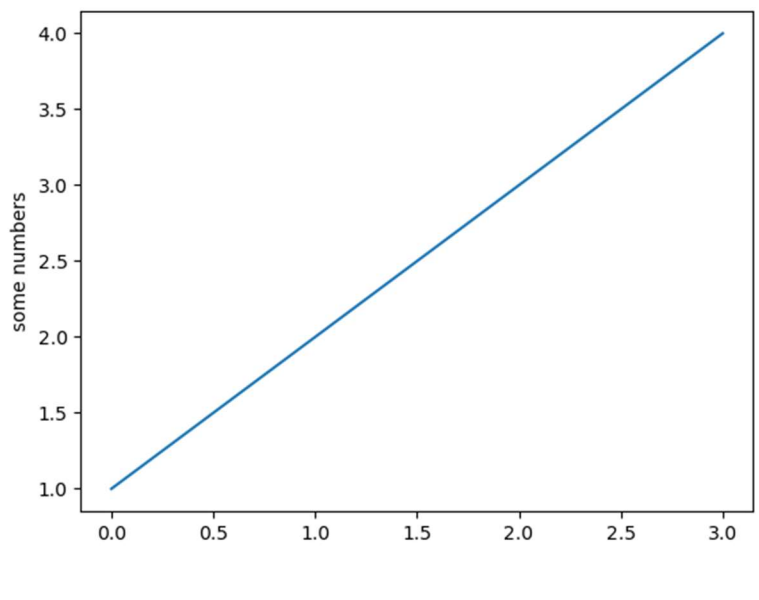
Task: description: Create a simple line plot to visualize the data in the DataFrame. Plot the integer values against their corresponding index positions on the x-axis. Include appropriate labels for the y-axis to represent the data.

Style: description: Style the plot with a clear and minimalist design. Use a solid blue line for the data points connecting them linearly. Ensure the axes are appropriately scaled to display all data points clearly. The plot should have labeled axes where the y-axis indicates the magnitude of the values presented in the DataFrame.

Generated



Ground truth

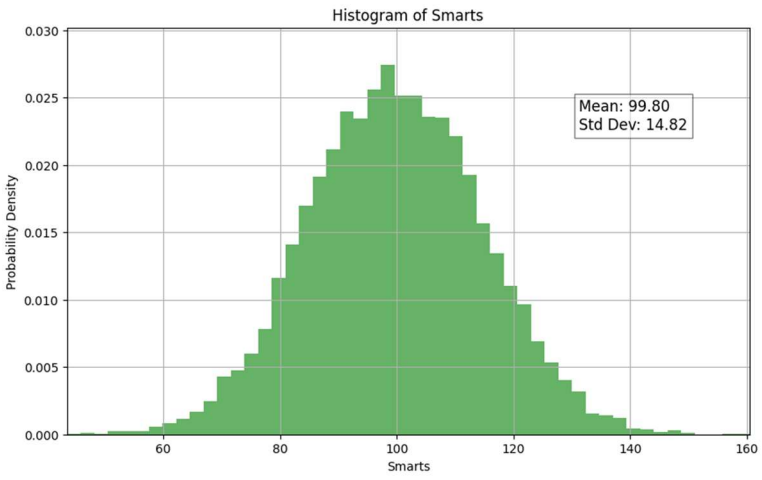


ID = 91
Score vis = 90
Score task = 100
Score human = 100

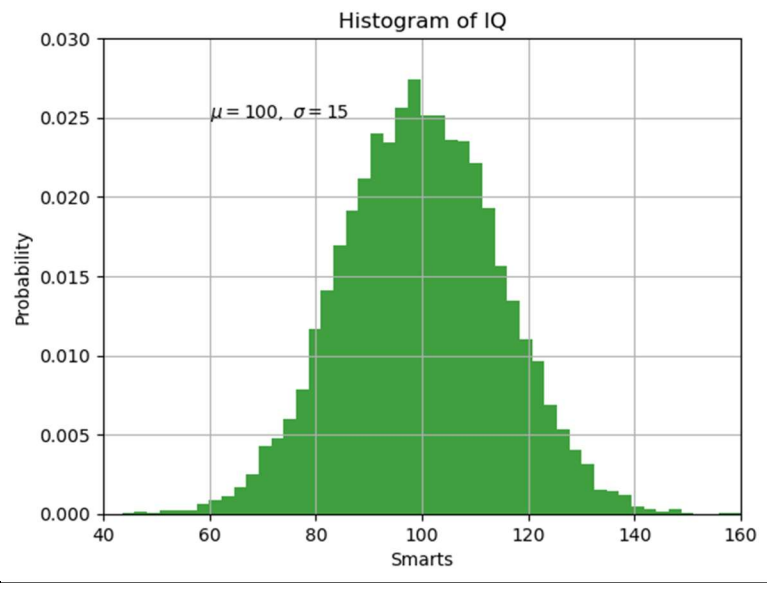
Task: Create a histogram based on the 'Smarts' column of the dataframe. Configure the histogram to divide the data into 50 bins, display the frequency as a probability density, and color the bars in green with a specified level of opacity. Add labels for both axes, a title for the graph, and an annotation within the plot area to indicate the mean and standard deviation. Set the limits for the x-axis and y-axis to frame the data effectively and enable grid lines for better readability.

Style: The style of the histogram should be clear and informative. Use a green color for the bars with a transparency setting to allow for overlay visibility. Clearly mark the axis labels and graph title with legible fonts. Annotations should be placed within the graph to provide statistical information at a glance. The x-axis and y-axis scales should be appropriately set to encompass all data points while maintaining a clean and uncluttered appearance. Grid lines should be subtle but useful for assessing the values represented in the histogram.

Generated



Ground truth

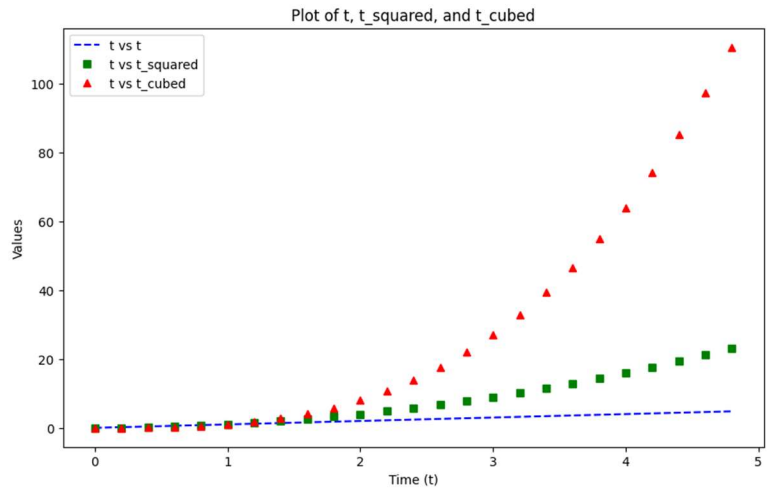


ID = 92
Score vis = 70
Score task = 100
Score human = 100

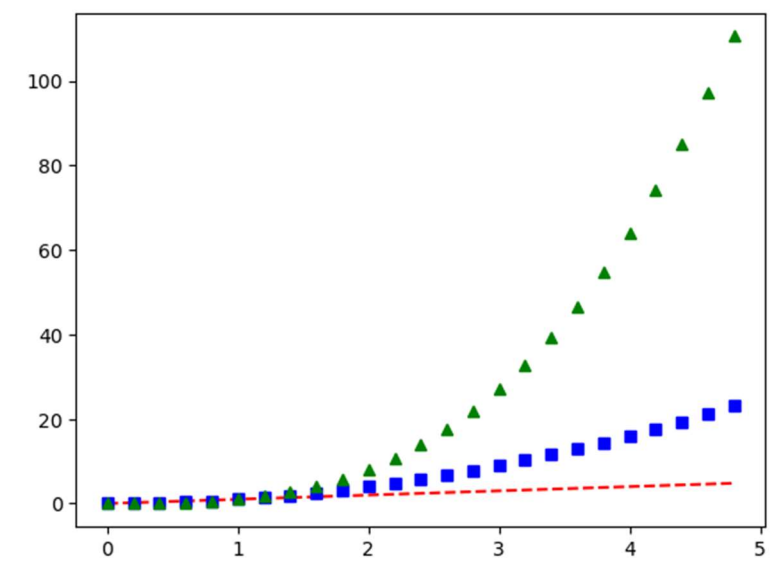
Task: aw three separate series on a single chart from the dataframe. The first series should be a simple line graph relating the time ('t') to itself with a specific line style. The second series should plot time against 't_squared' using a distinct marker. The third series should plot time against 't_cubed' with another unique marker. The aim is to differentiate the growth between these variables visually.

Style: he plot by defining specific markers and line types for each series to clearly distinguish them. The first series uses a dashed line, the second series uses a square marker for each data point, and the third uses a triangle marker. Choose different colors for each series to improve clarity and visual appeal. The axes should be labeled appropriately to reflect the quantities being displayed.

Generated



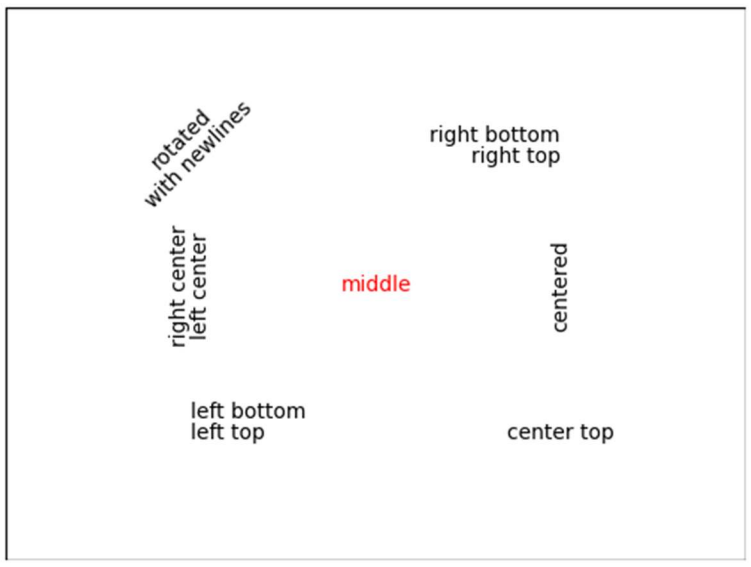
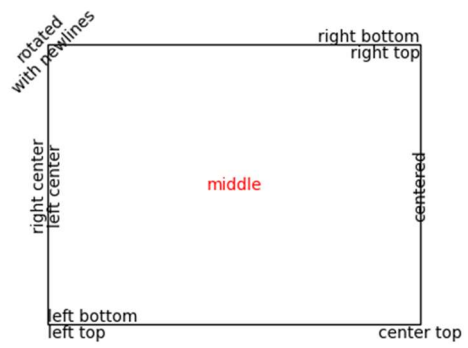
Ground truth



ID = 93
Score vis = 90
Score task = 100
Score human = 75

Task: description: The task involves generating a rectangular plot area with a specific rectangle drawn without fill. The rectangle's dimensions are defined relative to the figure's axes. Text elements, stored in the DataFrame, are plotted within this rectangle, positioned at coordinates defined by 'x' and 'y'. Each text element has custom properties for alignment, rotation, color, and font size applied to it, controlled by corresponding DataFrame columns.

Style: description: The plot background is transparent, with no axis visible. Text styling within the plot is controlled by data-derived attributes such as alignment (vertical and horizontal), font size, text rotation, and color, leading to a customized and informative graphical representation.

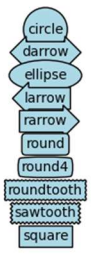
Generated	Ground truth
	

ID = 94
Score vis = 50
Score task = 60
Score human = 75

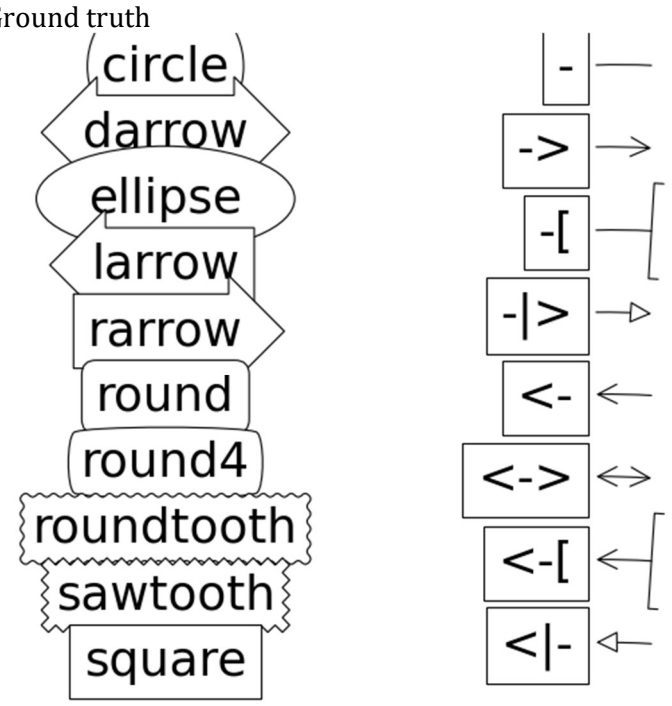
Task: The task involves creating two plots. The first plot will illustrate various box styles by positioning them vertically and aligning in the center. Each box will display a label with its respective style. The second plot will showcase arrows pointing to circles, both located based on their given position. Labels adjacent to arrows will indicate the arrow style.

Style: Customize figure size and resolution to enhance clarity and visual appeal. Text labels in the first plot will be placed within boxes tailored stylistically according to the dataframe's entries. In the second plot, arrows with distinct styles will point towards circles, visually representing the data pertaining to arrow types. Both plots will avoid displaying axes scales to maintain focus on the graphical elements, not on quantitative scales.

Generated



Ground truth

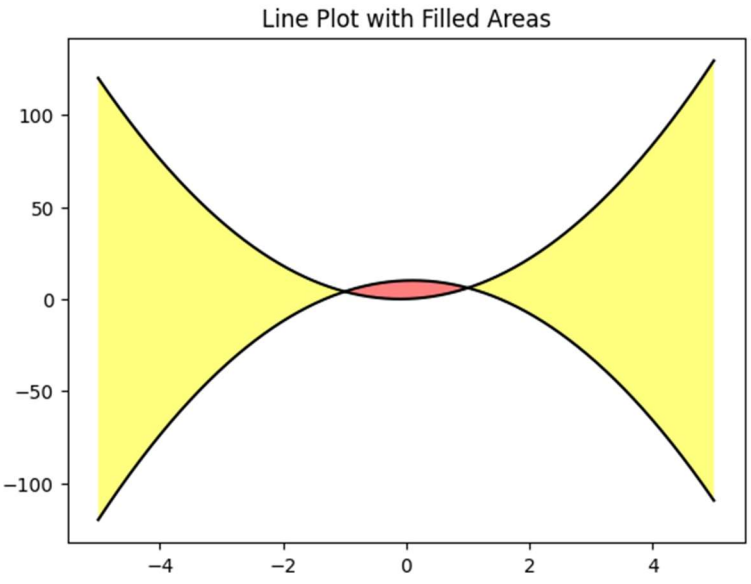


ID = 95
Score vis = 100
Score task = 95
Score human = 100

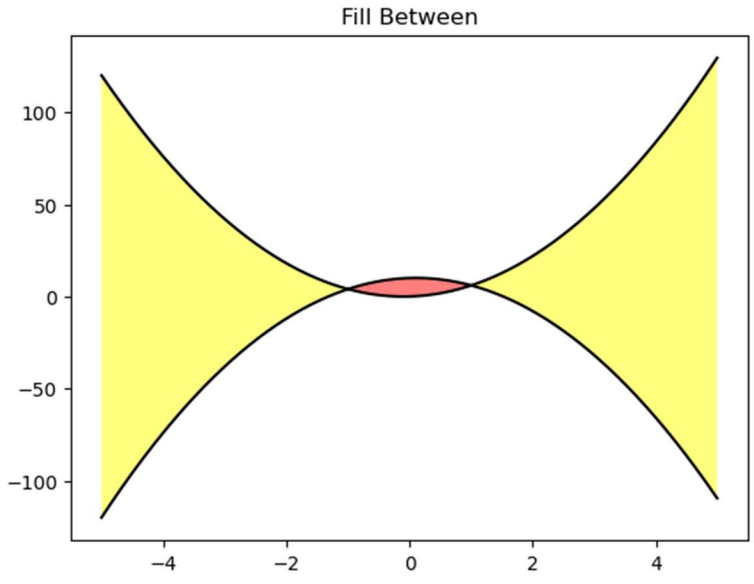
Task: The plot consists of two main components: line plots and filled areas. The line plots will display the values from 'y1' and 'y2' against 'x'. Additionally, the area between the 'y1' and 'y2' lines will be filled. This filling will vary in color depending on whether 'y2' is above 'y1' or not.

Style: The lines for 'y1' and 'y2' should be plotted in black color. Areas where the 'y2' values are greater than 'y1' will be filled with a yellow color, while areas where 'y2' is less than or equal to 'y1' will be filled with a red color. The filling will be translucent, allowing for the underlying grid or any other lines to be partially visible. The plot will have a title indicating its purpose or content.

Generated



Ground truth

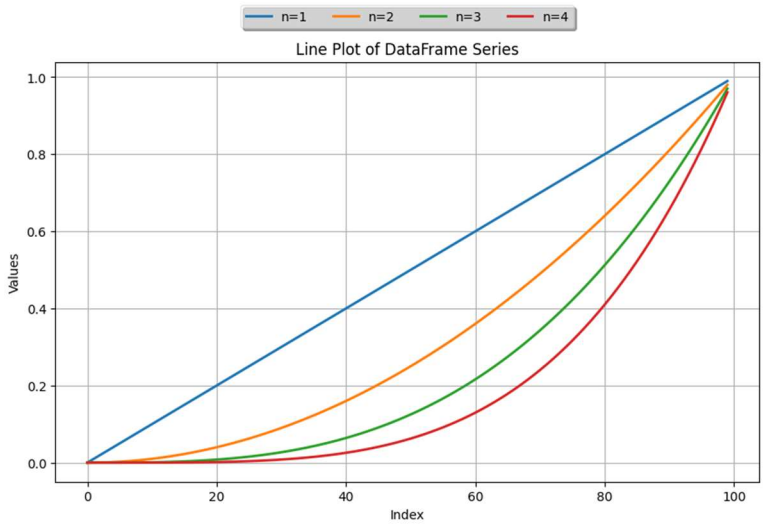


ID = 96
Score vis = 95
Score task = 90
Score human = 100

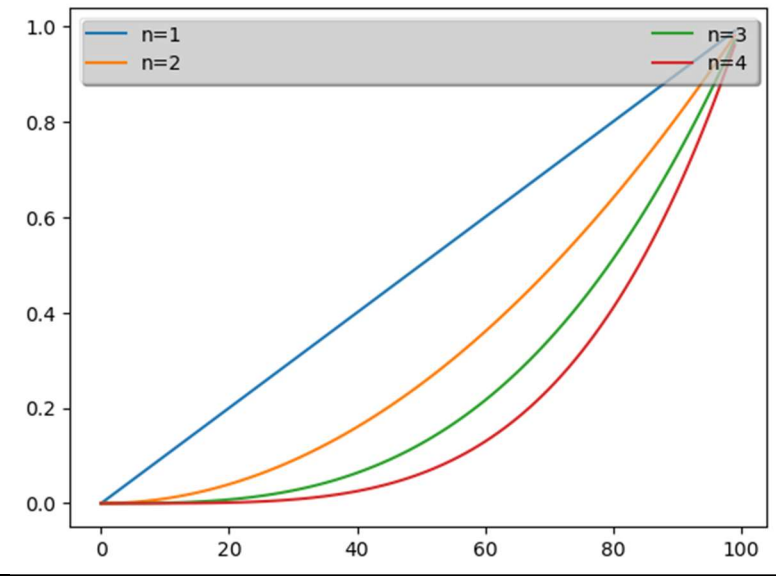
Task:
Create a line plot for each series in the dataframe on a common axis. Each series should be distinctly colored and included in the plot. Include a legend to differentiate among the plotted series, displaying the legend in an optimized position that accommodates all entries neatly without overlapping with data.

Style:
Enhance the visual appeal and readability of the plot by applying styles. The legend should be expanded across the top of the plot to include all series names and should feature a semi-transparent background. Additionally, add soft shadows to the legend to give the plot a more refined look. Adjust visual styling details such as line color and width to ensure clarity and aesthetics.

Generated



Ground truth

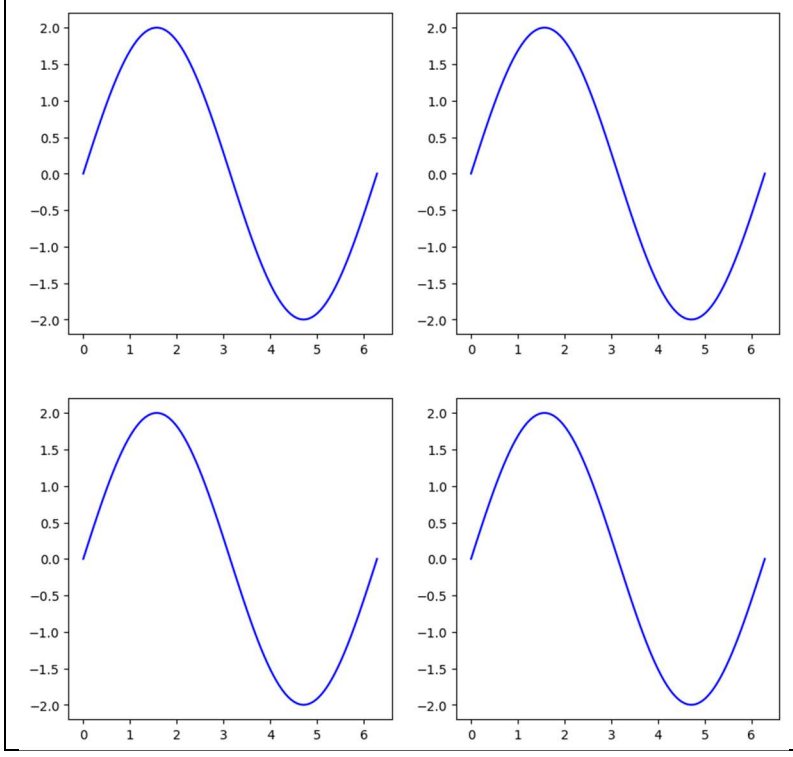


ID = 97
Score vis = 90
Score task = 20
Score human = 20

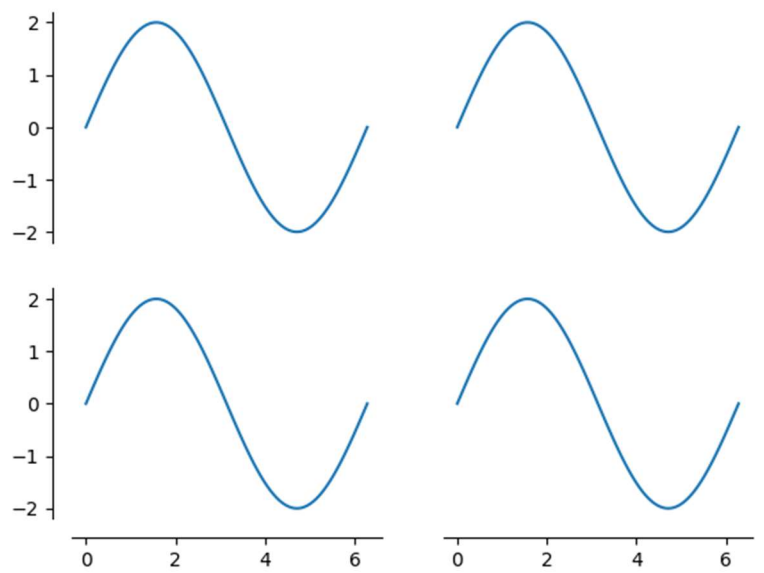
Task: Create a plot with four subplots arranged in a 2x2 grid. Each subplot should contain a line plot of 'y' versus 'x'. Customize each subplot's axes visibility and tick positions differently: the first with only the left spine visible, the second with none visible, the third with both the left and bottom spines visible, and the fourth with only the bottom spine visible.

Style: Adjust the visibility and positioning of the spines and ticks for each subplot according to the specified customization. Use outward positioning for visible spines and ensure that ticks are only shown where spines are visible. Style the line plot with a consistent color and line type across all subplots.

Generated



Ground truth

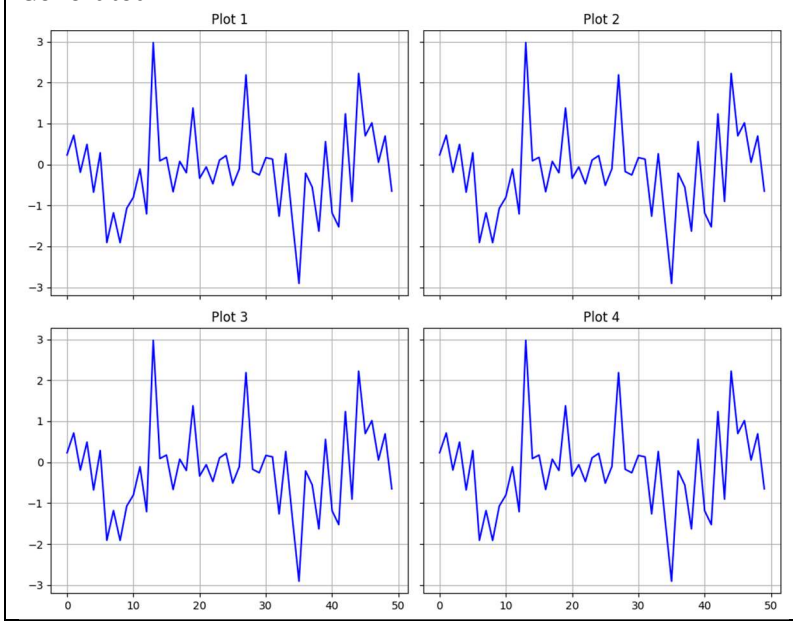


ID = 98
Score vis = 90
Score task = 100
Score human = 100

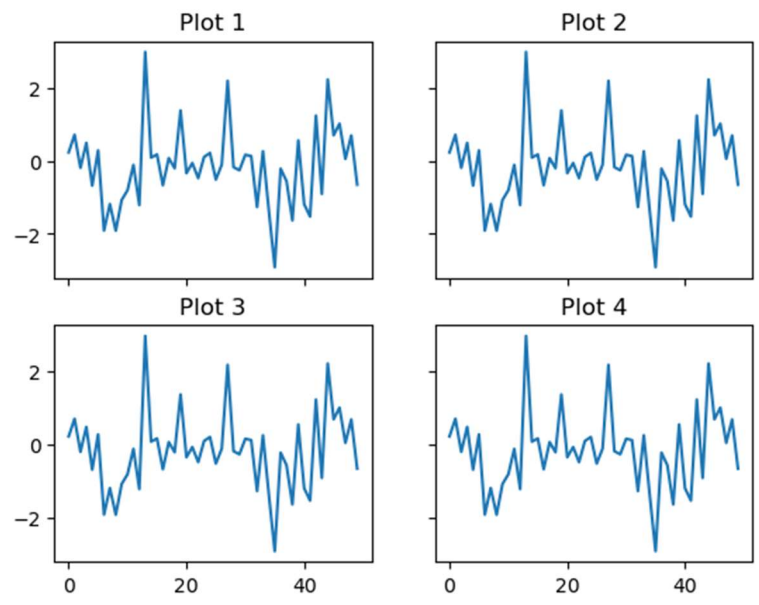
Task: Generate four line plots, each visualizing the same dataset 'data' from the DataFrame. All plots should be displayed within a single figure organized in a 2x2 grid, where each subplot shows the same data line but is individually titled (Plot 1, Plot 2, Plot 3, Plot 4, respectively). The x and y axes should be shared among all subplots to maintain consistency in data comparison.

Style: ay each plot with a line style that clearly denotes changes in data values over row indices. Use a standard color for visibility—preferably a clear contrast like blue. Each plot should include grid lines for easier readability of the data points, axis labels could be optional given the shared axis feature, and each plot must have a title showing its unique identifier.

Generated



Ground truth

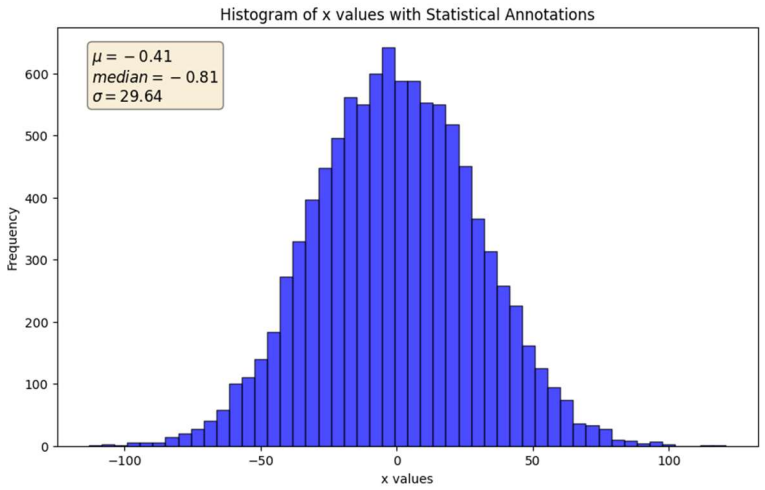


ID = 99
Score vis = 95
Score task = 95
Score human = 100

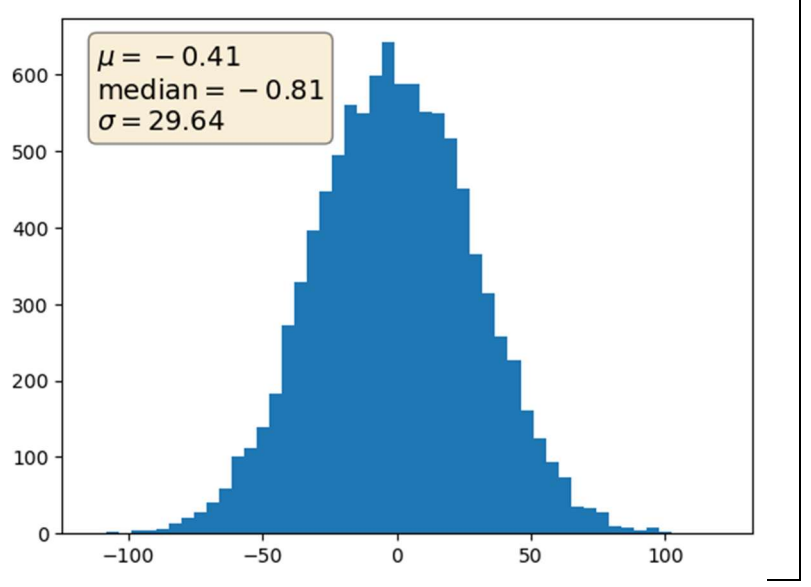
Task: Create a histogram to visualize the distribution of the 'x' values across a specified number of bins. Overlap this histogram with a text box that displays the unique values of 'mu', 'median', and 'sigma' from the DataFrame.

Style: The histogram should have a specific bin count to properly display the data distribution. The text box containing the statistical values should be positioned in the upper left corner of the plot, and it should have rounded corners, a background color to increase readability, and a moderate level of transparency. The font size should be set to ensure readability without overshadowing the plot.

Generated



Ground truth

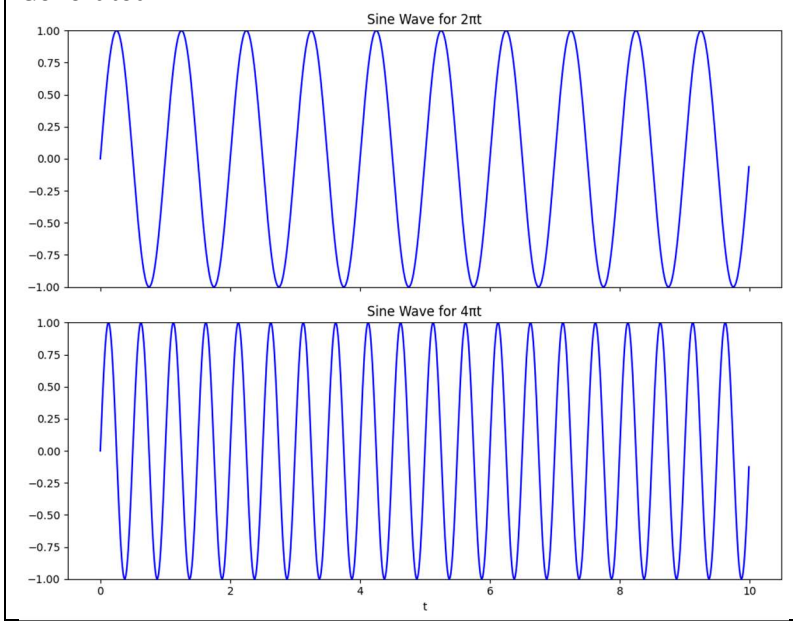


ID = 100
Score vis = 95
Score task = 100
Score human = 100

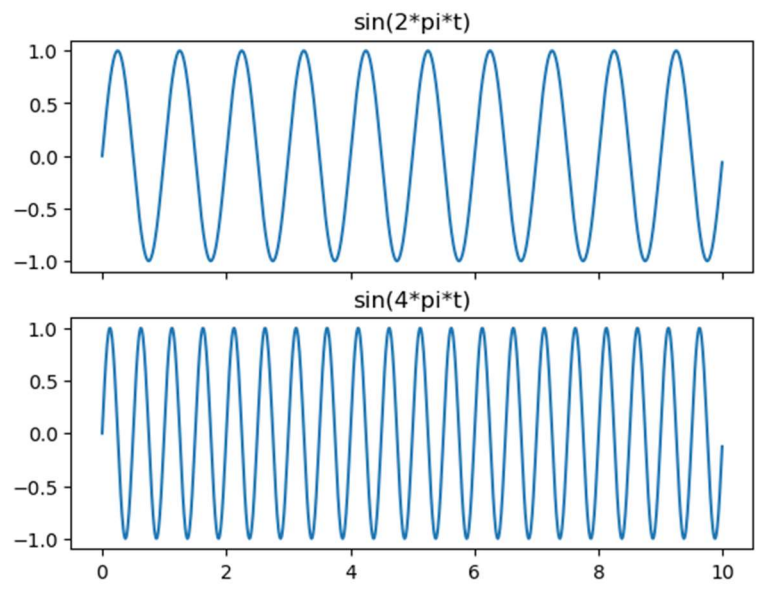
Task: Generate a vertical layout of subplots (2 plots stacked vertically) displaying two distinct sine functions against 't'. The upper plot should depict the sine wave for the function 2πt and the lower plot should depict the sine wave for 4πt. Each plot should share the same x-axis that represents 't'.

Style: Assign titles to each subplot corresponding to their respective sine functions. Opt for a consistent and clear style, using the same color for both lines. Ensure the y-axis of both subplots ranges from -1.0 to 1.0 to maintain uniformity. Additionally, configure the plots to share the x-axis labels, enhancing readability and saving space.

Generated



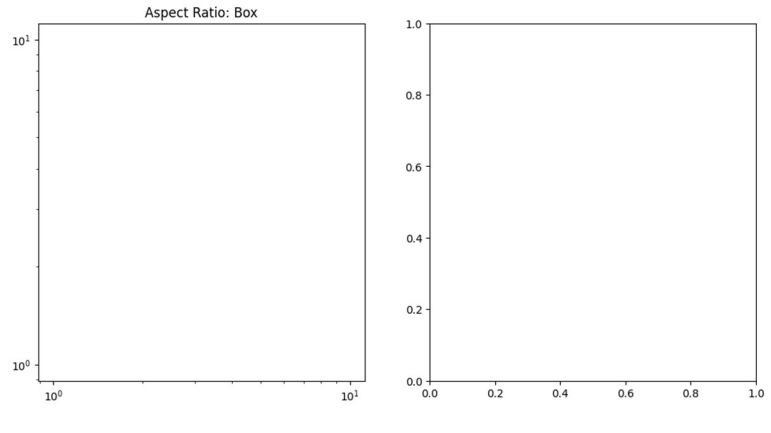
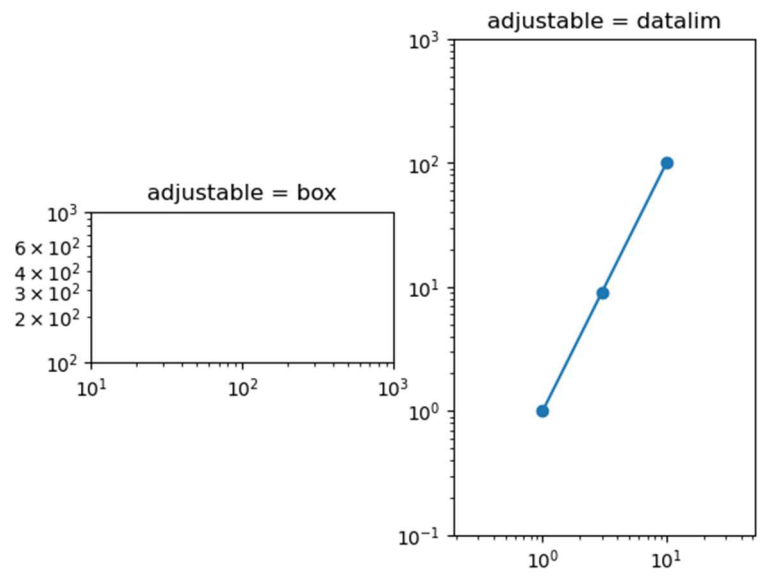
Ground truth



ID = 101
Score vis = 10
Score task = 30
Score human = 30

Task: Create two subplots side by side. Each subplot should have logarithmic scaling on both x and y axes to handle the wide range of the data values. In the first subplot, only the axes and aspect settings are adjusted without displaying data, focusing on the effects of 'box' adjustment on aspect ratio. In the second subplot, plot data points from the dataframe using a line and markers, showcasing the effect of 'datalim' adjustment on aspect ratio and data representation.

Style: Both subplots should have logarithmic scales to properly display the data range. The first subplot should maintain the aspect ratio to highlight the scaling settings, while the second should include data points connected by lines. Set titles to distinguish the adjustment settings ('box' and 'datalim'). Adjust the parameters such as axis limits and aspect ratios to ensure clarity and proportionality in visual presentation.

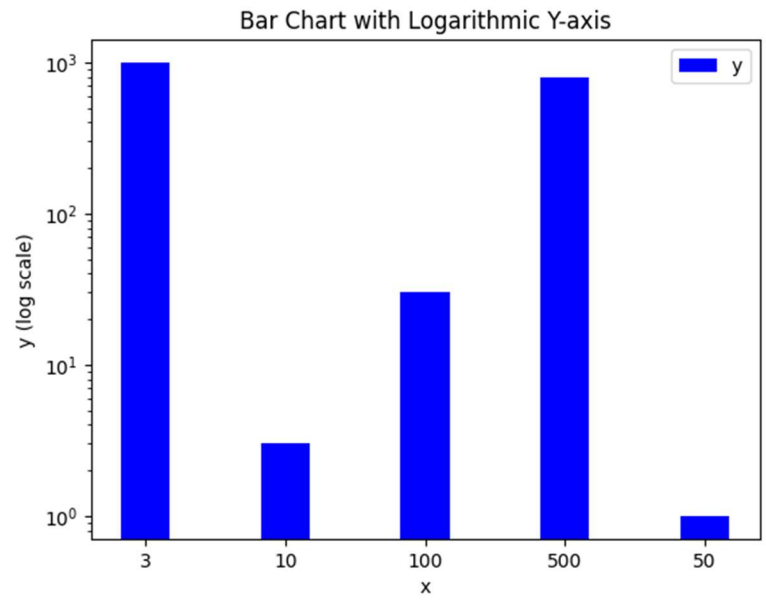
Generated	Ground truth
	

ID = 102
Score vis = 10
Score task = 85
Score human = 85

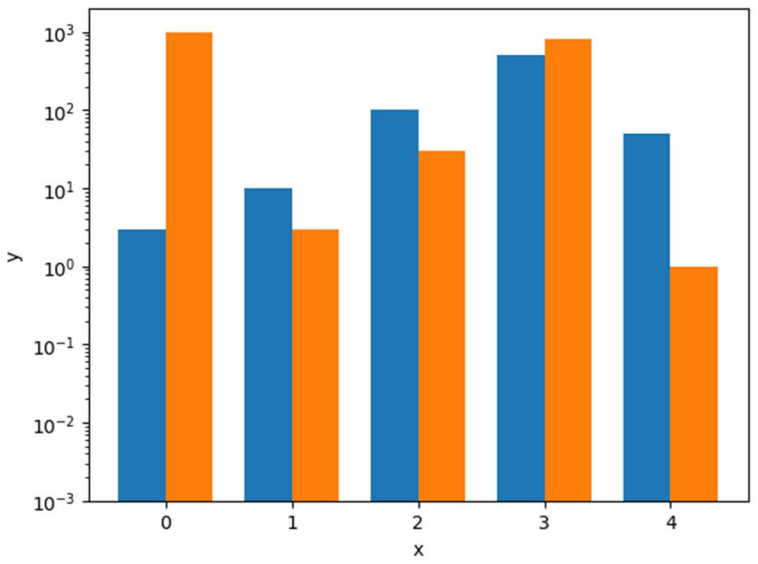
Task: The plot should be a bar chart with each bar's height representing the corresponding value from the 'y' column of the DataFrame. Each bars' placement on the x-axis should be derived from the 'x' column. Multiple sets of data present in the DataFrame will be represented side by side for each x-value, requiring adjusting bar widths and positions appropriately.

Style: The scale of the y-axis should be logarithmic to accommodate wide value ranges effectively. The plot should have a clear label for both x and y axes. Each set of bars, related to different DataFrame columns, should vary in color for better visual distinction.

Generated



Ground truth

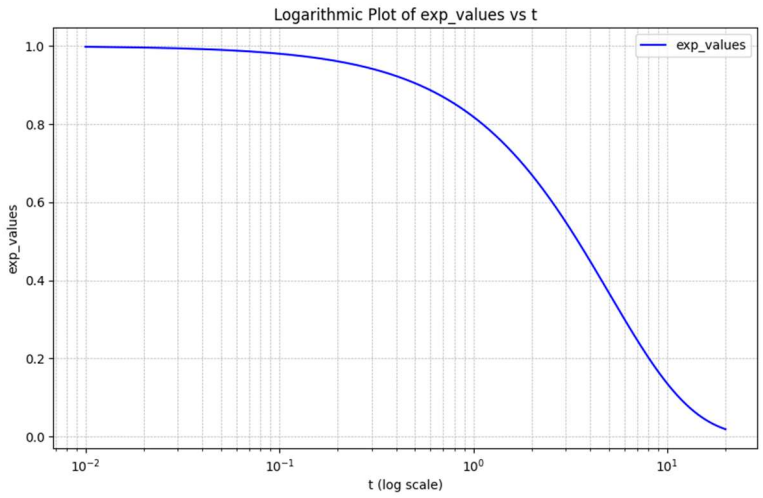


ID = 103
Score vis = 95
Score task = 90
Score human = 100

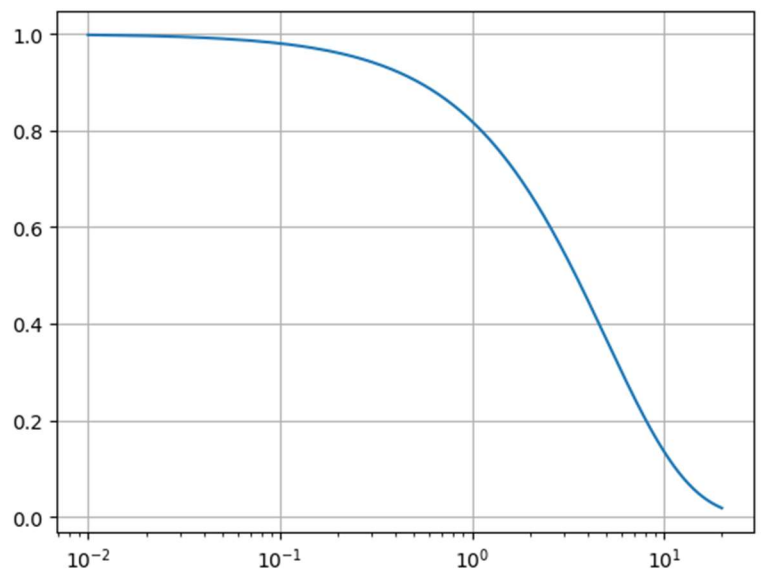
Task: Create a logarithmic plot where the x-axis values are plotted on a logarithmic scale. The plot should directly relate 't' values as the x-axis and 'exp_values' on the y-axis. The line graph should smoothly connect the data points to visualize any trends effectively.

Style: he plot to include a grid for better visibility and interpretation of data points at various scales. Focus on ensuring that the grid complements the logarithmic scale of the x-axis, and enhances overall readability of the plot. Use a clear color and line style for the data line to ensure it stands out against the grid backdrop.

Generated



Ground truth

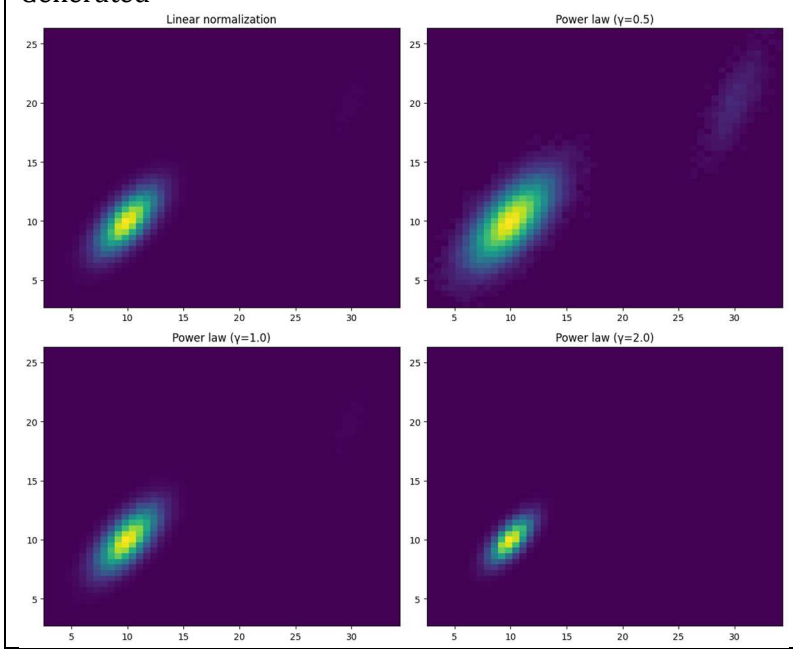


ID = 104
Score vis = 70
Score task = 95
Score human = 100

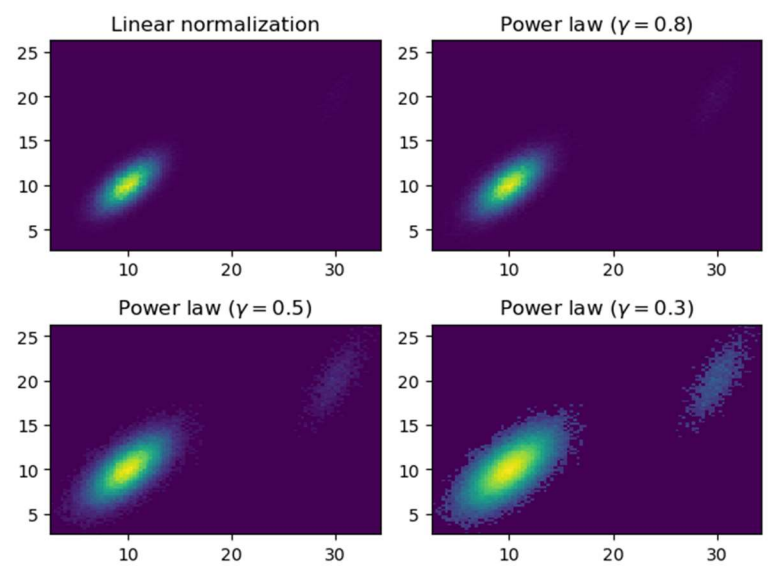
Task: Create a set of four 2D histograms showing the distribution of two variables from the dataframe, arranged in a 2x2 grid format. The first plot should display the data using linear normalization. The subsequent plots should apply a power law normalization with varying gamma values to demonstrate how the distribution visualization changes with different scaling parameters.

Style: Title each subplot to reflect the type of normalization applied (e.g., "Linear normalization" and "Power law (γ=x)"). Adjust the layout to ensure clear spacing and non-overlapping elements among the subplots. Use a consistent color map across all plots to facilitate visual comparison.

Generated



Ground truth

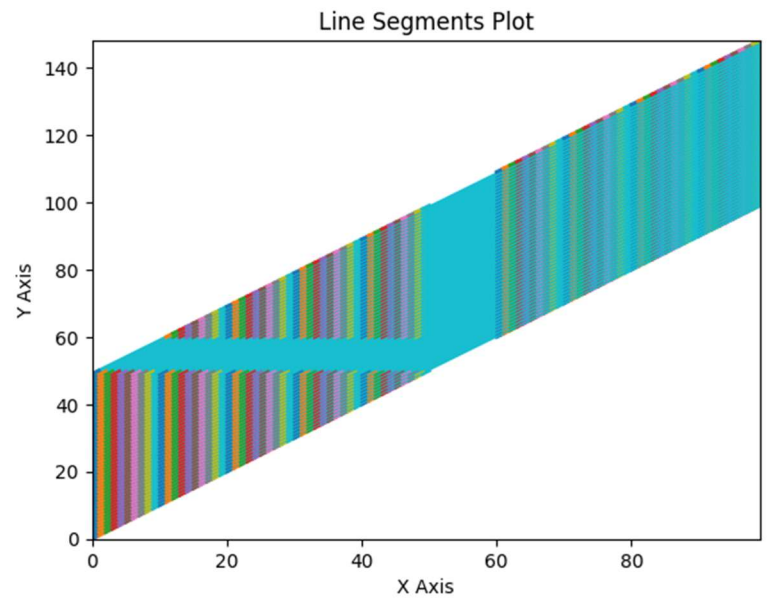


ID = 105
Score vis = 60
Score task = 80
Score human = 80

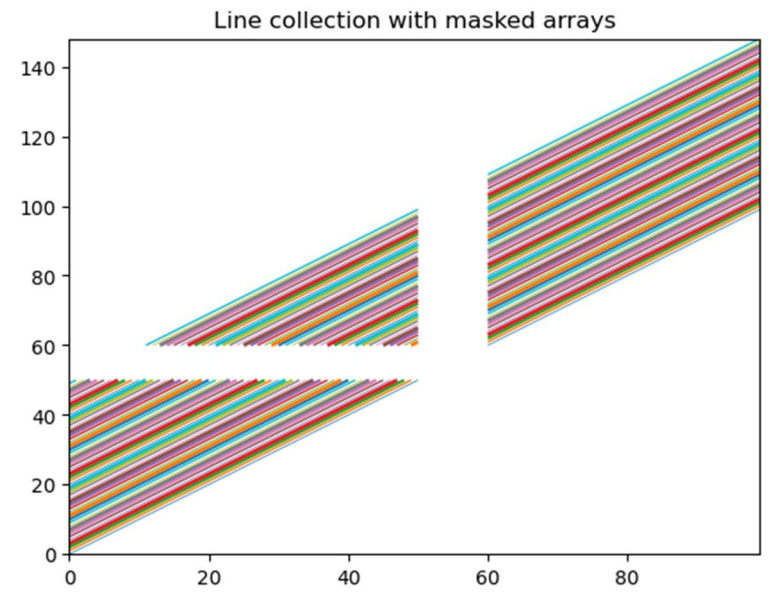
Task: The task involves creating a plot that visualizes data from the DataFrame as multiple line segments. Each line segment should be plotted using the data reshaped to separate the coordinates. The plot should also include custom settings for line colors and styles, adjusting the plot limits to the range of the data, and appropriately labeling the axes.

Style: he plot with varied line widths and solid linestyles, alongside a color scheme that utilizes the default cycle of plot colors. Additionally, include an axis range determined by the data's minimum and maximum values and a plot title. Ensure clear visibility and distinction among the multiple line segments.

Generated



Ground truth

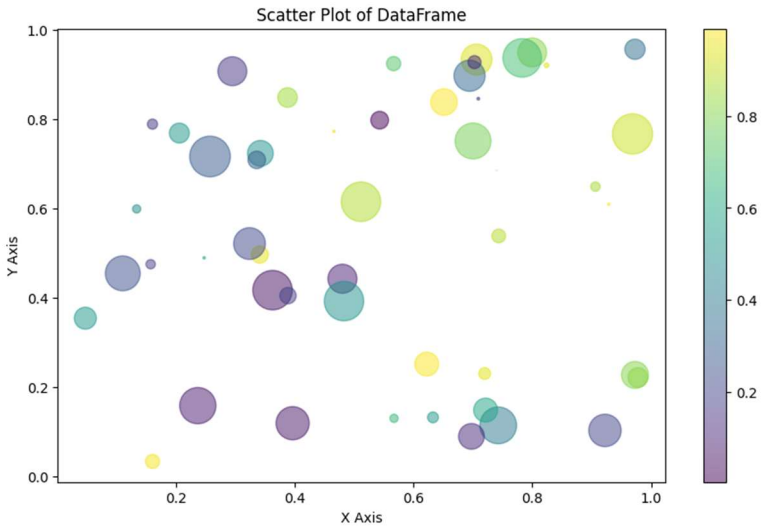


ID = 106
Score vis = 90
Score task = 95
Score human = 100

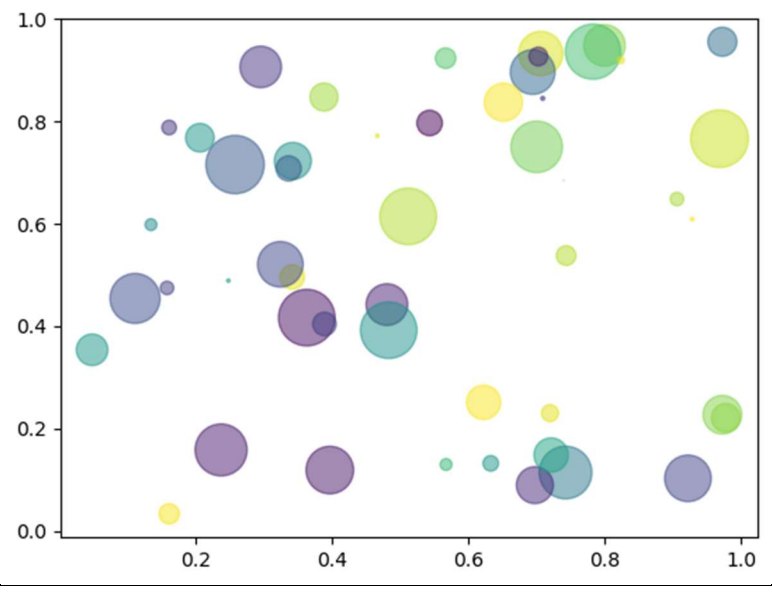
Task: The code should create a scatter plot where the x and y columns of the DataFrame are used as coordinates. Each point in the scatter plot will have a size corresponding to the 'area' column and a color intensity based on the 'colors' column. Each point's transparency should be consistent to illustrate overlap clearly.

Style: The plot style entails having variable sizes for points based on one of the DataFrame columns and varying color intensities for these points according to another column. The plot will also have a semi-transparent aesthetic for each point to enhance the visual overlap of points.

Generated



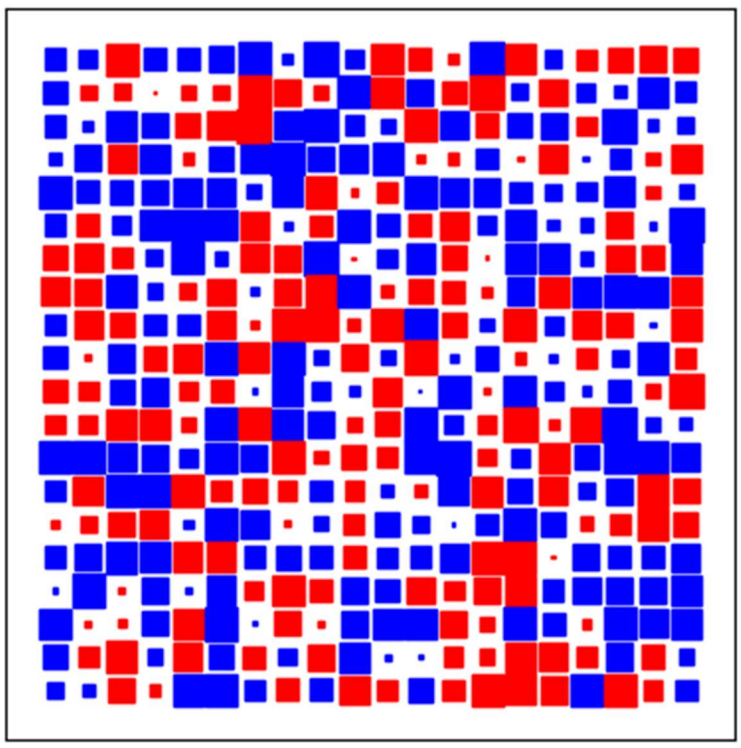
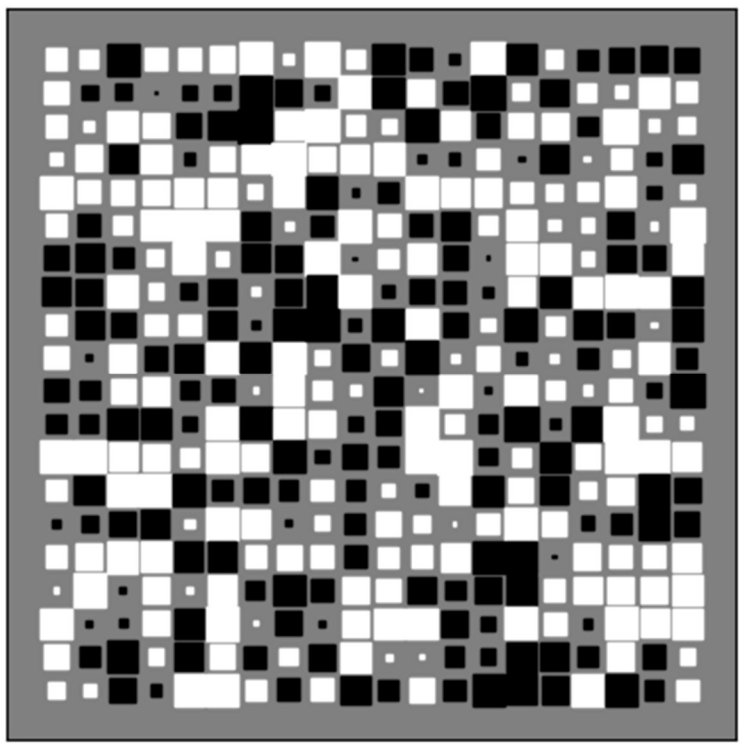
Ground truth



ID = 107
Score vis = 90
Score task = 95
Score human = 100

Task: Create a visual representation using a Hinton diagram to display the magnitude and sign of the values in the DataFrame. Positive and negative values should be differently colored for distinction, and their magnitude represented by the size of the squares in the plot.

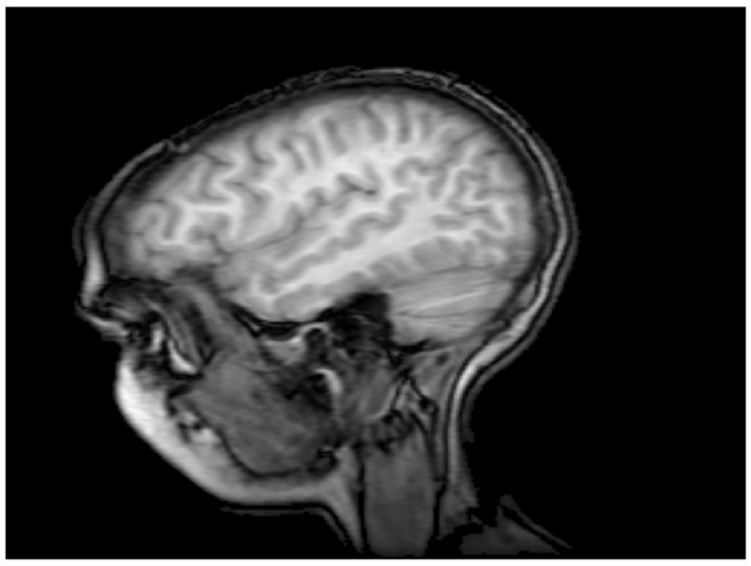
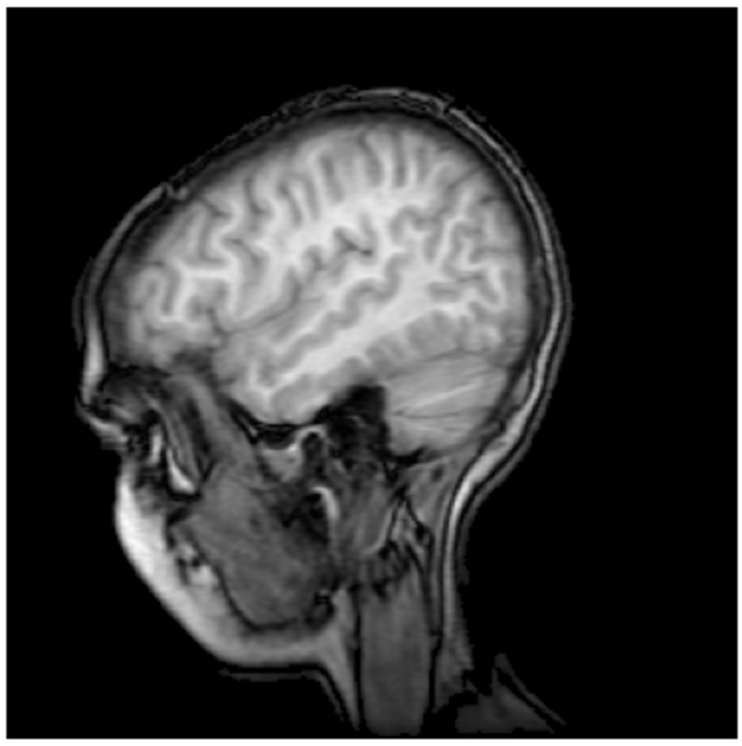
Style: Use distinct colors for positive and negative values, with a consistent background color to enhance the contrast. Apply equal aspect ratio for the axes to maintain square proportions for each value, and ensure that the axis markers are hidden to emphasize the data representation over the grid layout.

Generated	Ground truth
	

ID = 108
Score vis = 100
Score task = 100
Score human = 100

Task: The plot to be generated should visually represent the DataFrame data as an image without axes. The data values across the DataFrame should be interpreted as pixel intensity values to reconstruct an image. The visualization should adopt a grayscale color map to appropriately represent the data's intensity variation for visual clarity and interpretability.

Style: The style of the plot should be set to exclude axis labels and ticks to emphasize the visualization of the image data alone. The grayscale color map should be used to map the intensity values to the appropriate shades in the plot. This approach enhances the clarity of the image presentation, emulating formats commonly used in medical imaging display.

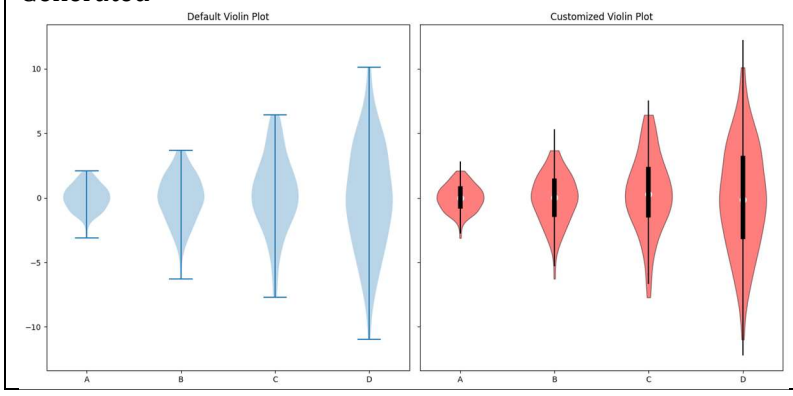
Generated	Ground truth
	

ID = 109
Score vis = 95
Score task = 100
Score human = 100

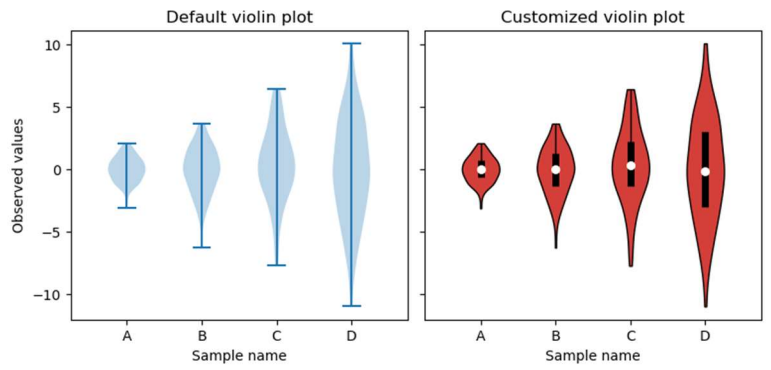
Task: Create two subplots in a single figure. The left subplot to display the default style violin plot for the DataFrame data, detailing overall distribution per column. The right subplot to show a customized violin plot where medians, IQR (Interquartile range), whiskers (corresponding to 1.5*IQR beyond the quartiles), and individual column's 'extreme' data points are emphasized. Both plots should share the Y-axis for easy comparison.

Style: Customize the appearance of both plots. For the default plot, use default styling options. For the customized plot, color the violin bodies in red, with black outlines and set transparency for clarity. Highlight the medians as white circles, IQR as thick black lines, and whiskers as thin black lines. Also, ensure that the X-axis is labeled with distinct sample names ('A', 'B', 'C', 'D'), and all plots are properly titled and labeled for clarity and aesthetics. Ticks should be oriented outward, and spacing should be adjusted for visual ease.

Generated



Ground truth

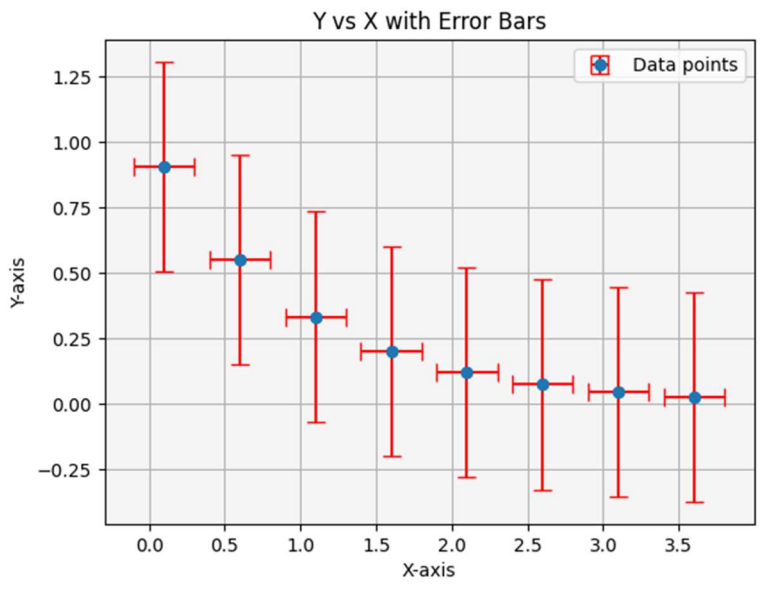


ID = 110
Score vis = 90
Score task = 95
Score human = 100

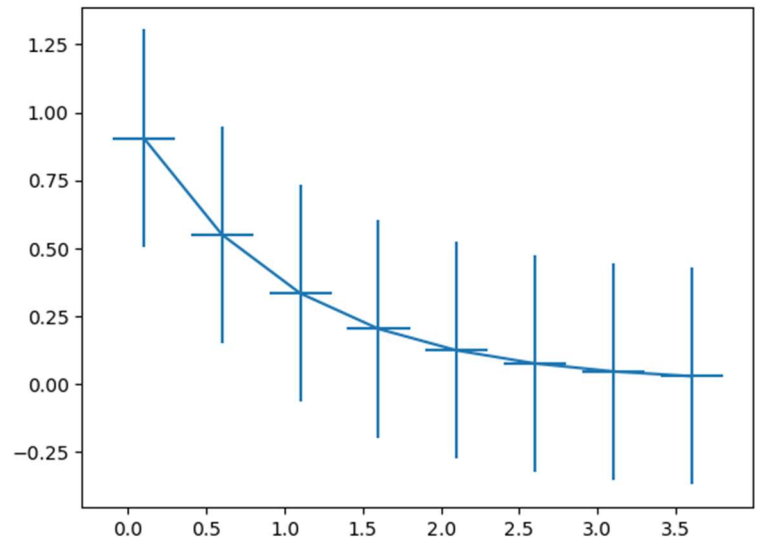
Task: Create a plot showing data points for y versus x. Include error bars along both the x and y dimensions, denoting the uncertainty in measurements. These error bars should be based on the 'xerr' and 'yerr' values present in the DataFrame. Use a line plot with markers at each data point to depict how y varies with x. Each point should be distinctly visible to identify possible trends or patterns in the underlying data.

Style: Enhance the visual appeal and clarity of the plot with a clean and simple aesthetic. The plot should have readable axis labels, a grid for easier visualization of the data points and their error bars, and a light background. The color choice for data points and error bars should be distinct and maintain good contrast against the background for clarity. Select a line color that stands out and ensures that data markers are clearly discernible. Use error bars that are easily visible without overwhelming the data points.

Generated



Ground truth

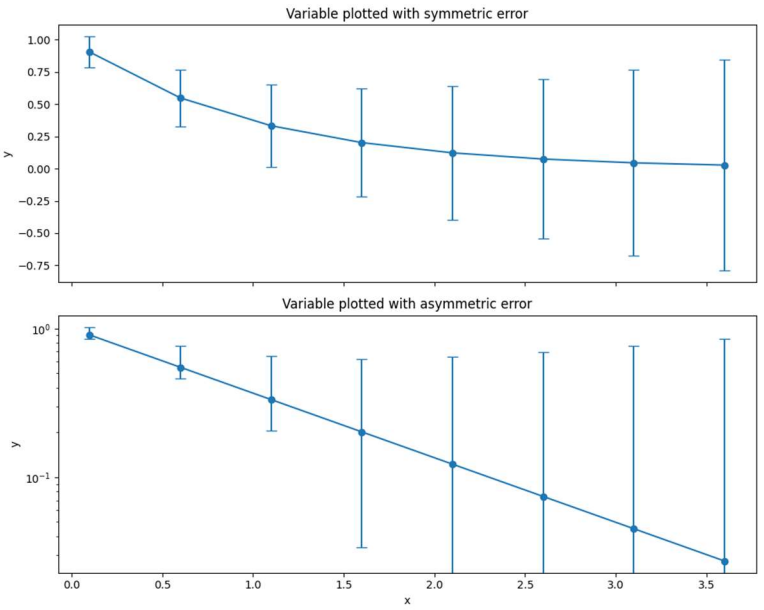


ID = 111
Score vis = 70
Score task = 100
Score human = 70

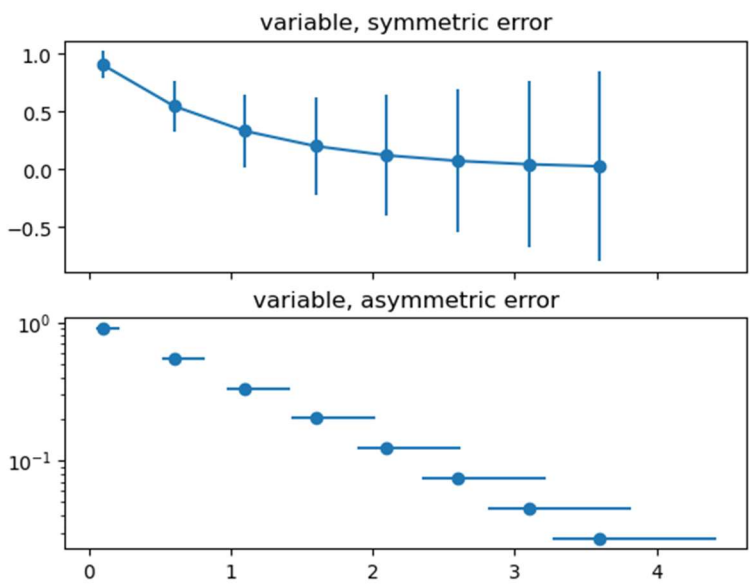
Task: Create a visualization comprising two subplots stacked vertically, sharing the x-axis. The top plot should display the 'y' values against 'x' with symmetric error bars represented by the 'error' column. The bottom plot should also display 'y' against 'x' but with asymmetric error bars, where the errors in the 'x' direction are derived from the 'lower_error' and 'upper_error' columns. The y-axis of the bottom plot should be on a logarithmic scale.

Style: For both plots, format the data points as circles connected by lines. In the top plot, error bars should be attached directly to each data point, indicating confidence intervals. The bottom plot should similarly display error bars, yet with distinct lengths on either side of each data point, representing the variability specific to either side. The top plot should carry a title indicating the variable is plotted with symmetric error, and the bottom plot should state the variable is plotted with asymmetric error.

Generated



Ground truth

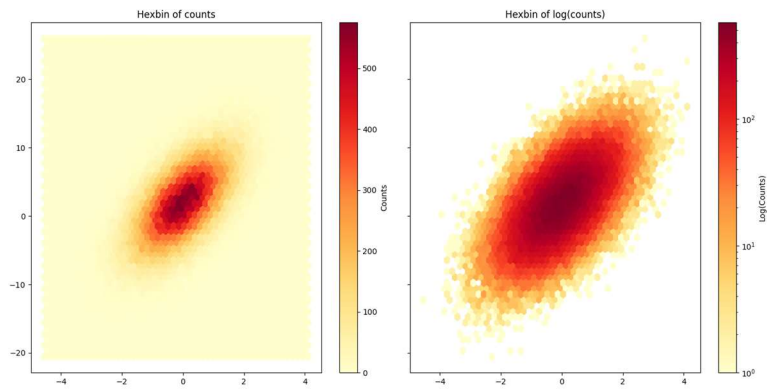


ID = 112
Score vis = 90
Score task = 100
Score human = 100

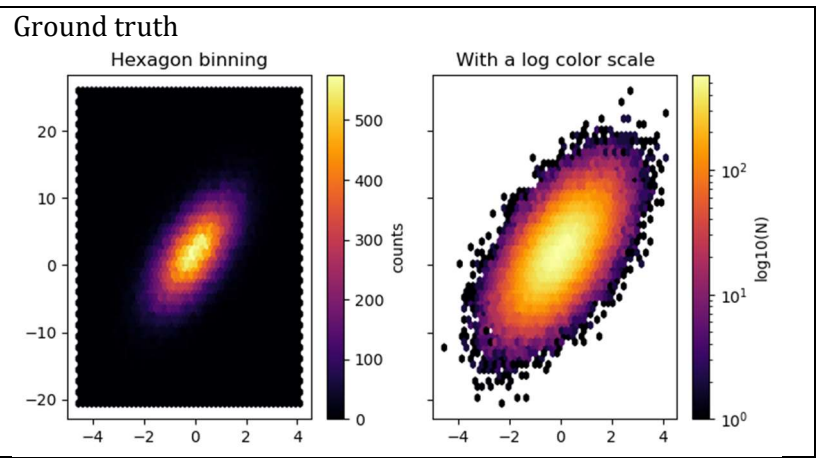
Task: Generate two subplots side by side. Both plots should implement hexagon binning of the 'x' and 'y' data from the dataframe but differ in the way they scale visually: the left plot showing the raw count of points in each bin and the right plot using a logarithmic scale for counts. The axes should be shared across the plots for better comparison.

Style: Both plots should have appropriate titles reflecting their scaling (normal and logarithmic). Employ a warm colormap that transitions from dark to light to represent increasing counts. Additionally, add color bars to each plot showing the range of counts or the logarithm of counts, respectively. Adjust the spacing and size of the plots for clarity and aesthetics.

Generated



Ground truth

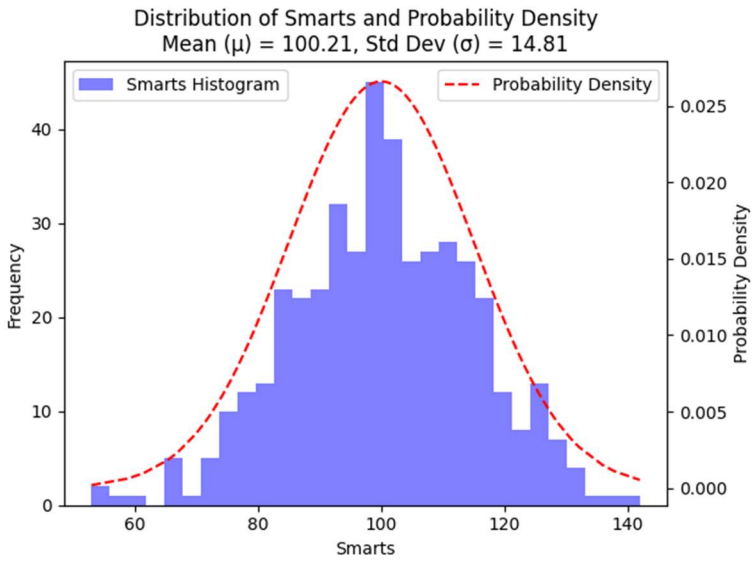


ID = 113
Score vis = 90
Score task = 95
Score human = 100

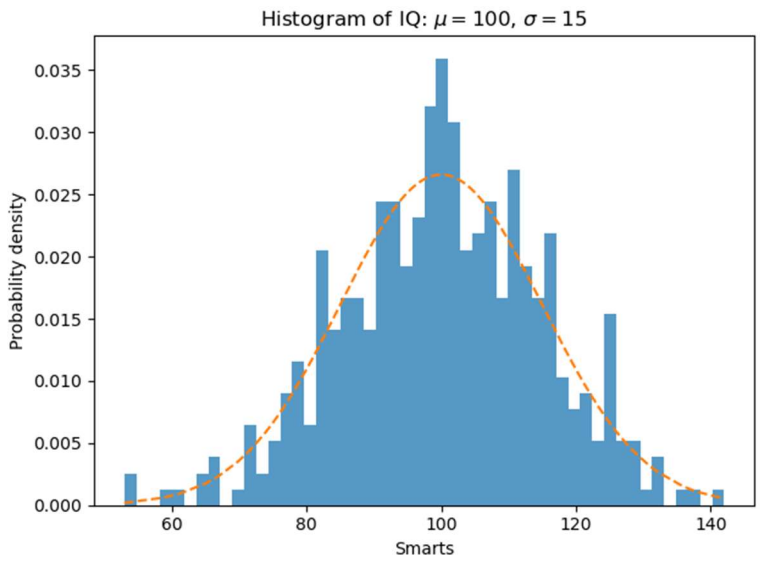
Task: Generate a histogram from the 'Smarts' column to visualize the distribution of data points across specified bins. Overlay this with a line plot that traces the probability density related to the 'Smarts' values, providing a dual representation of the data.

Style: abels for the x-axis and y-axis describing the variables. Include a title on the plot incorporating mathematical notation for mean (μ) and standard deviation (σ). Opt for a semi-transparent fill for the histogram for visual depth, and a dashed line style for the probability density line to distinguish it from the histogram. Use layout adjustments to ensure all labels and titles are cleanly visible without clipping.

Generated



Ground truth

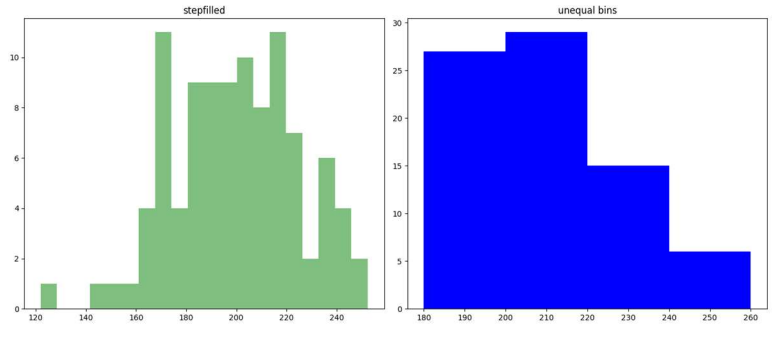


ID = 114
Score vis = 30
Score task = 95
Score human = 80

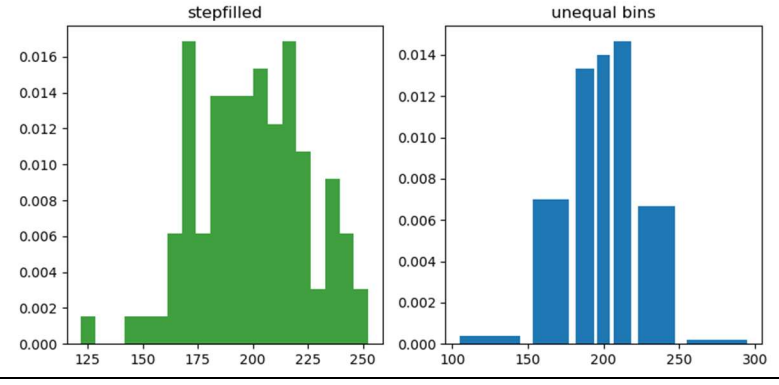
Task: Generate two histograms from the 'Data' column in a side-by-side configuration within a single figure. The first plot should feature a filled step plot of the data distribution, using regular bin sizing. The second plot should display the data distribution using bars with specified unequal bin edges to highlight distribution variations more clearly.

Style: For the first histogram, use a green color with a semi-transparent fill. The second histogram should be displayed using a solid blue color for the bars. Each subplot must have its own title reflecting the binning strategy ('stepfilled' for the first and 'unequal bins' for the second). Adjust layout to ensure clarity and non-overlapping elements within the visual display.

Generated



Ground truth

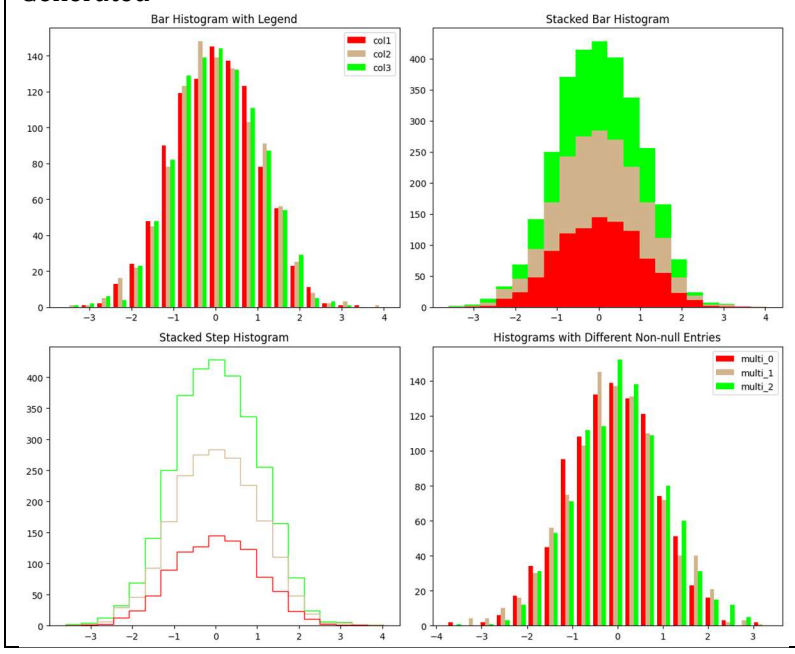


ID = 115
Score vis = 90
Score task = 95
Score human = 100

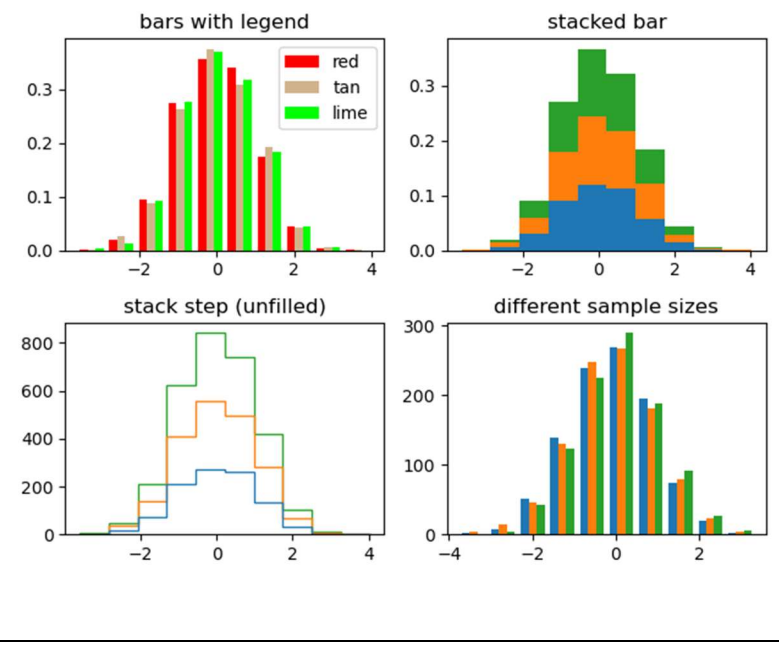
Task: Create a 2x2 grid of histograms with each subplot presenting a different style and subset of the dataframe columns. The first plot should use the 'bar' histogram type with a legend showing different colors for each dataset. The second plot should display a stacked bar histogram. The third should feature stacked step histograms without filling. The fourth is a set of histograms displaying datasets that may have different numbers of non-null entries, using regular bar type. Titles should be added to each subplot to describe the content.

Style: Apply specific color schemes to different histogram bars and steps. Use red, tan, and lime colors for the different datasets where applicable. Enable legends for plots that include multiple colors to aid in dataset identification. Format the layout tightly so that there is no overlapping of elements within the plot. Each histogram should be clear and easily interpretable, with appropriate axis labels and plot titles based on the content.

Generated



Ground truth

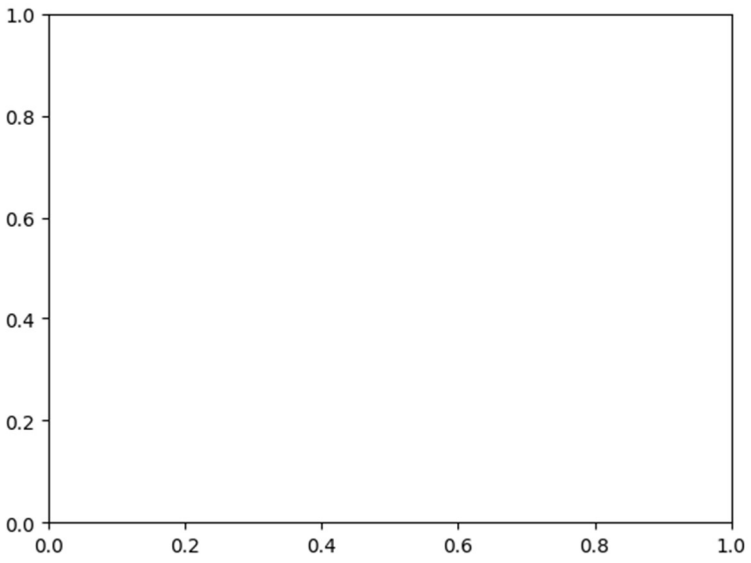


ID = 116
Score vis = 0
Score task = 0
Score human = 0

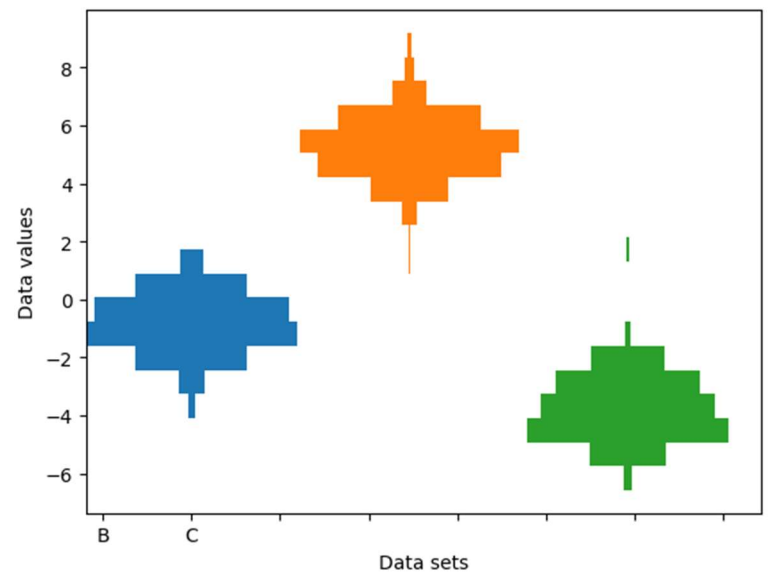
Task: The plot will be a horizontal bar chart, displaying grouped data. For each category in the 'label' column, horizontally align bars based on the 'centers'. The length of each bar will represent the 'binned_data' values, the bars will start from the positions specified in 'lefts', and their heights will be uniform as specified in 'heights'.

Style: Enhance the plot aesthetics by assigning distinctive colors to bars based on their category. Label the horizontal and vertical axes appropriately to indicate the type of the data they represent. Add category labels to the horizontal axis to denote the different datasets represented.

Generated



Ground truth

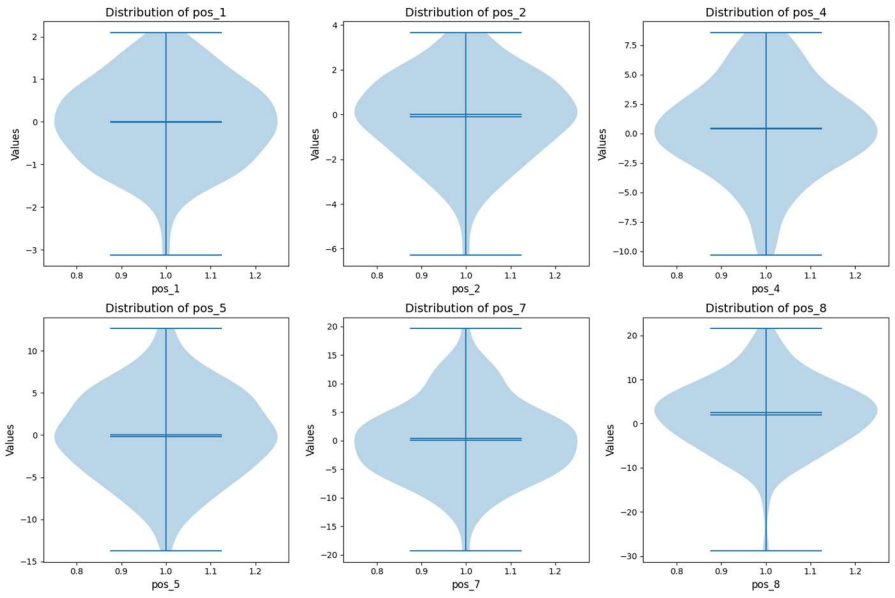


ID = 117
Score vis = 70
Score task = 90
Score human = 100

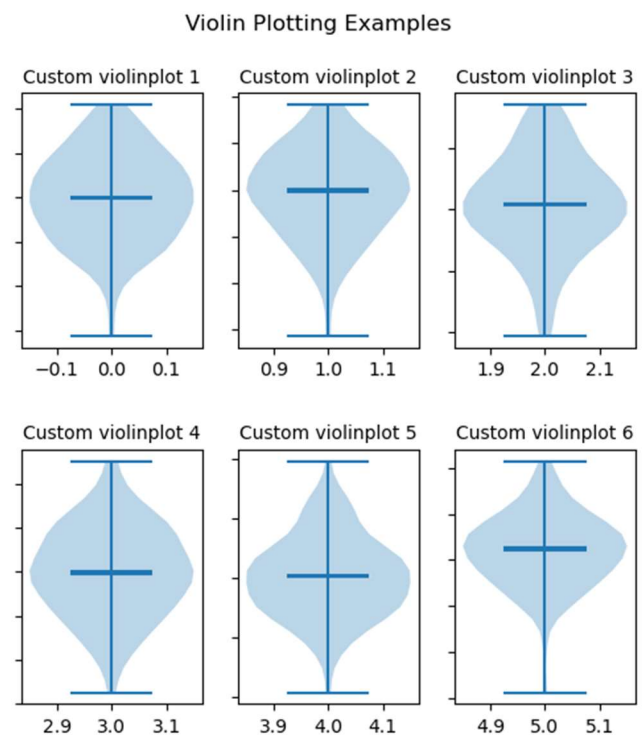
Task: Generate multiple violin plots for each column in the dataframe, arranged in a 2x3 grid layout. Each plot will display the distribution of values for its respective column, highlighting characteristics like the median, mean, and extrema. The plots should be clearly labeled to identify which plot corresponds to which column.

Style: he plots with a custom fontsize and adjust the layout to prevent overlapping plots and labels. Set titles for each subplot to enhance readability. The overall appearance should be clean and professional, with appropriate adjustments to spacing and dimensions to ensure clarity and visual appeal.

Generated



Ground truth

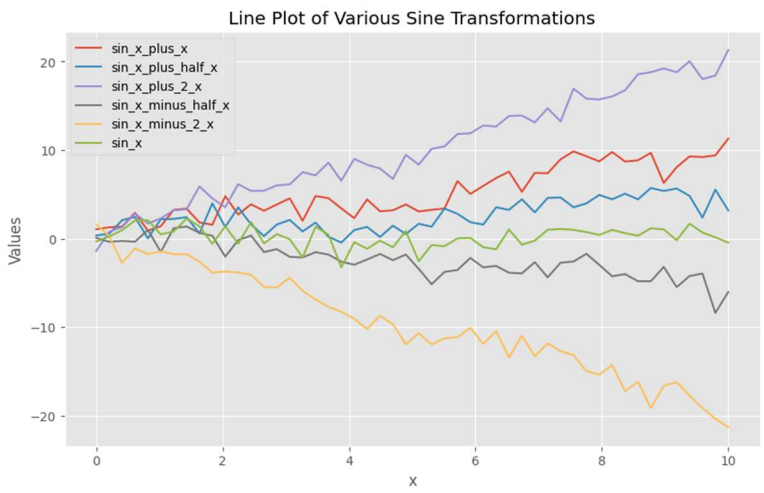


ID = 118
Score vis = 80
Score task = 95
Score human = 95

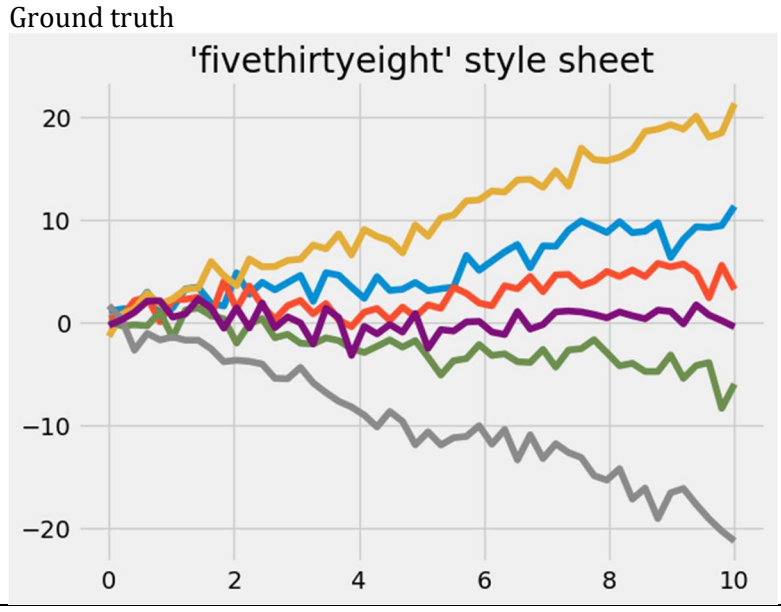
Task:
Generate a line plot for each column of the dataframe, excluding the first column. Each column's values are plotted against the values of the first column 'x', resulting in multiple lines on a single graph. Include a title for the plot that describes its style or theme.

Style:
Apply a specific style to enhance the visual appeal and clarity of the plot. The style should give the plot a modern and clean look, featuring distinct colors for each line to facilitate easy differentiation among them.

Generated



Ground truth

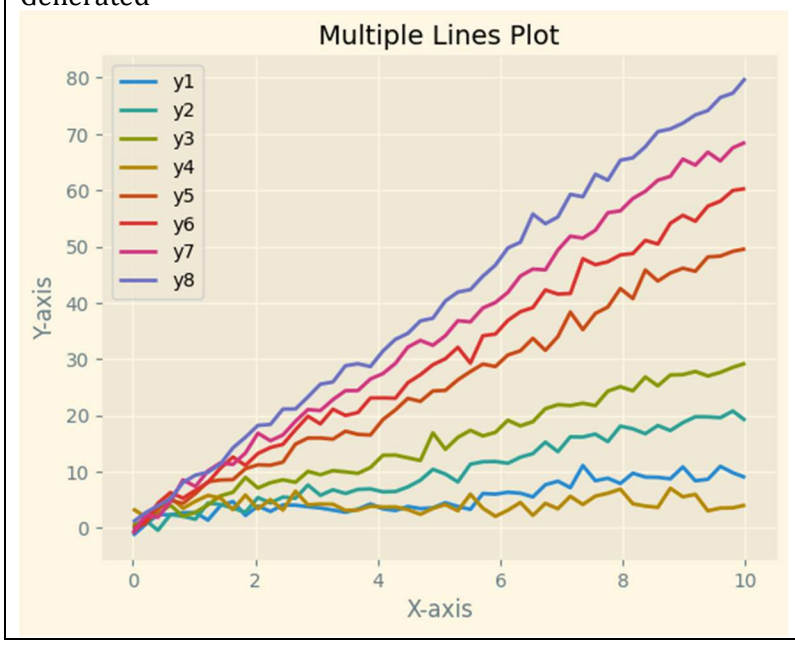


ID = 119
Score vis = 95
Score task = 100
Score human = 100

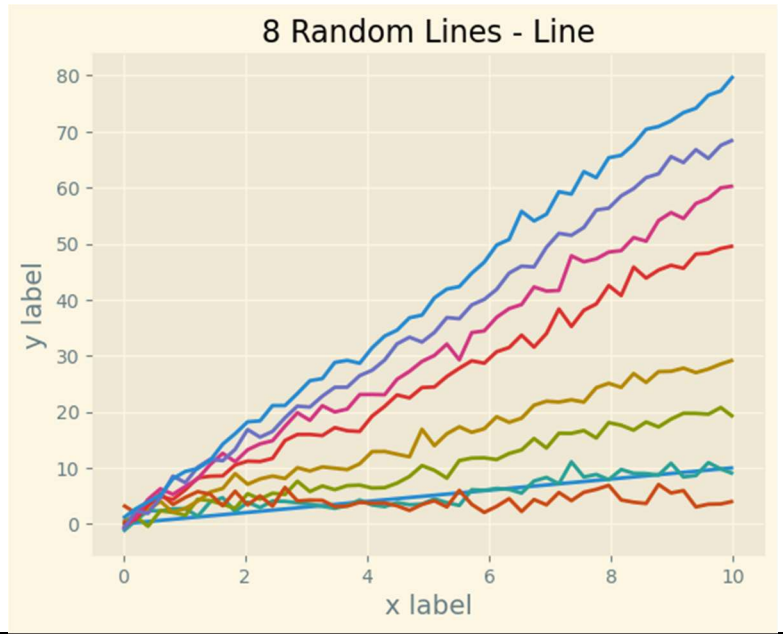
Task: The visualization will involve plotting multiple lines on a single chart. For every 'y' column in the DataFrame ('y1' to 'y8'), a line will be plotted against the 'x' column values. This will result in multiple distinct lines representing the progression of each 'y' column's values against 'x'. A title, as well as labels for the x-axis and y-axis with specific font sizes, will be configured.

Style: The plot will utilize a predefined style ('Solarize_Light2') to format its appearance. This style will influence the color theme, background, grid lines, and overall aesthetics of the plot, giving it a distinctive look while ensuring clarity and readability of the data represented by the lines.

Generated



Ground truth

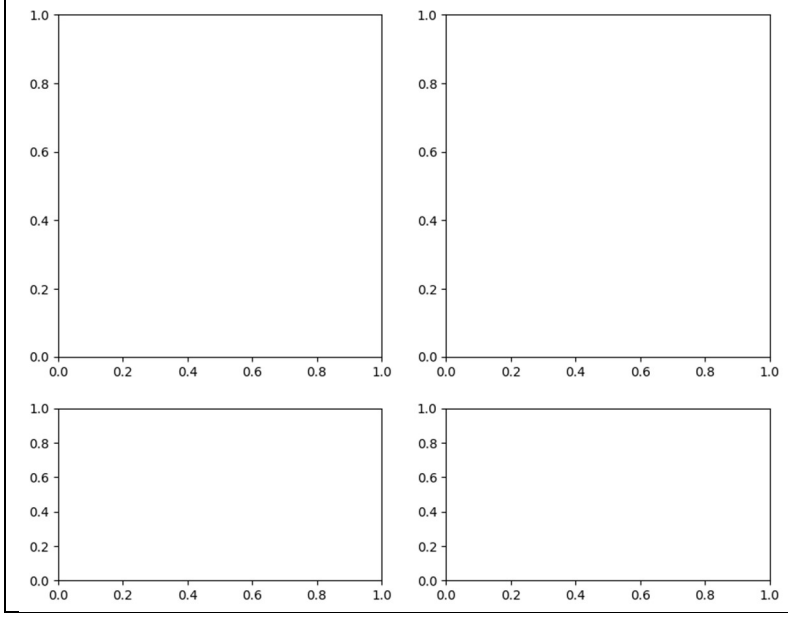


ID = 120
Score vis = 10
Score task = 0
Score human = 15

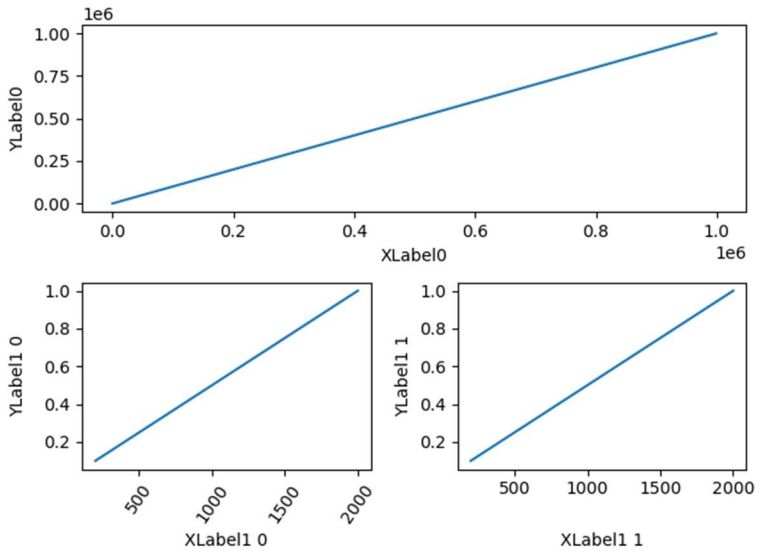
Task: Create a figure divided into a grid with one row of one plot across the top and two smaller plots below it, each displaying different data sets from the dataframe. For each subplot, plot the respective X and Y data as line plots, label the axes appropriately, and for the bottom row, rotate x-axis labels for better readability.

Style: Enable tight layout to manage spacing between plots. Above each subplot, carefully label X and Y axes, and ensure axis labels across the figure are aligned. Use line plots to clearly demonstrate the trends in data across different subplots.

Generated



Ground truth

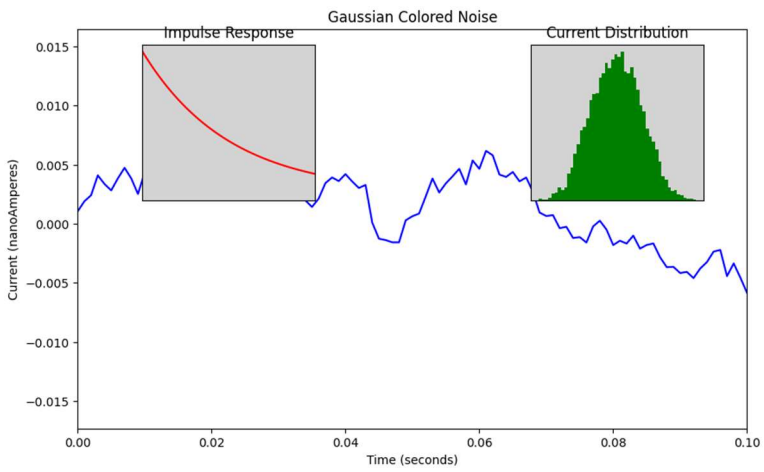


ID = 121
Score vis = 60
Score task = 95
Score human = 85

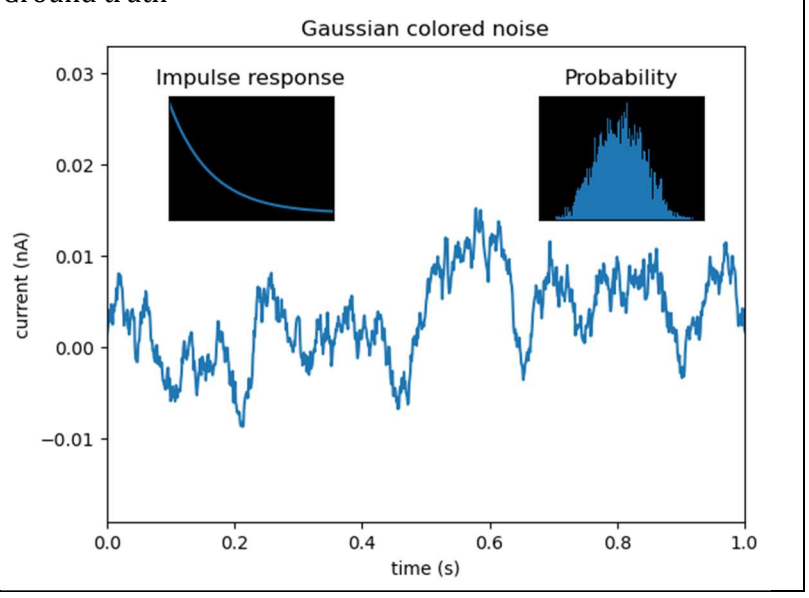
Task:
The primary plot will be a line graph representing 'Current' as a function of 'Time', aimed at showing signal behavior over time. Adjust axes limits to focus on a subset of the data and increase display precision. Include two additional inset plots: one histogram showing the distribution of 'Current' values to assess the probability density, and one small line graph for the 'Impulse_Response' to illustrate a response pattern over a limited time segment.

Style:
The main plot should have axes labeled to indicate that 'Time' is in seconds and 'Current' is in nanoAmperes, with a title labeling the graph as representing Gaussian colored noise. Inset plots should be highlighted with a contrasting background color to stand out against the main plot. The histogram and impulse response graphs should have titles but no axis ticks or labels, ensuring a clean visual emphasis on the graphical data rather than specific numerical values.

Generated



Ground truth

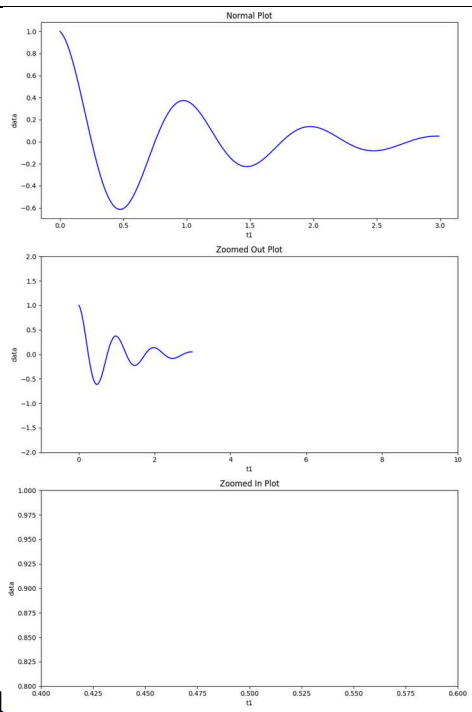


ID = 122
Score vis = 80
Score task = 95
Score human = 85

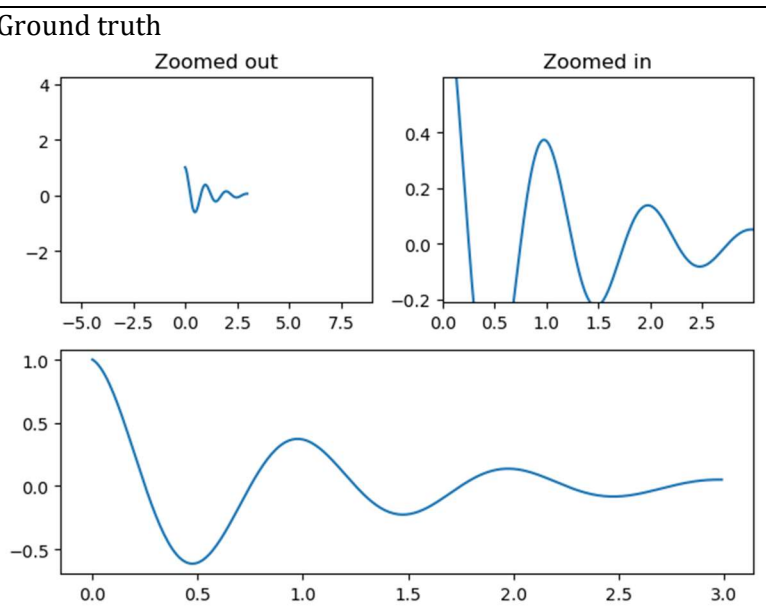
Task: Generate three separate plots. Begin by subdividing the plotting area into suitable sections to accommodate all plots. Each sub-plot will depict a line graph of the same dataset with distinct margin settings to change how the graph views the data. One plot will appear normal, the second will be zoomed out significantly to include much data beyond the typical range, and the third will zoom into the center of the data range. Properly title each section to reflect the different margin settings.

Style: Adjust the style and layout of each plot to suit different margin settings. Employ tight layout adjustments to ensure the plots are neatly organized without overlapping elements. Set titles for at least two of the plots to direct the observer to the effect of the different margin settings (zoomed out and zoomed in). Use a consistent line style and color across all plots for visual consistency and clarity. Adjust the margins to demonstrate the effects of zooming in and out by varying the scale and focus on the dataset in each subplot.

Generated



Ground truth

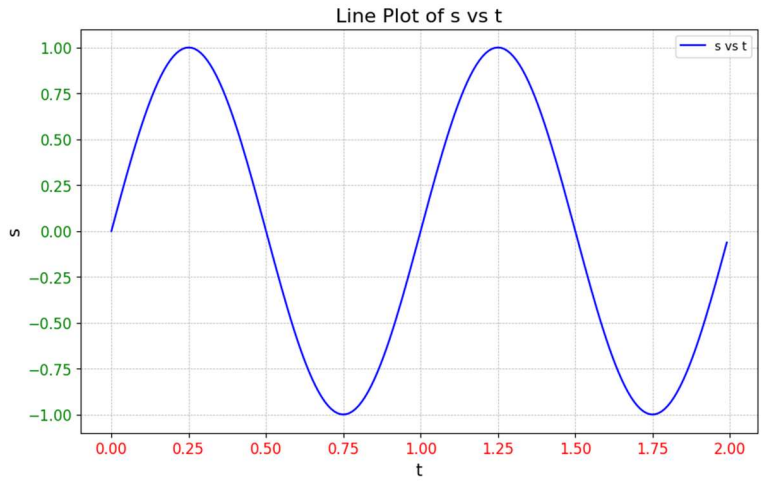


ID = 123
Score vis = 90
Score task = 90
Score human = 100

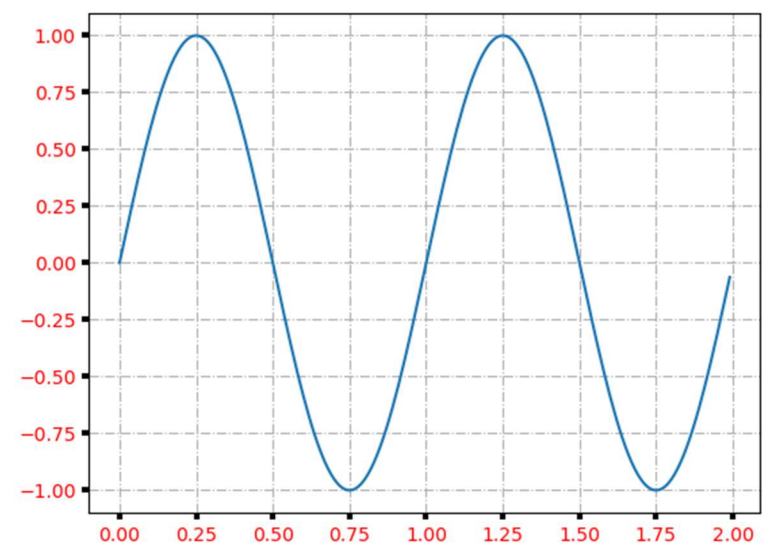
Task: description: Generate a line plot where the x-axis represents the values from column 't' and the y-axis represents the values from column 's'. The plot should effectively reflect the sequential relationship and variation between the two variables.

Style: description: The plot should include a grid with a specific line style to enhance readability. The axes ticks should be colored and styled distinctly to emphasize the data points, enhancing both aesthetics and clarity.

Generated



Ground truth



ID = 124
Score vis = 60
Score task = 100
Score human = 95

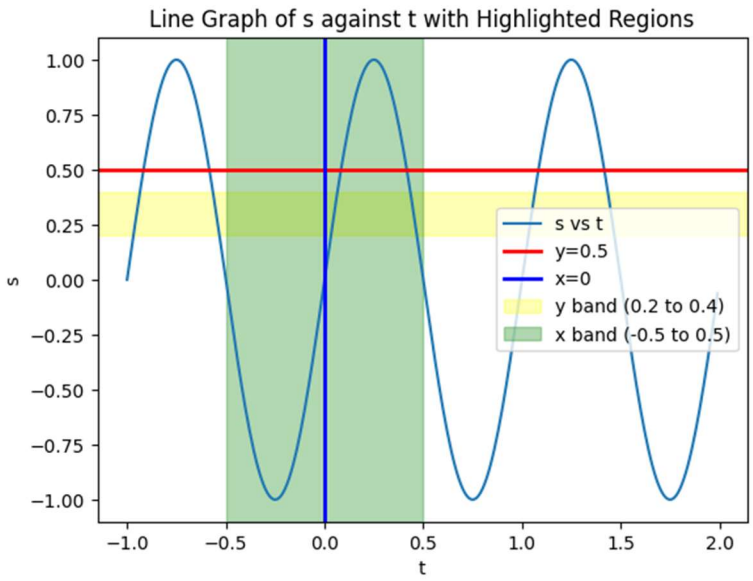
Task:

The plot is a line graph of 's' against 't'. Additional graphical features include horizontal and vertical lines at specific y-values and x-values, respectively. There are also shaded horizontal and vertical bands at certain intervals to highlight specific ranges along the x and y axes. These graphical elements are used to emphasize points, thresholds, or regions of interest in the plot.

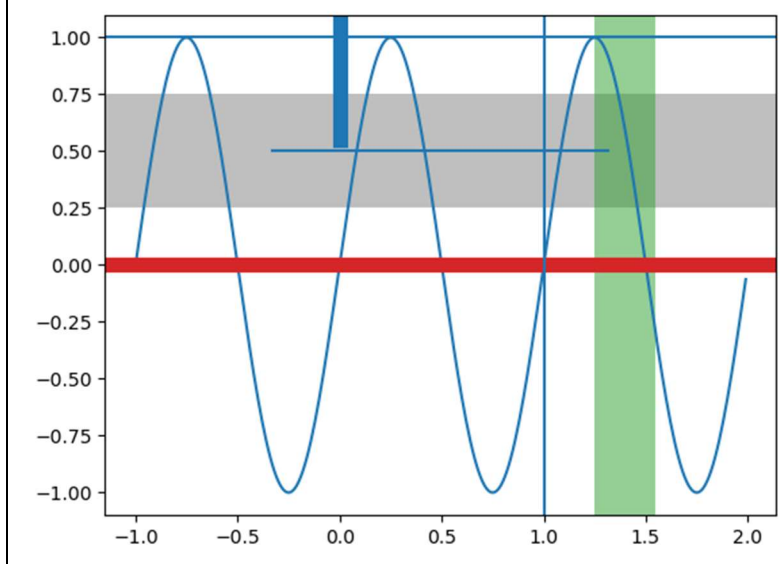
Style:

Stylistic attributes include a thick red horizontal line and a thick blue vertical line at defined positions. Also included are less prominent default lines along with shaded areas with specific opacities and colors to distinguish different ranges visually. The plot employs a color palette that enhances the visualization, aiding in the differentiation of the various graphical components for clarity and visual appeal.

Generated



Ground truth

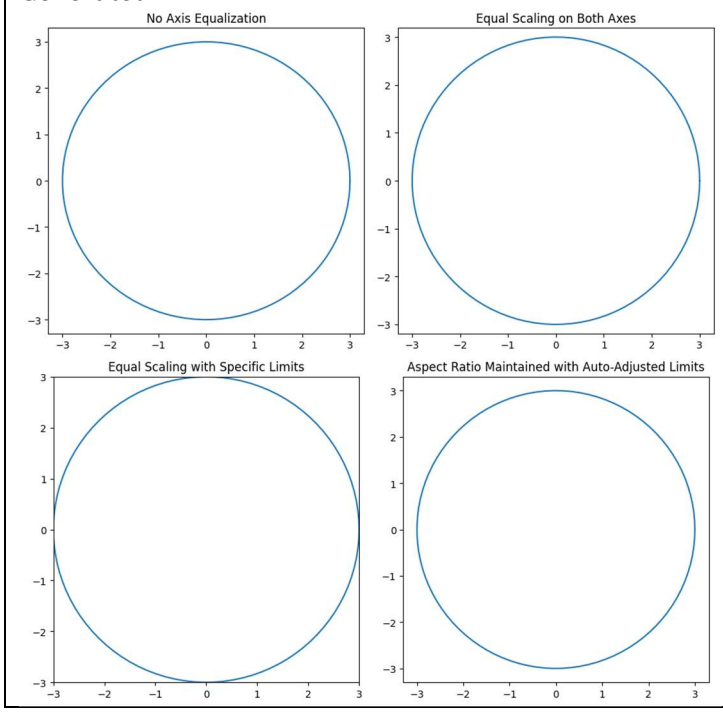


ID = 125
Score vis = 80
Score task = 100
Score human = 95

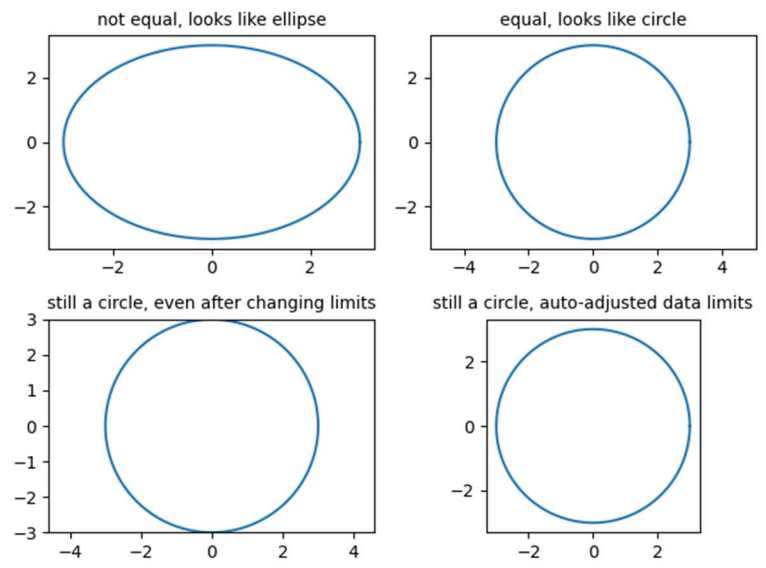
Task: Create four subplots arranged in a 2x2 grid. Each subplot should plot 'x' versus 'y' from the DataFrame. Customize each subplot to show different axis properties that affect the aspect ratio and scaling:
- First subplot should have no specific axis equalization, demonstrating how the plot looks typically.
- Second subplot should enforce equal scaling on both axes to transform the visual into a circle.
- Third subplot should not only ensure equal scaling but also apply specific x and y limits.
- Fourth subplot should maintain the aspect ratio using an 'equal' setting with auto-adjusted data limits.

Style: Apply a title to each subplot that describes the visual effect resulting from the different plot configurations. Adjust the subplot layouts for optimal spacing to ensure that the titles and axes do not overlap, enhancing readability.

Generated



Ground truth

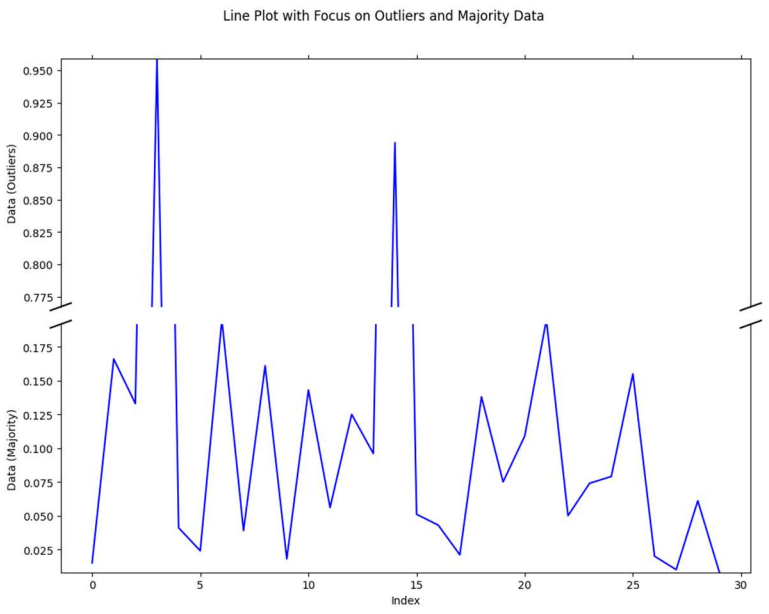


ID = 126
Score vis = 95
Score task = 95
Score human = 100

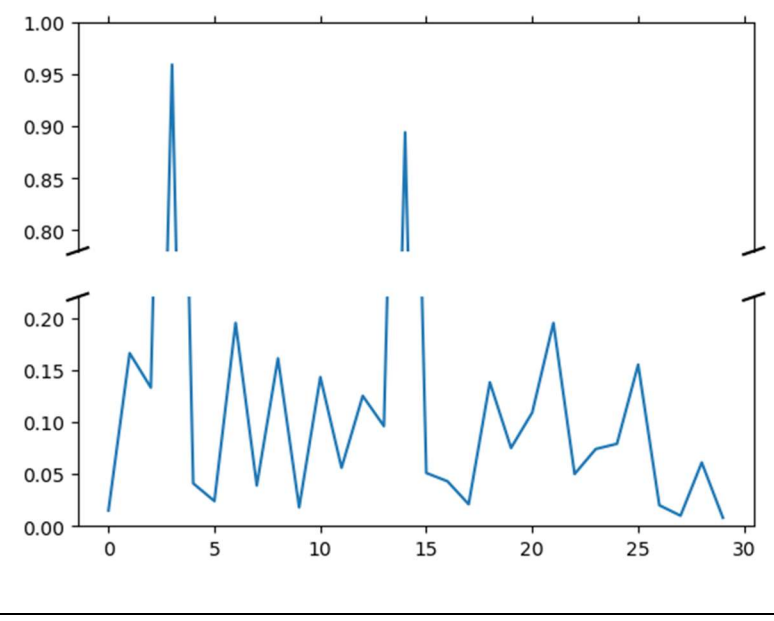
Task: Construct a line plot that displays all the values in the DataFrame. Due to the range of data values, employ two vertically stacked subplots (one for outliers and the other for the majority of data) with a shared x-axis to better visualize the distinctions between typical values and outliers within the dataset.

Style: The upper subplot should focus on the outliers by zooming into the upper range of the data values, whereas the lower subplot should cover the majority of the data by focusing on the lower range. Enhance clarity by hiding the intersecting spines between the two plots and use diagonal lines at the breaks to indicate the discontinuity in the y-axis. Use the same coloring and style for consistency in both subplots.

Generated



Ground truth

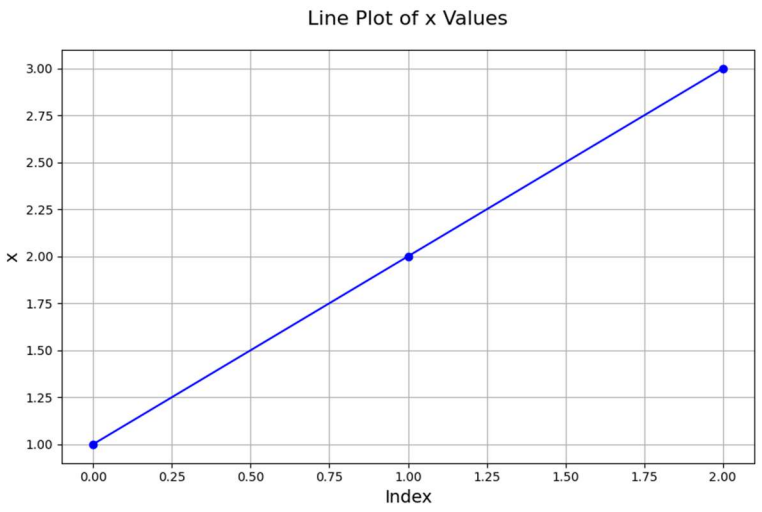


ID = 127
Score vis = 80
Score task = 100
Score human = 100

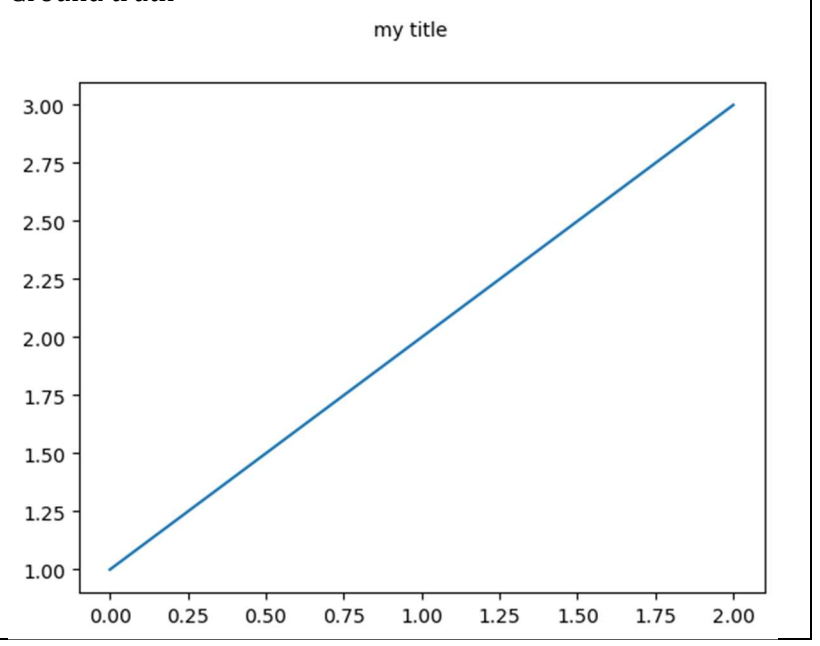
Task: The task involves creating a plot of the single column 'x' from the dataframe against its index. The plot will depict changes in 'x' values across the rows, leading to a line graph representation of these values.

Style: Customize the plot with a special figure title that is set at the top of the graph. The appearance of the plot should include this title centered above the graph, and the graphical representation itself should be clear and styled adequately to reflect trends in the data efficiently.

Generated



Ground truth

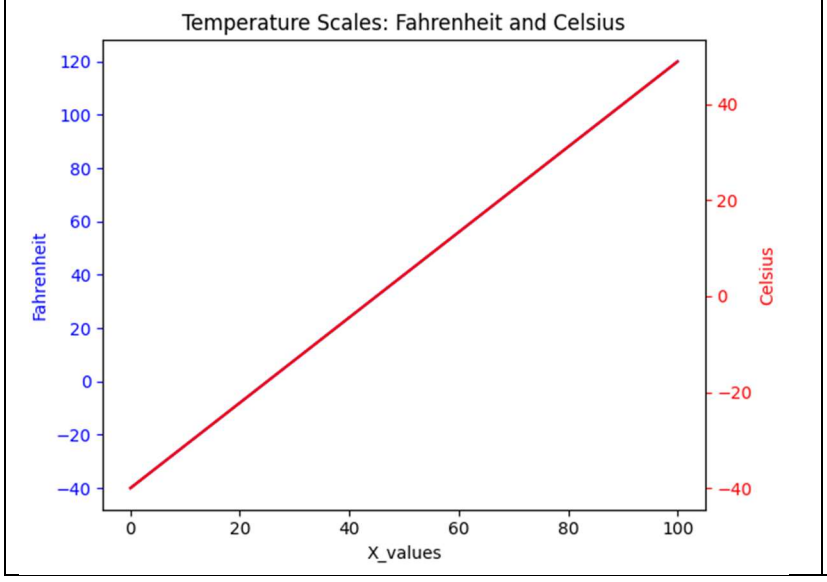


ID = 128
Score vis = 95
Score task = 95
Score human = 100

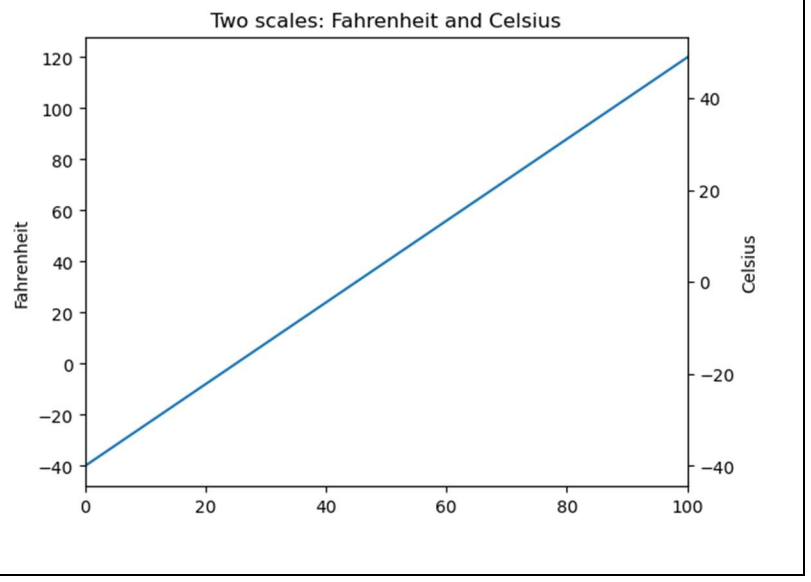
Task: The task involves generating a plot that has two vertical axes sharing the same horizontal axis. The primary vertical axis should display temperature values in Fahrenheit as derived from the DataFrame against the 'X_values'. The secondary vertical axis, which shares the same horizontal axis, should display corresponding temperatures in Celsius. An updating mechanism should be implemented so that changes in the scale of the Fahrenheit axis are reflected correctly on the Celsius axis.

Style: The plot should include labels for both axes and a title indicating the use of two different temperature scales. The range of the horizontal axis should be fixed to encompass all 'X_values' from the DataFrame. The visual style should ensure clarity and distinction between the two scales, possibly using different colors or line styles, and update interactively reflecting changes in axis limits or data updates.

Generated



Ground truth

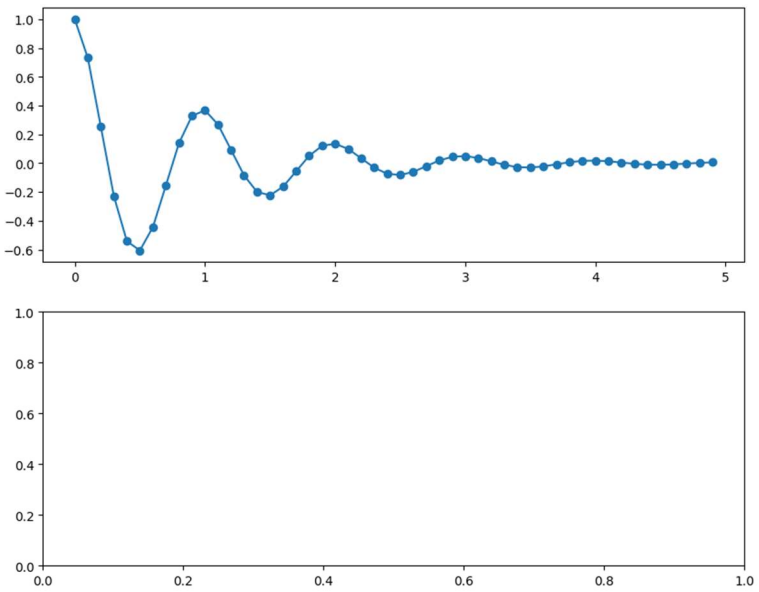


ID = 129
Score vis = 50
Score task = 30
Score human = 50

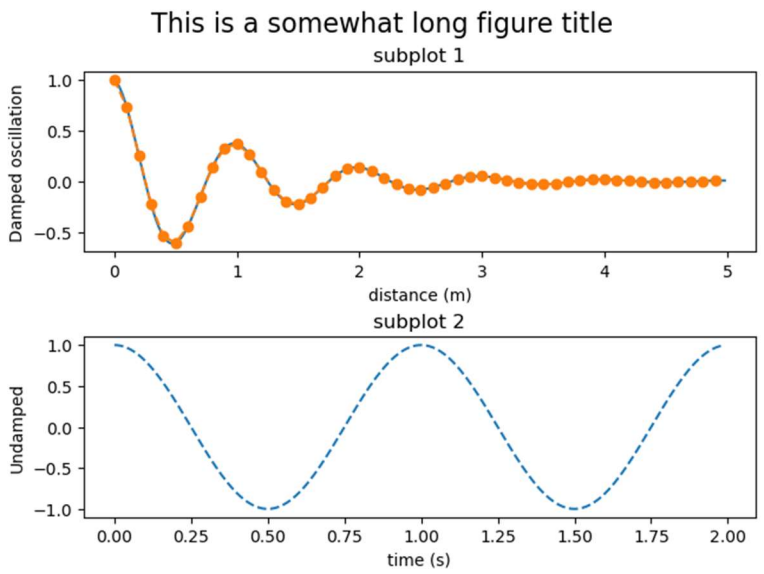
Task: The code should generate a figure with two subplots, distinguished by their respective labels. For each subplot, the data should be plotted with a line style and marker type according to the 'type' column. Each subplot should visualize how 'ft' evolves with respect to 't', arranged according to the label categories.

Style: he two subplots individually based on their labels. Each subplot should have a title and axes labels specified accordingly. The whole plot should have a main title, with appropriate font size, and must ensure clear spacing and alignment of subplots for visual clarity. The plot style should use different line styles and markers as mentioned in the DataFrame's 'type' column.

Generated



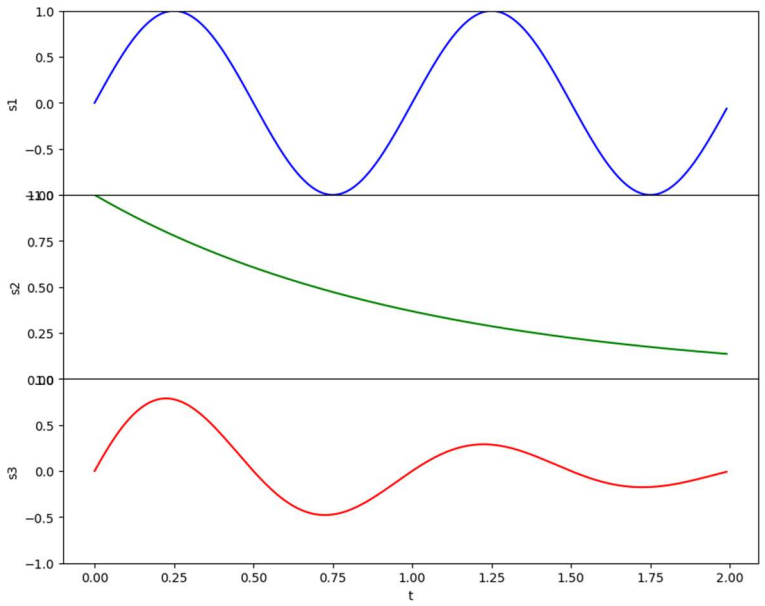
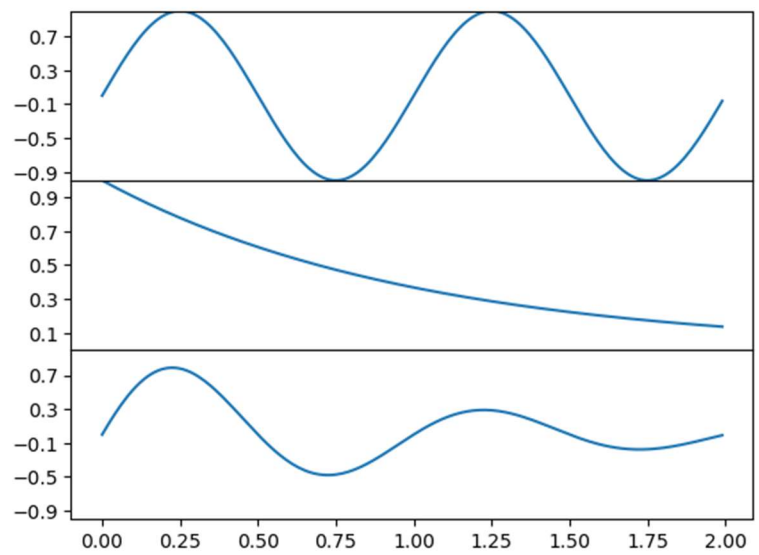
Ground truth



ID = 130
Score vis = 90
Score task = 90
Score human = 100

Task: Generate a plot that comprises three subplots arranged in a vertical stack, sharing the same x-axis. Each subplot will depict one of the signal columns ('s1', 's2', 's3') against 't'. The first and third subplots should have a y-axis limit set to range from -1 to 1, while the second subplot should have a y-axis limit ranging from 0 to 1. Each subplot needs to have its y-axis ticks manually defined for specific tick values.

Style: All subplots should be displayed in a single column with no horizontal spacing between them, allowing for a clean presentation of the data curves. The y-axis ticks should be strategically set to clearly demarcate the ranges and intervals, enhancing the readability of distinct values in the plots. The overall aesthetic should aim for clarity and simplicity, facilitating quick visual interpretation of the data trends.

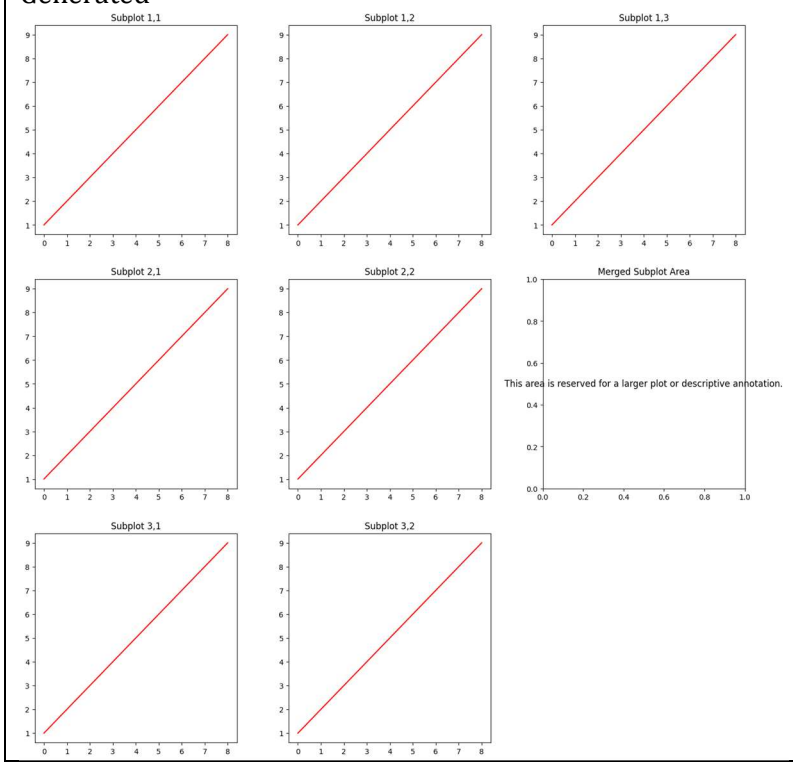
Generated	Ground truth
	

ID = 131
Score vis = 80
Score task = 100
Score human = 30

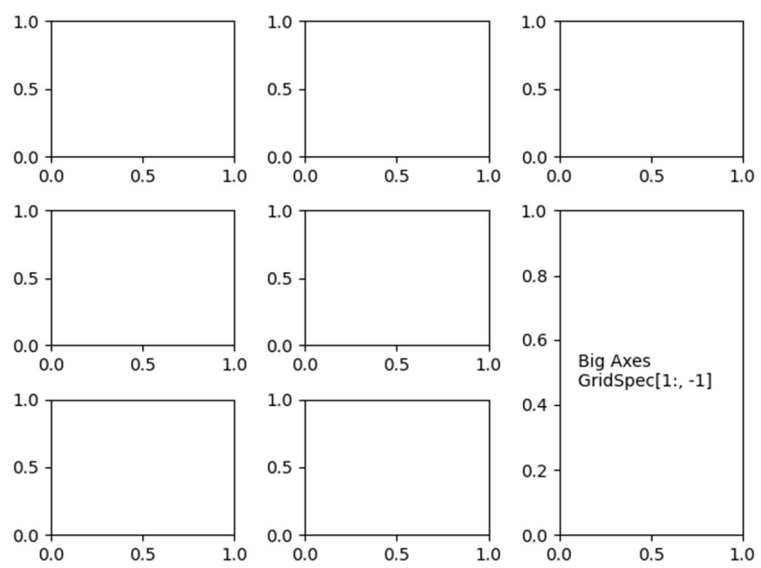
Task: description: Construct a subplot grid layout of 3x3 within a single figure. Modify this subplot to merge the last column of the lower two rows into a single larger plotting area where a descriptive annotation is placed. The code should create placeholders within this grid possibly for future data plotting.

Style: description: Adjust the layout of the plot to ensure there is an appropriate amount of spacing between subplots for clarity and non-overlapping of subplot elements. Utilize an annotation within the large subplot area to describe the function or purpose of this larger space. The final plot should visually align and be clear to interpret.

Generated



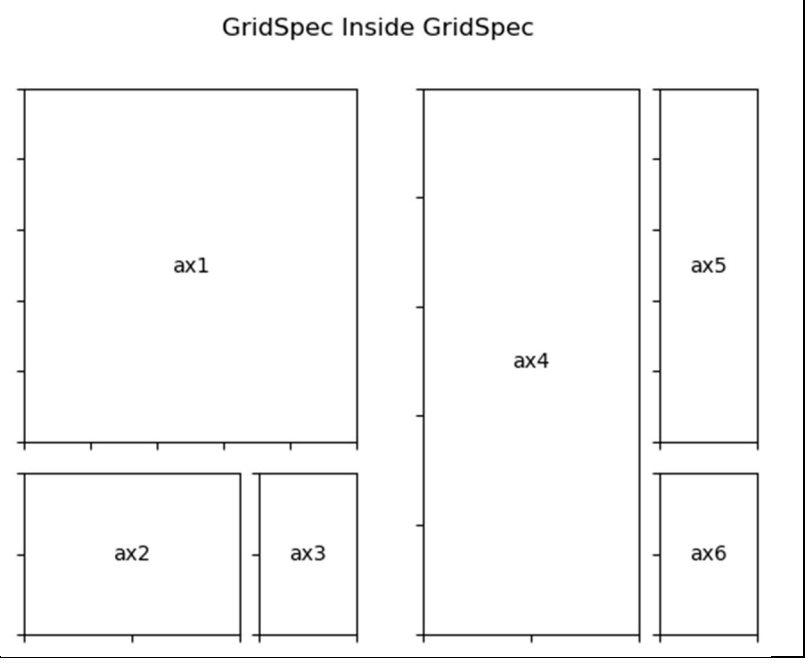
Ground truth



ID = 132
Score vis = 0
Score task = 0
Score human = 0

Task: The plotting task involves creating a complex multi-panel figure where different subplots (six in total) are arranged within a primary grid layout. The main grid layout is divided into two major sections, each containing a nested grid layout. These nested grids encompass multiple subplots arranged in various configurations (horizontal, vertical, mixed) and sizes.

Style: Each subplot is meant to be devoid of typical axis labels and tick marks, emphasizing a clean and clear presentation focused on the placement of text annotations at specified locations within each subplot. Additionally, text annotations provide clear identifiers (e.g., 'ax1', 'ax2', etc.) for each subplot, placed centrally within their respective subplots according to the 'x' and 'y' values from the DataFrame. The overall figure should have a title that succinctly describes the plot's structural theme.

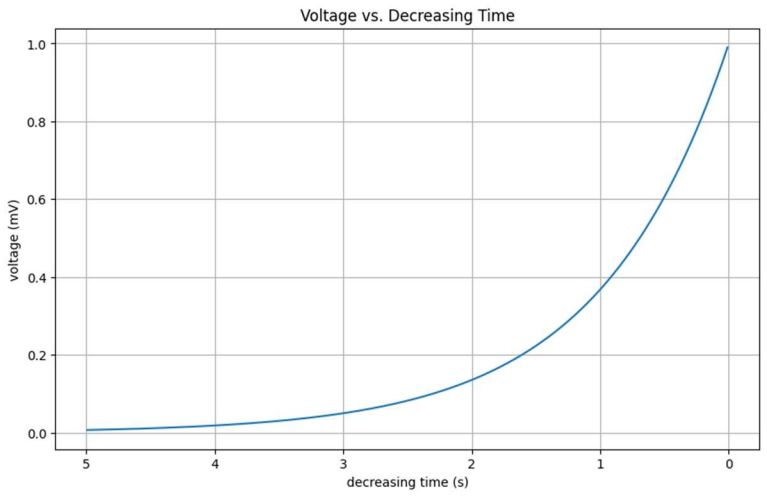
Generated	Ground truth
	

ID = 133
Score vis = 95
Score task = 100
Score human = 100

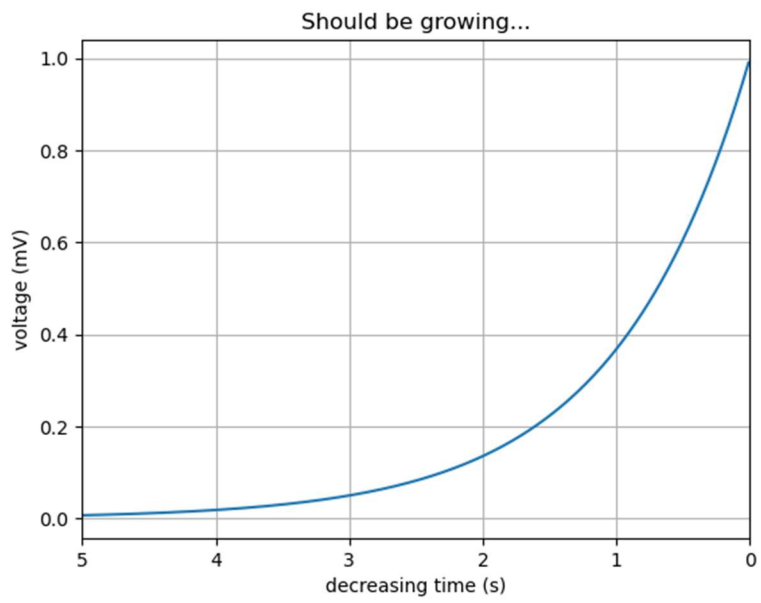
Task: Create a line plot displaying 'voltage' (y-axis) against 'time' (x-axis). However, the 'time' axis should appear in decreasing order, representing time flowing backward. This plot will visualize how voltage changes over a reversed time sequence. Include axis labels, a plot title, and enable the grid for better readability of the plot.

Style: The plot needs to be styled in a manner that time decreases from left to right (maximum to minimum). Label the 'x-axis' as "decreasing time (s)" and the 'y-axis' as "voltage (mV)". Set a title for the graph. The grid should be visible to aid in the visual interpretation of the plot points in relation to the scale.

Generated



Ground truth



ID = 134
Score vis = 95
Score task = 95
Score human = 100

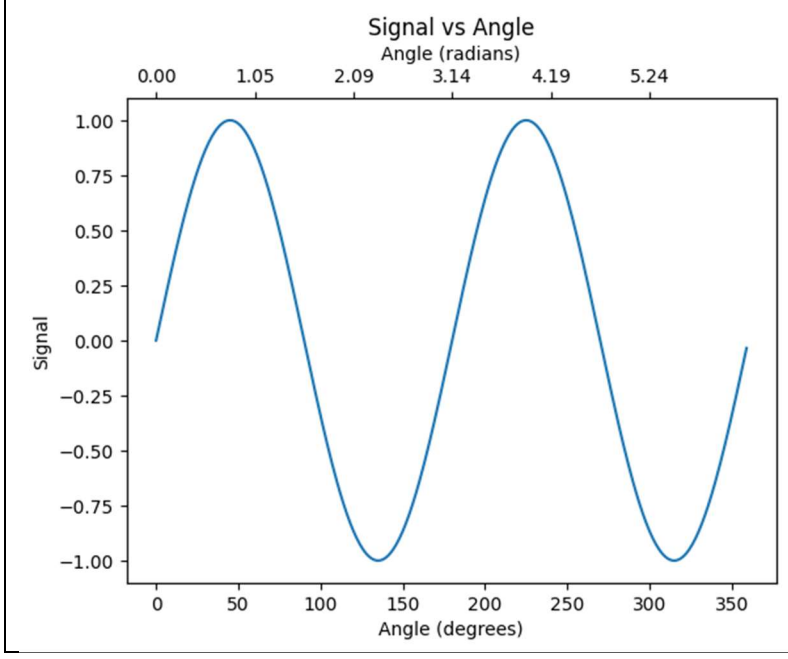
Task:

Create a line plot representing the relationship between the angle in degrees and the sinusoidal signal value. Enhance readability and interpretability by adding appropriate labels to both axes, a title to the plot, and displaying a secondary x-axis that translates the angle measurements from degrees to radians.

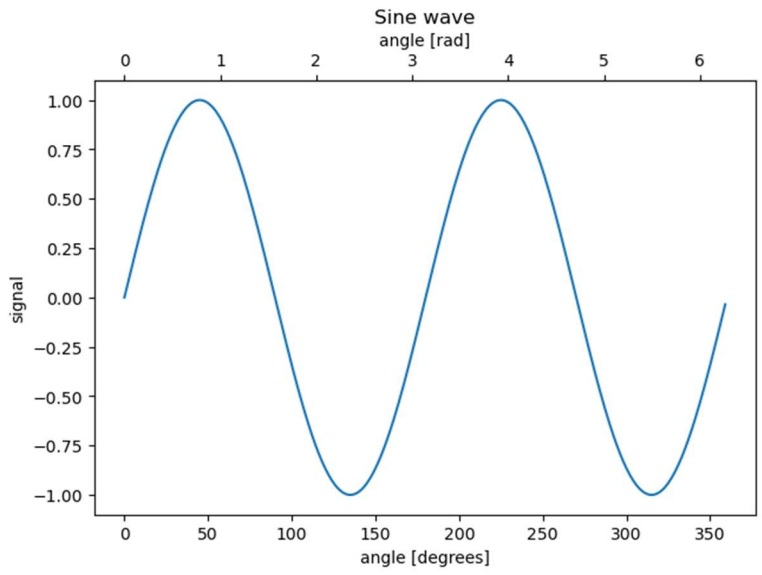
Style:

Utilize a clear and uncluttered style for the plot, ensuring the main and secondary axes are properly labeled to distinguish between degrees and radians. The main x-axis should show the angle in degrees, while the top x-axis should display corresponding values in radians. Overall design should maintain simplicity to focus on the wave pattern visualization.

Generated



Ground truth

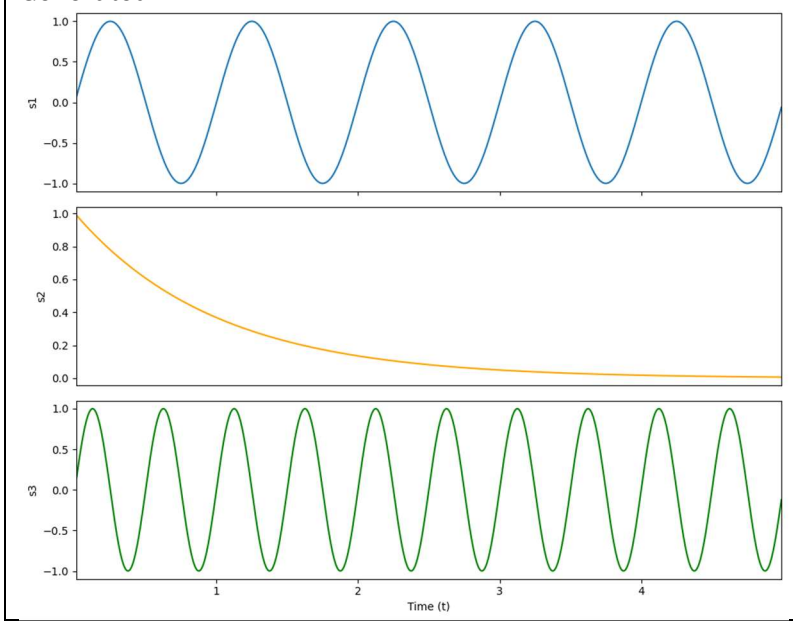


ID = 135
Score vis = 95
Score task = 95
Score human = 100

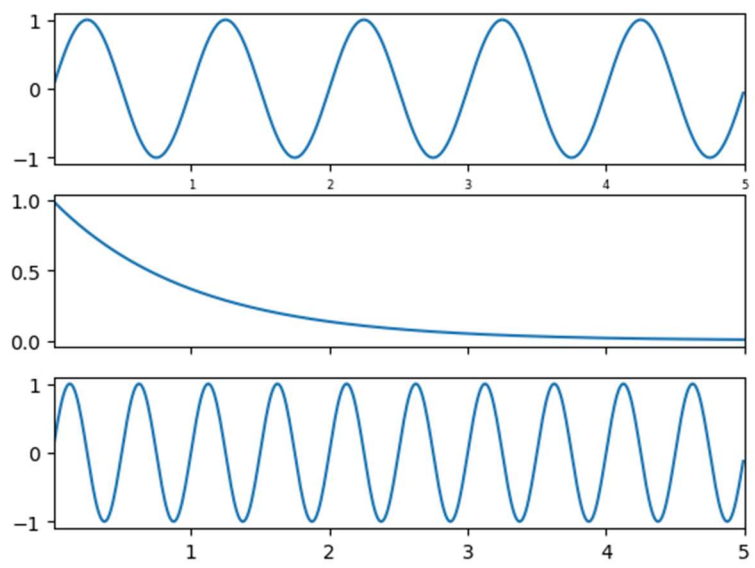
Task: The task involves plotting three subplots vertically aligned. Each subplot represents one of the signal columns (s1, s2, s3) plotted against the time column (t). The x-axis (time) should be shared among all plots, but only the bottom plot will display x-axis labels. The first and third plots will share the y-axis scale as well.

Style: Customize the font size of the x-axis tick labels in the first plot for clarity. For the middle plot, the x-axis tick labels should be hidden to maintain a clean visual relationship between the plots. The x-axis range should be fixed across all subplots to ensure a consistent scale.

Generated



Ground truth

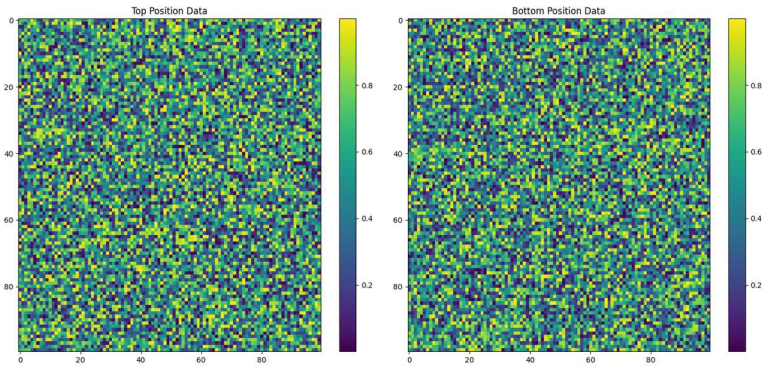


ID = 136
Score vis = 70
Score task = 95
Score human = 100

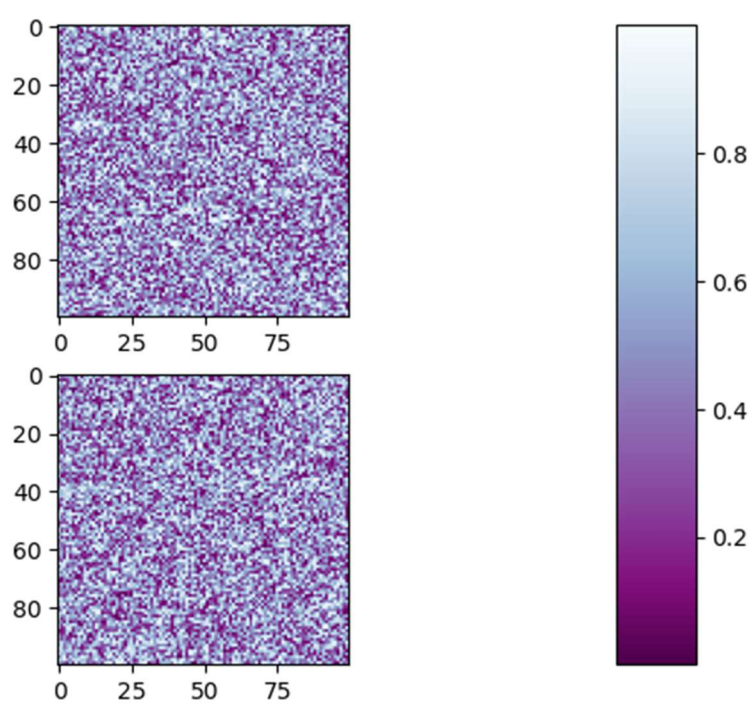
Task: Create two subplots, where each subplot visualizes numerical data from rows corresponding to either 'Top' or 'Bottom' as specified in the 'Position' column. Extract and drop unnecessary columns ('Position' and 'index') for the visualization. Each subplot should depict the data as an image with a color mapping scale.

Style: Use a sequential color map to differentiate the data values clearly in the images of the subplots. Ensure that the color intensity reflects the value magnitude, making higher values more prominent. Adjust the layout to prevent any overlap between subplots and include a color bar on the side to represent the data value scale effectively, enhancing interpretability.

Generated



Ground truth

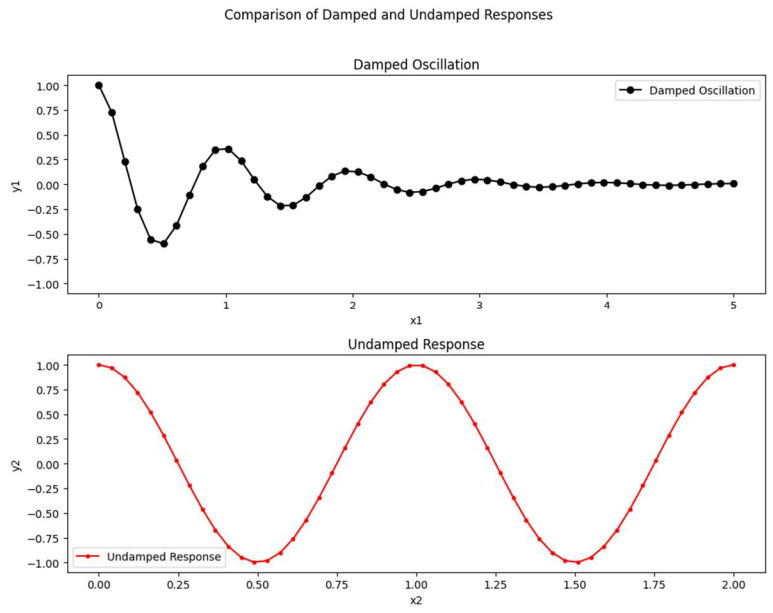


ID = 137
Score vis = 90
Score task = 100
Score human = 100

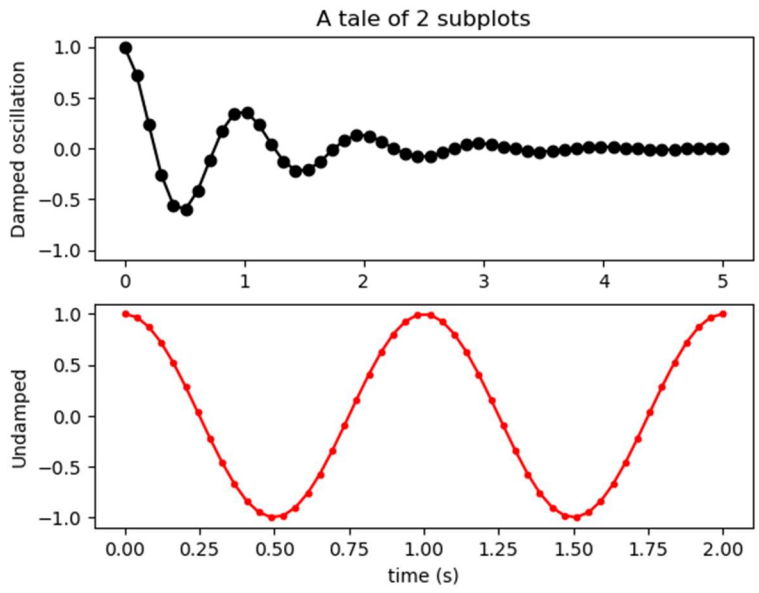
Task: Create two subplots vertically aligned, sharing the same y-axis scale. The first subplot should plot the values from columns 'x1' and 'y1' as a black line with circle markers. The second subplot should plot the values from columns 'x2' and 'y2' as a red line with dot markers. Align the axes so that they are appropriate for the data scales and label the axes according to the data they represent.

Style: Adjust the style of the plot to have titles for each subplot to distinguish between the datasets. The first plot should be labeled to indicate it represents a damped oscillation, and the second plot should indicate an undamped response. Ensure the overall figure has a title that covers the thematic representation of the dual plots, and format the plots for clear visibility and differentiation between the two data presentations.

Generated



Ground truth

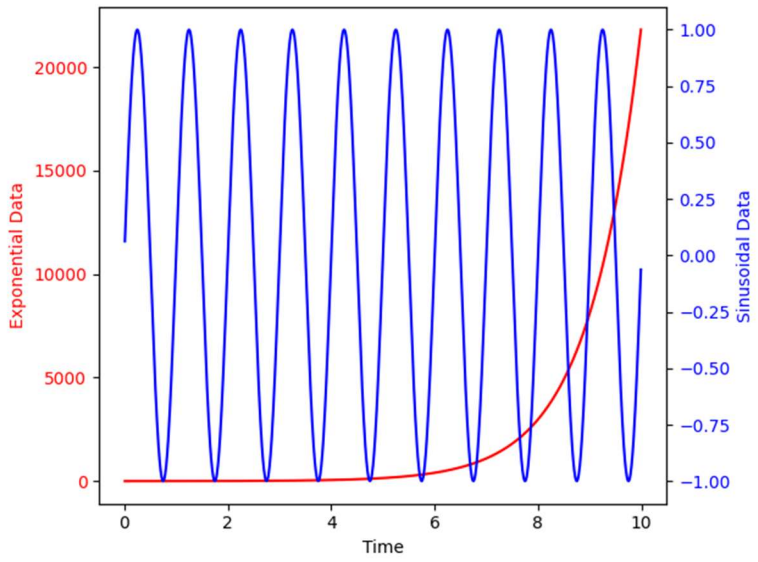


ID = 138
Score vis = 95
Score task = 95
Score human = 100

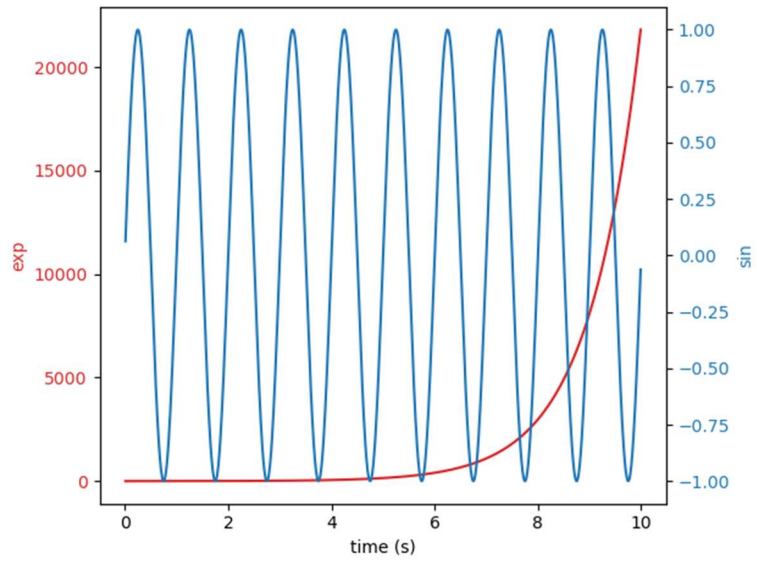
Task: The script should plot two datasets on a common x-axis with different y-axes. The 'time' column should be set as the x-axis. The first dataset, 'exp_data', should be plotted with its scale and labeled on the left y-axis. The second dataset, 'sin_data', should be on a second y-axis aligned to the right with its own scale. Both should be clearly distinguishable via different colors and include appropriate labels for each axis.

Style: The exponential data ('exp_data') should use a warm color like red and the sinusoidal data ('sin_data') should use a cool color like blue to ensure visual distinction. Each axis should have labels matching the color of the corresponding dataset. The layout must be tight so that all parts of the plot, especially the right y-label, are clearly visible.

Generated



Ground truth

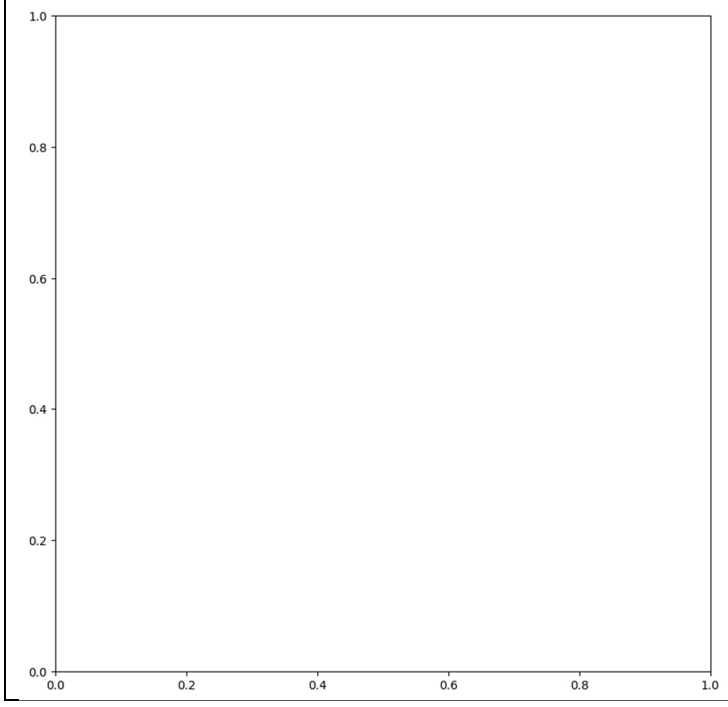


ID = 139
Score vis = 0
Score task = 0
Score human = 0

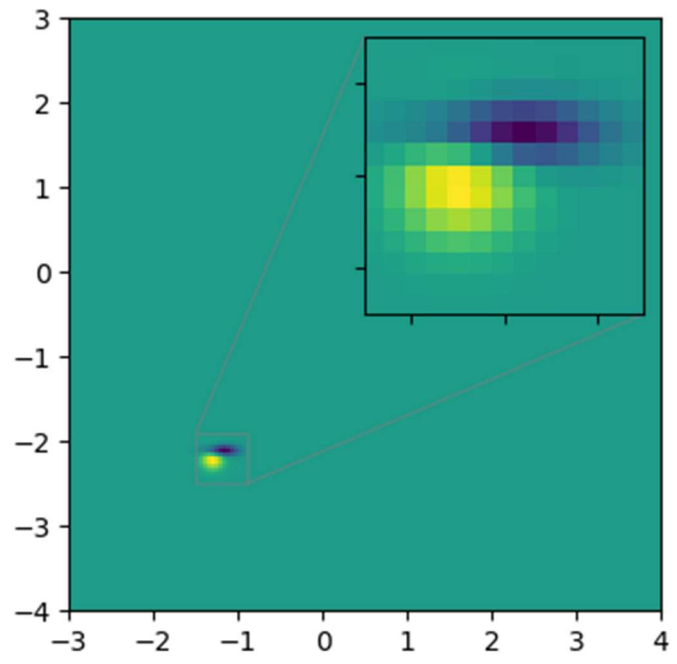
Task: description: Create a main plot displaying the data of the dataframe as an image. Utilize a predefined extent to set the boundary of the visual data representation on this main plot area. Add a zoomed-in inset plot within the main plot to highlight a specific region of interest in the larger dataset. Ensure that this inset plot is placed appropriately within the main plot, and dynamically adjusts the view to this specific region without displaying axis tick labels.

Style: description: Use specific styling configurations to improve the visual clarity and presentation of the plots. Ensure that the plots use nearest interpolation and have their origin set to lower to maintain grid alignment. Ensure that the transition between the main plot and inset plot is clearly indicated with linked lines to guide viewers, and use aesthetically pleasing color schemes suitable for image display. Adjust the figure size and aspect ratios appropriately to effectively communicate the data insights.

Generated



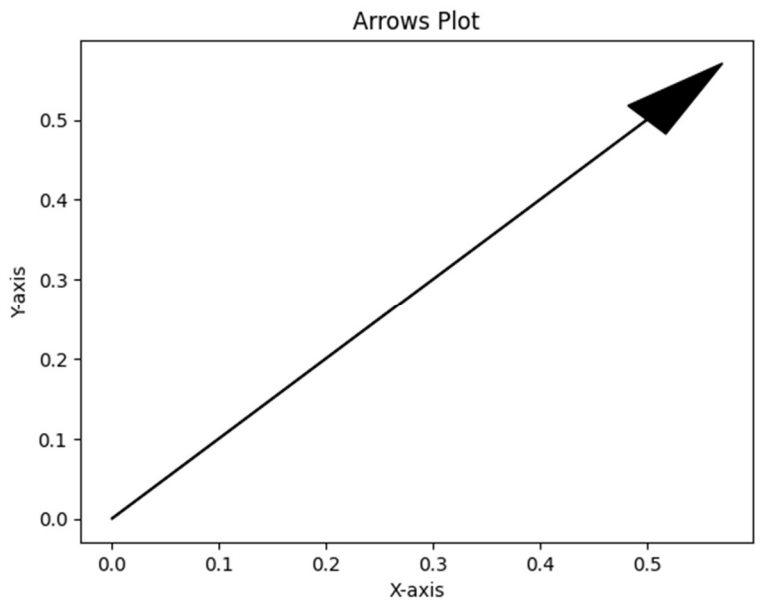
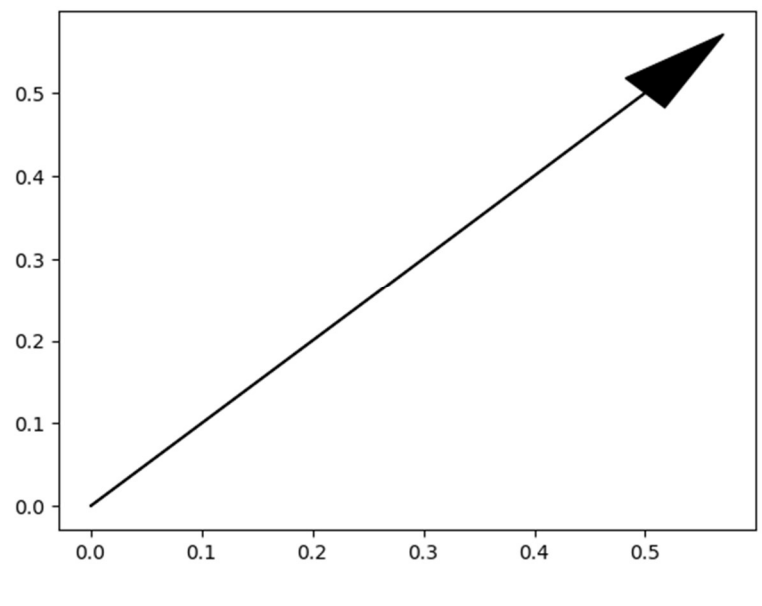
Ground truth



ID = 140
Score vis = 95
Score task = 90
Score human = 100

Task: Create a plot where each row from the dataframe is represented as an arrow on a 2D graph. These arrows are drawn from a starting point (start_x, start_y) to an ending point (stop_x, stop_y) with specified properties for the head's width, length, and color.

Style: Format the plot with aesthetic features, including specifying the arrow color, edge color, and adjusting the sizes of the arrowhead. The plot should provide a clear visualization of the directional movement between two coordinates.

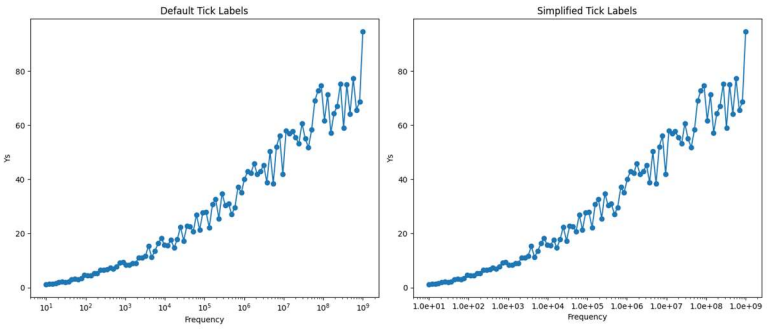
Generated	Ground truth
	

ID = 141
Score vis = 60
Score task = 100
Score human = 100

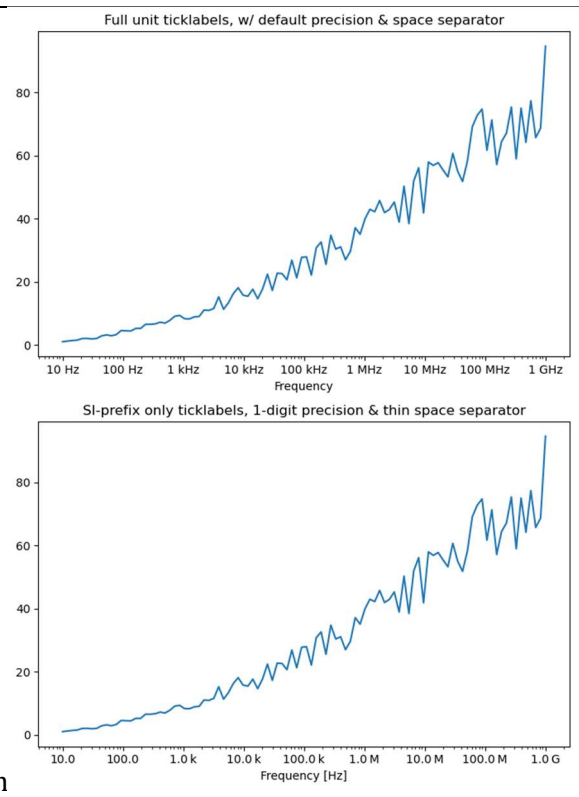
Task: Generate a side-by-side visualization of two subplots, both sharing a similar dataset. Each plot should display frequency on a logarithmic x-scale versus 'Ys' on the y-axis. The subplot titles should indicate different settings, such as formatting preferences for tick labels on the x-axis.

Style: Adjust the style of each subplot to differentiate their x-axis tick labels. The first plot should show full unit tick labels with default precision and formatting, while the second plot should feature simplified tick labels with controlled digit precision and a non-standard numeric separation. Set layout options to make the visual display appealing and comprehensive. Use tight layout parameters for clean spacing between plots.

Generated



Ground truth

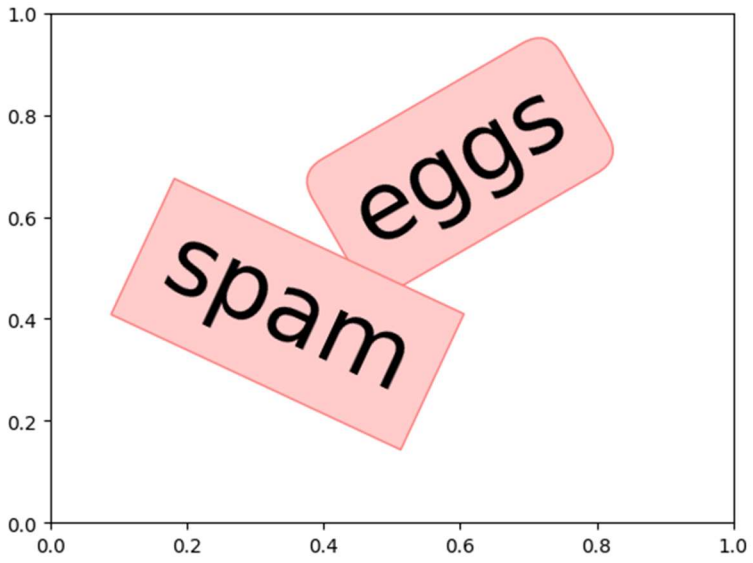


ID = 142
Score vis = 100
Score task = 100
Score human = 100

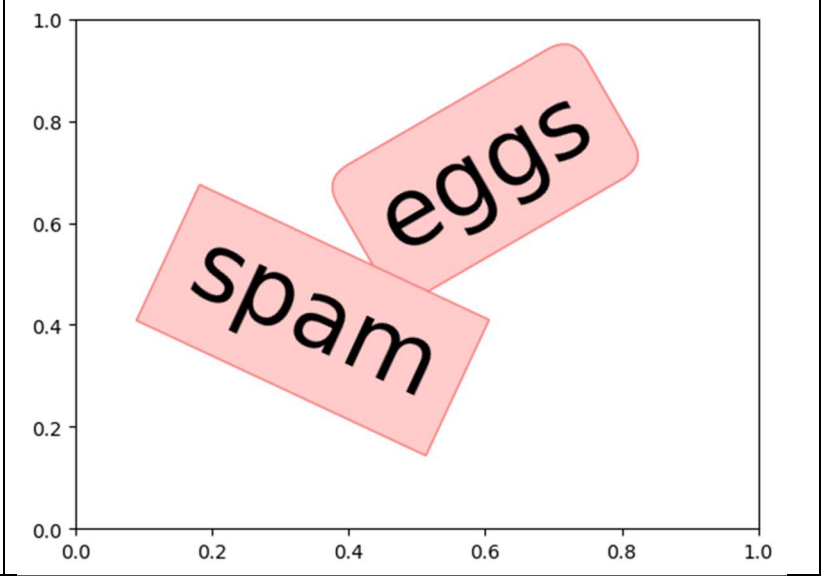
Task: Iterate through each row of the dataframe to plot text at specified 'x' and 'y' coordinates. Each text element uses attributes from the dataframe for size, rotation, horizontal alignment, vertical alignment, and a bounding box with custom style, edge color, and fill color.

Style: Text elements are graphically enhanced with a bounding box that has a specified box style. The edges and filling of these boxes are colored using predefined RGB tuples, creating a visually distinctive style against the plot background.

Generated



Ground truth

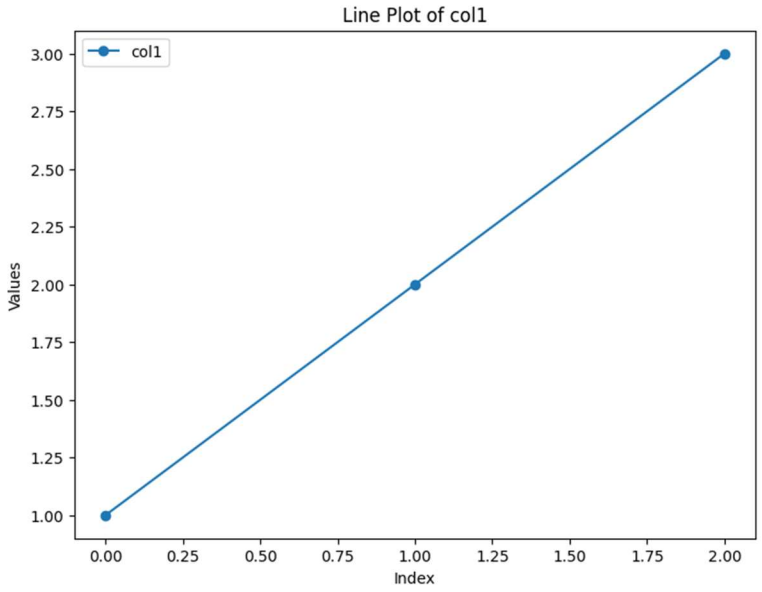


ID = 143
Score vis = 95
Score task = 90
Score human = 100

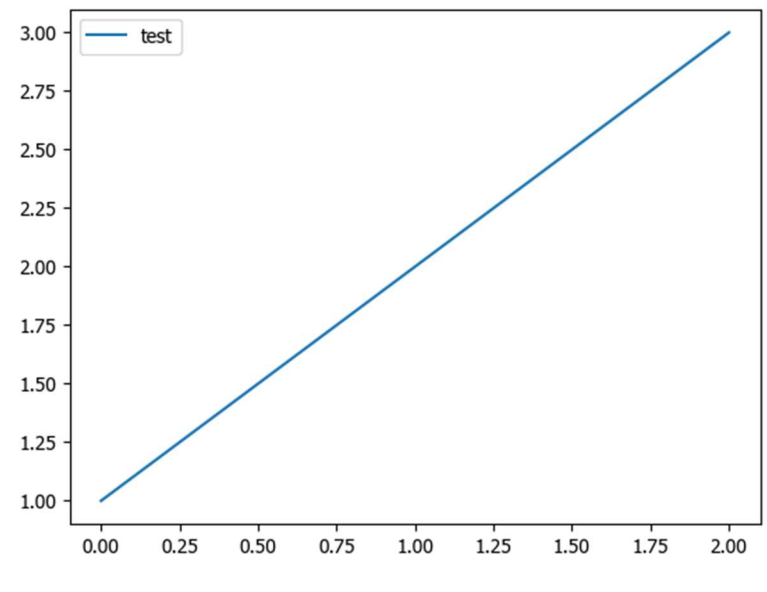
Task: Create a line plot of the data in the DataFrame. The x-axis should represent the index of the DataFrame automatically, and the y-axis should plot the values from 'col1'. Include a legend to identify the plotted line.

Style: he font family of the plot text to a sans-serif type, utilizing 'Tahoma' as the specific sans-serif font. The plot should have clean and simple aesthetics to emphasize the data visualization effectively.

Generated



Ground truth

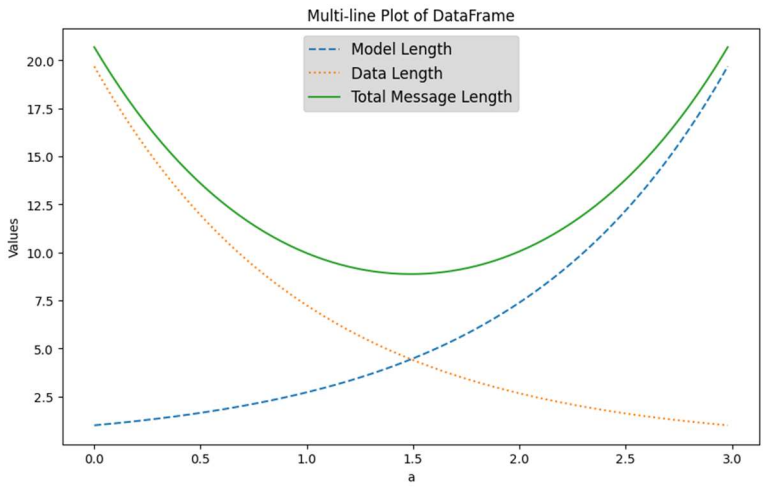


ID = 144
Score vis = 95
Score task = 100
Score human = 100

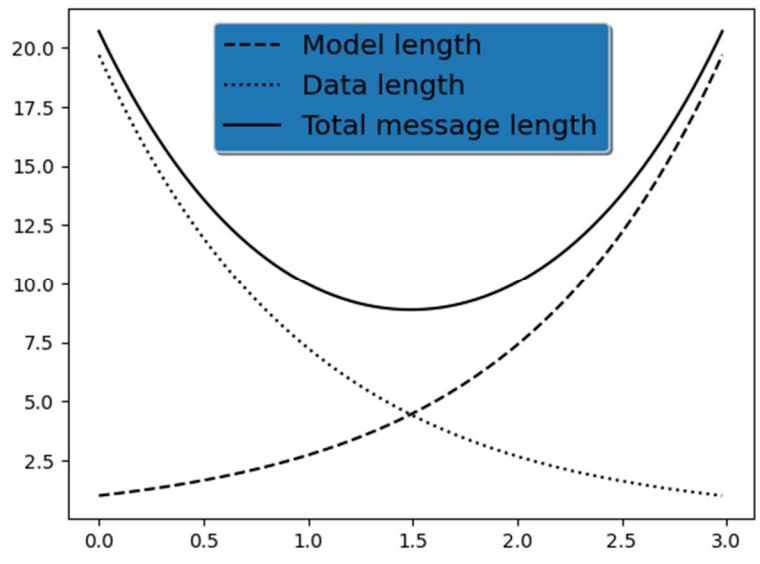
Task: Generate a multi-line plot from the given DataFrame with three separate lines representing different sets of data ('c', 'd', and 'tot') plotted against the values in column 'a'. Include legends for each line, reflecting each corresponding aspect being represented (model length, data length, and total message length).

Style: he plot such that each data series is uniquely styled (dashed, dotted, and solid lines). Enhance the readability of the plot using a large font size in the legend and differentiate the legend background color to improve visual appeal.

Generated



Ground truth

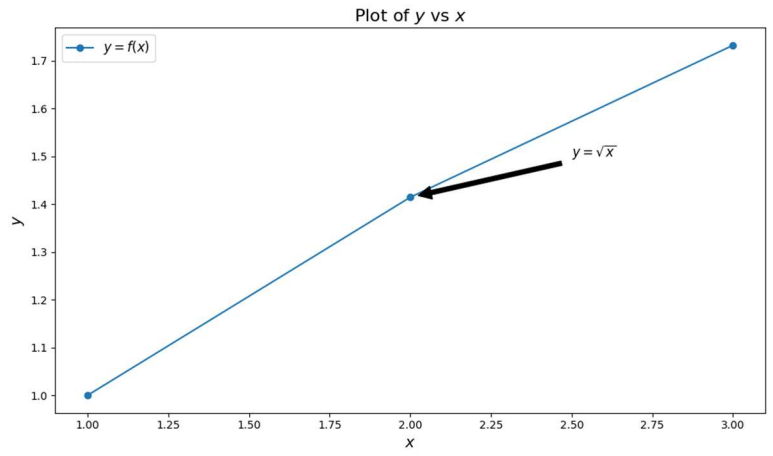


ID = 145
Score vis = 60
Score task = 95
Score human = 100

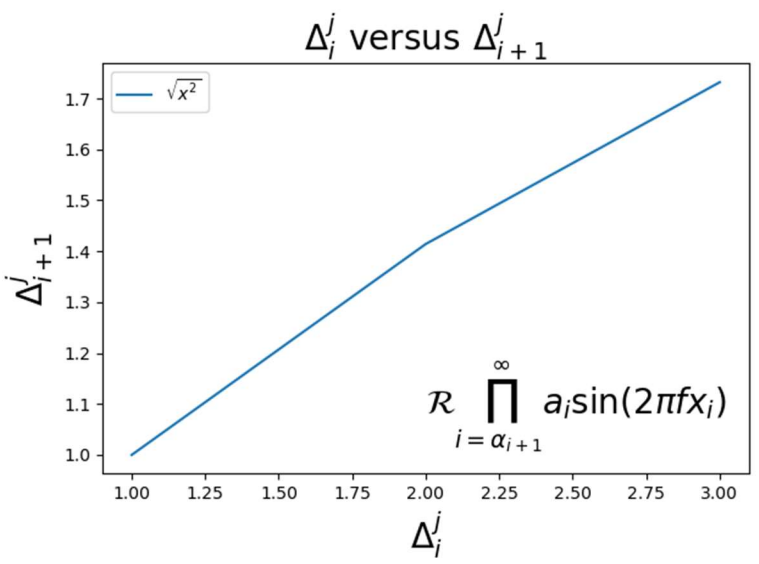
Task: A 2D line plot needs to be created from the DataFrame. The x-axis will represent the 'x' column and the y-axis will represent the 'y' column, with a line connecting these points. Include legend representation for the curve, and descriptive labels and title for the axis and the plot, employing mathematical notation where applicable.

Style: Enhance the plot aesthetics by using larger font sizes for axis labels and the title to ensure better readability. Mathematical text should be rendered using a TeX-like format for symbols and expressions. Additionally, incorporate a textual annotation within the plot area, using similar styled mathematical expression. Adjust layout to prevent clipping of text or labels.

Generated



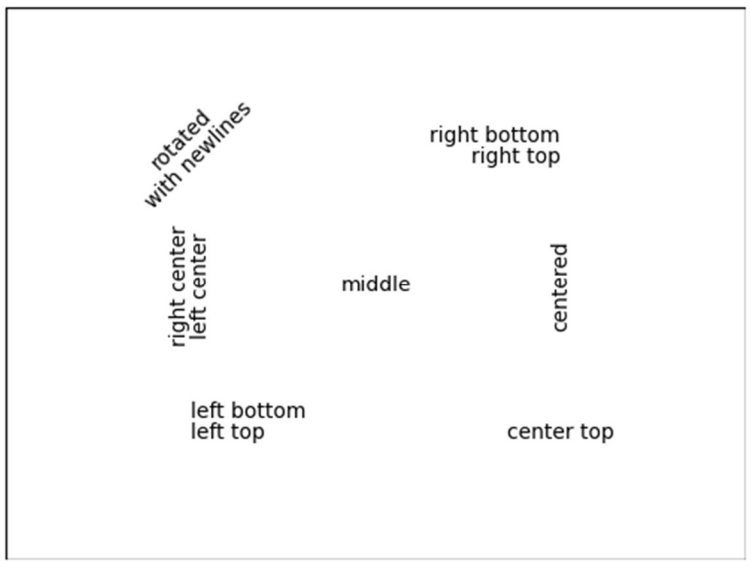
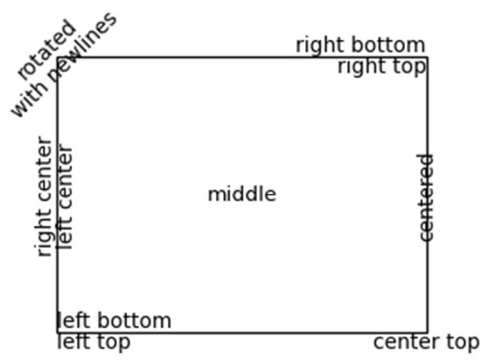
Ground truth



ID = 146
Score vis = 90
Score task = 90
Score human = 75

Task: The code aims to plot annotated text at specified positions onto a graph. Initially, a rectangle is defined and added as a non-filled shape within the axes coordinates of the plot to serve as a background or reference. Then, looping over each row in the DataFrame, text annotations are created and added at the specified (x, y) positions with particular alignment and rotation settings. The text is drawn using coordinates transformed to match the axes scaling.

Style: The plot style is designed to emphasize annotations without displaying axes, ticks, or other graph elements. The plot area will have annotations aligned as specified in the DataFrame, with the main focus being on the clarity and positioning of the text. The presence of a rectangle outline helps to visually organize the text annotations while maintaining a clean and uncluttered graphical output.

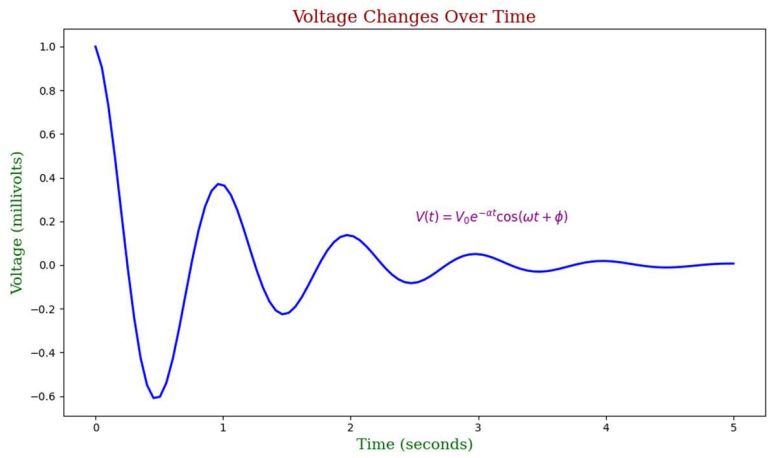
Generated	Ground truth
	

ID = 147
Score vis = 90
Score task = 100
Score human = 100

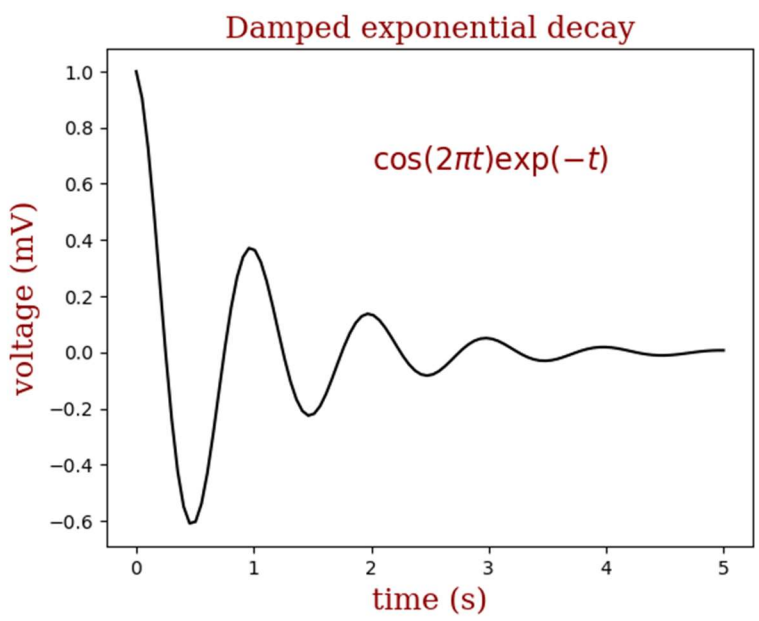
Task: Create a line plot representing voltage changes over time. Add a title at the top of the plot and annotate a specific formula directly on the plot graph area. Label each axis to indicate which physical quantities they represent (e.g., time in seconds, voltage in millivolts).

Style: Customize the title, axis labels, and formula annotation to have a specific font family, color, and size. Adjust the layout to ensure no part of the y-axis label is clipped off. Use a standard color and line style for the plot curve.

Generated



Ground truth

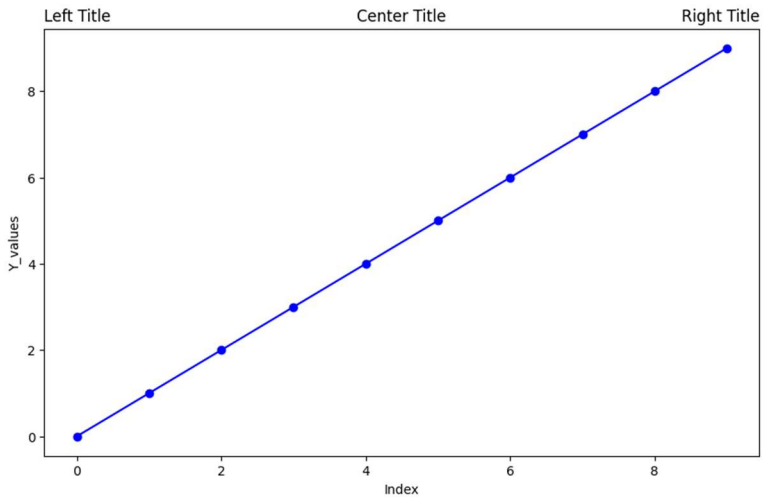


ID = 148
Score vis = 90
Score task = 100
Score human = 100

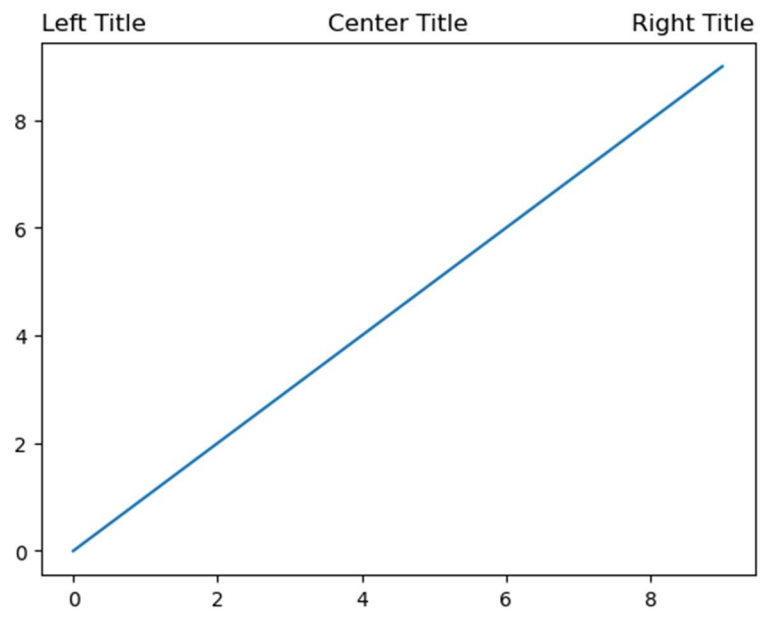
Task: Create a line plot using the 'Y_values' from the dataframe. The plot should feature a line graph representing these values against their corresponding index on the x-axis. Additionally, three titles need to be added in different locations on the plot—left, center, and right.

Style: he plot by adding titles at specified locations across the top of the plot. Each title should be placed distinctly at the left, center, and right alignments, helping to enhance the visual layout and emphasize different textual elements. The line in the plot should be clear and easily distinguishable, with the axes properly labeled to reflect the indices and the corresponding 'Y_values'.

Generated



Ground truth

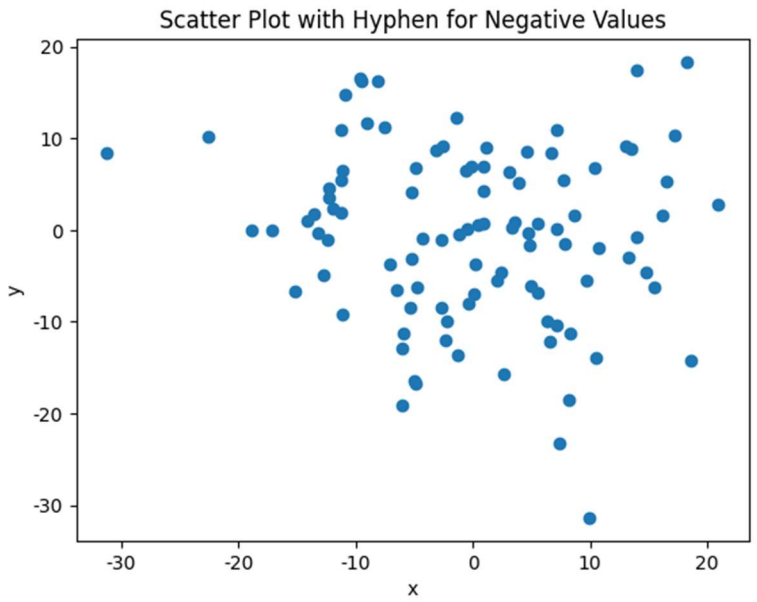


ID = 149
Score vis = 95
Score task = 95
Score human = 100

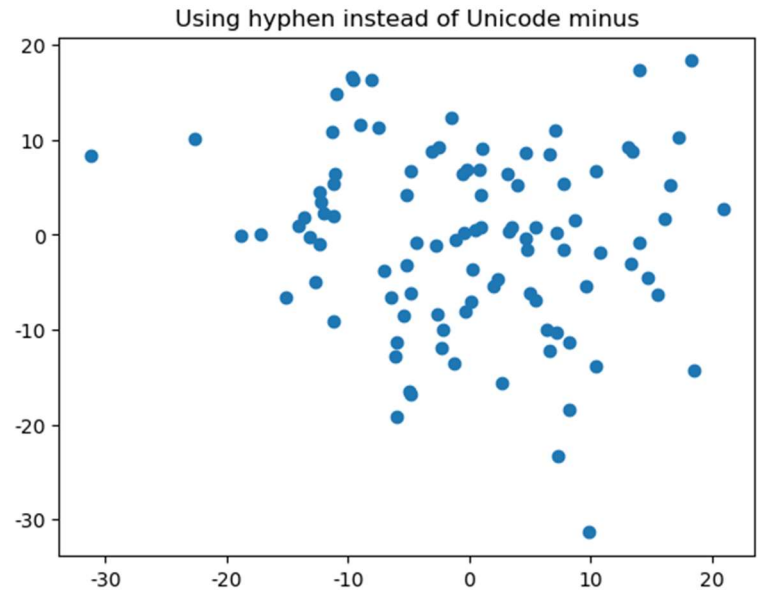
Task: Generate a scatter plot using the values in the 'x' and 'y' columns of the DataFrame. Mark each data point with a circle. Configure the plot to use a standard hyphen for negative values instead of the Unicode minus symbol.

Style: Include a descriptive title for the plot that notifies about the usage of hyphen instead of Unicode minus. Set labels for the x and y axes if required. Keep the visual style simple and clean to emphasize the data and customization noted about negative values.

Generated



Ground truth

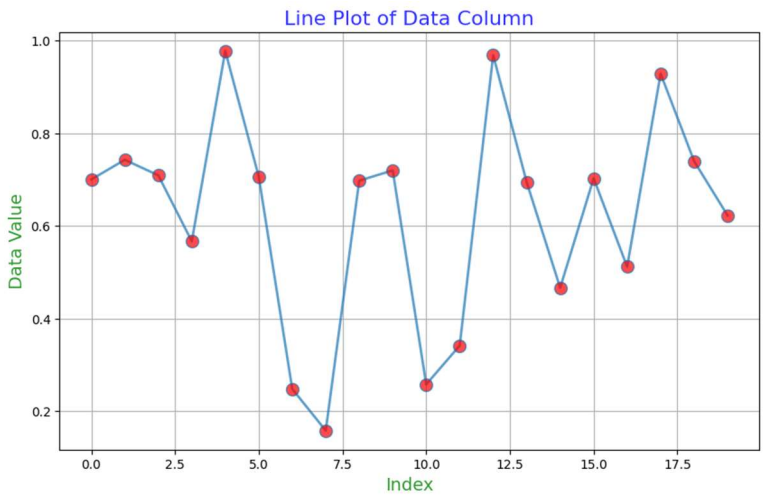


ID = 150
Score vis = 90
Score task = 95
Score human = 100

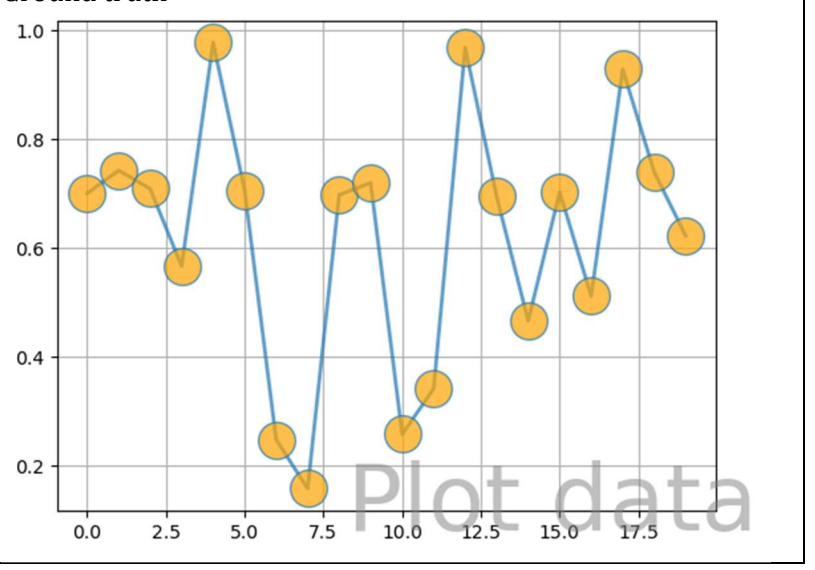
Task: Create a line plot representing the values in the 'data' column. The points on the graph should be distinctively marked. Additionally, enable grid lines for better readability of the plotted values.

Style: he plot by enlarging the markers on the line plot and setting their face color. Adjust the line width and the transparency of these elements for better visual appeal. Add descriptive text to the plot with styling adjustments such as size, color, alignment, and transparency.

Generated



Ground truth

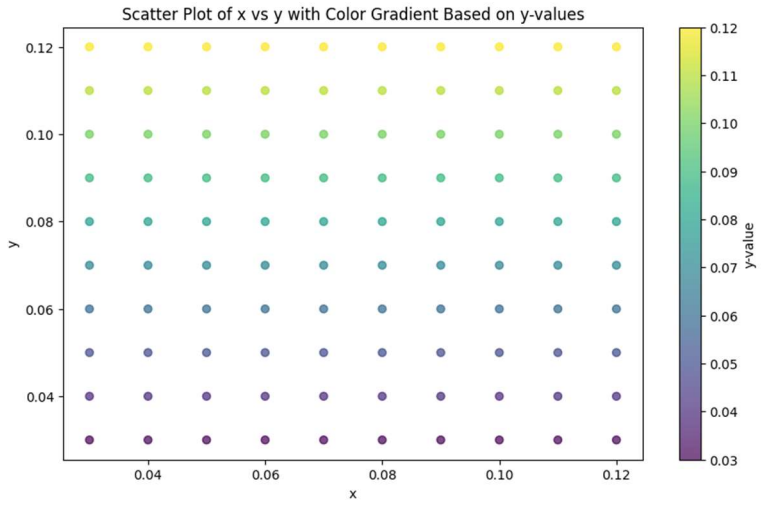


ID = 151
Score vis = 95
Score task = 100
Score human = 100

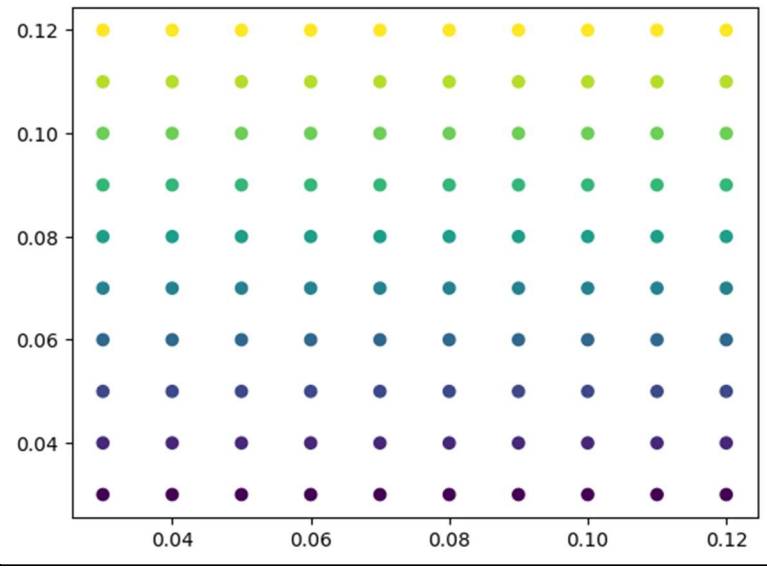
Task: Generate a scatter plot with the DataFrame's x-values on the horizontal axis and y-values on the vertical axis. Use the y-values to determine the color of each point in the scatter plot to visualize a gradient based on the y-value size.

Style: Adjust the color intensity or hue based on the value of y to represent data depth and distribution visually. The plot should clearly differentiate data clusters or variations by using this color encoding, modifying visual aspects as necessary to enhance clarity and visual appeal.

Generated



Ground truth

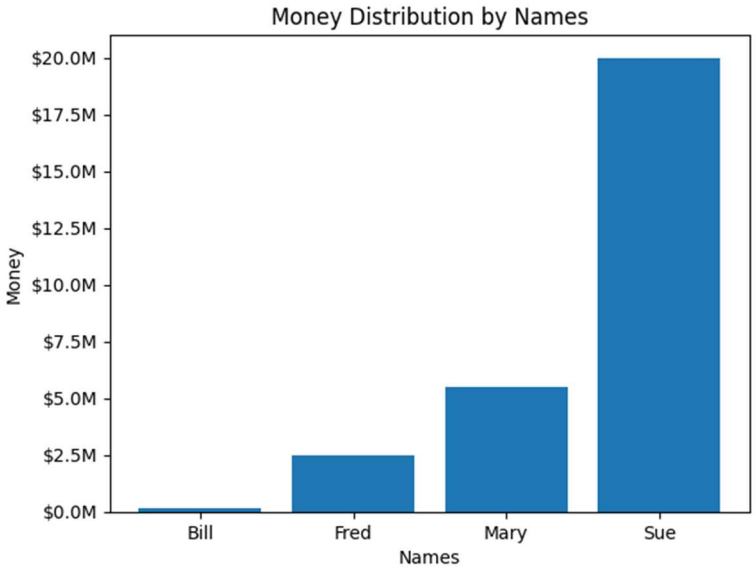


ID = 152
Score vis = 95
Score task = 90
Score human = 100

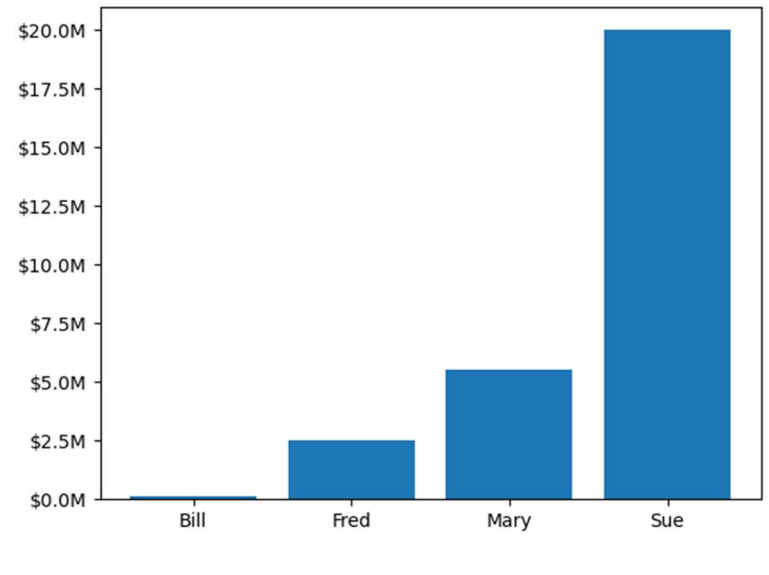
Task: Create a bar chart where the x-axis represents the categories from the 'names' column and the y-axis represents the monetary values from the 'money' column. Format the y-axis labels to display monetary values in terms of millions of dollars.

Style: Use a clear and simple style for the bar chart. The y-axis values should be formatted to read as monetary amounts (e.g. $2.0M, $5.0M), enhancing readability and context for the user. Ensure the plot visually distinguishes between different names for immediate comprehension of the data comparison.

Generated



Ground truth

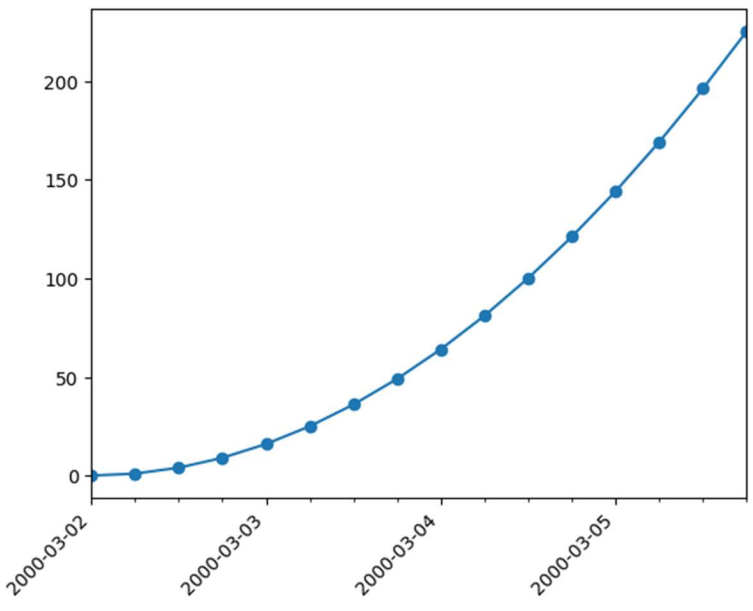


ID = 153
Score vis = 90
Score task = 90
Score human = 100

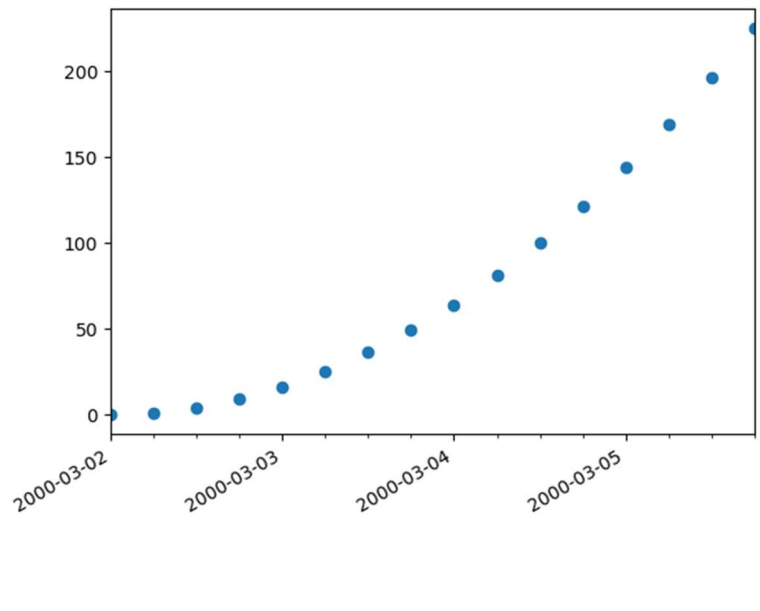
Task: Create a line plot where the x-axis represents dates and times, and the y-axis represents numerical values. Set the limits of the x-axis to the range of the dates in the dataframe to ensure all points are displayed. Utilize major and minor locators to specify major ticks on the x-axis per day and minor ticks every few hours.

Style: Configure the date format on the x-axis to show full dates in year-month-day format and set the format for tooltip display on hover or selection to include precise timing in year-month-day hour-minute-second format. Rotate the date labels to improve readability. Ensure the plot is clear and the axis scales are appropriately set for easy interpretation of timeline trends.

Generated



Ground truth

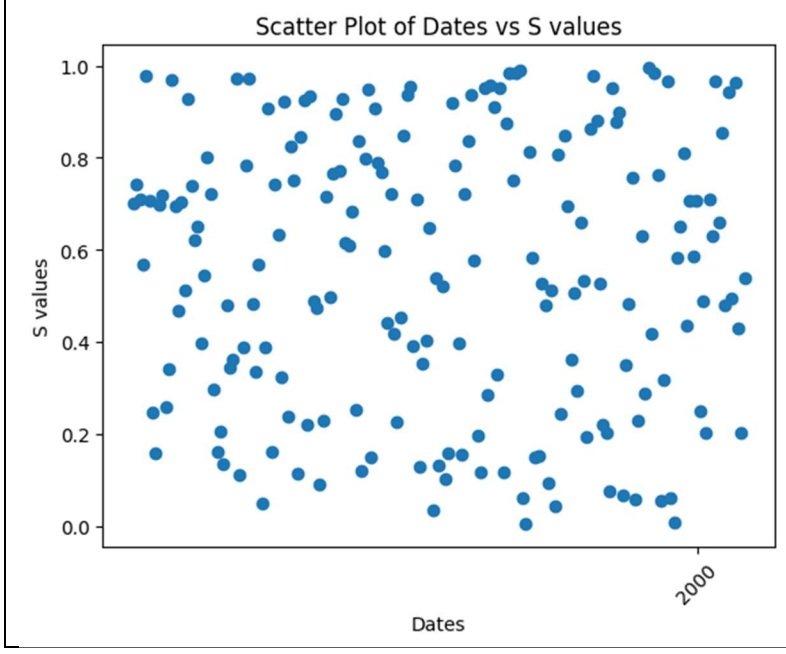


ID = 154
Score vis = 80
Score task = 70
Score human = 80

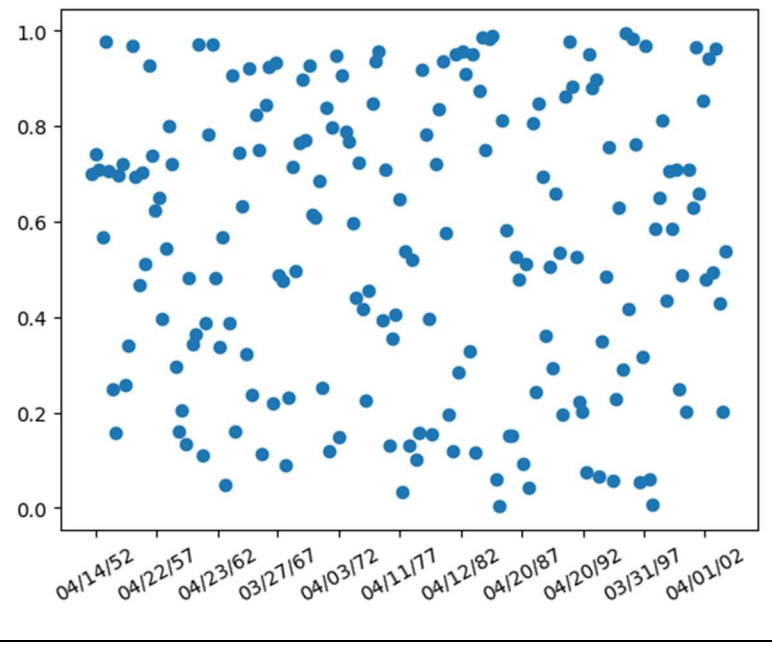
Task: Create a scatter plot showing the relationship or trend between the 'dates' and the 's' values. The dates should be formatted and displayed along the x-axis using a specific rule that highlights particular dates with a custom interval and format. Adjust the x-axis tick labels for better readability by rotating them and modifying their size.

Style: Customize the axis formatting by setting a major locator to control how dates are displayed based on a specified rule. Use a date formatter to change the date display format. Style the tick labels by setting their rotation and size to enhance plot clarity and aesthetics. The final plot should effectively communicate the trends or patterns in the data across time.

Generated



Ground truth



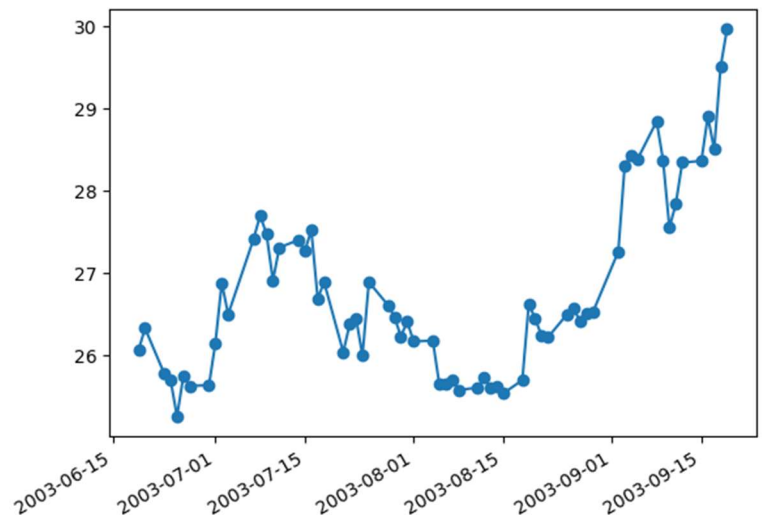
ID = 155
Score vis = 95
Score task = 100
Score human = 100

Task: Create a plot to visualize how the 'Close' price varies over time. Convert the 'Date' column from string to a datetime format before plotting. Use a scatter plot connected by lines to display this data, ensuring that each data point on the x-axis (dates) is labeled properly.

Style: Customize the x-axis to format the date labels using a user-defined formatting class. All dates should be displayed in a human-readable format (e.g., YYYY-MM-DD) and should auto-adjust to prevent overlap. Use blue color for the plot elements (points and lines) to enhance visual clarity.

Generated

Ground truth

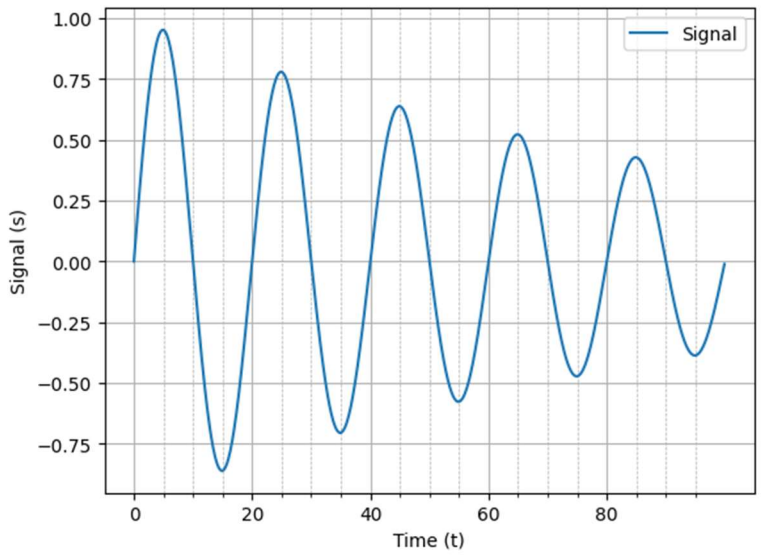


ID = 156
Score vis = 90
Score task = 95
Score human = 100

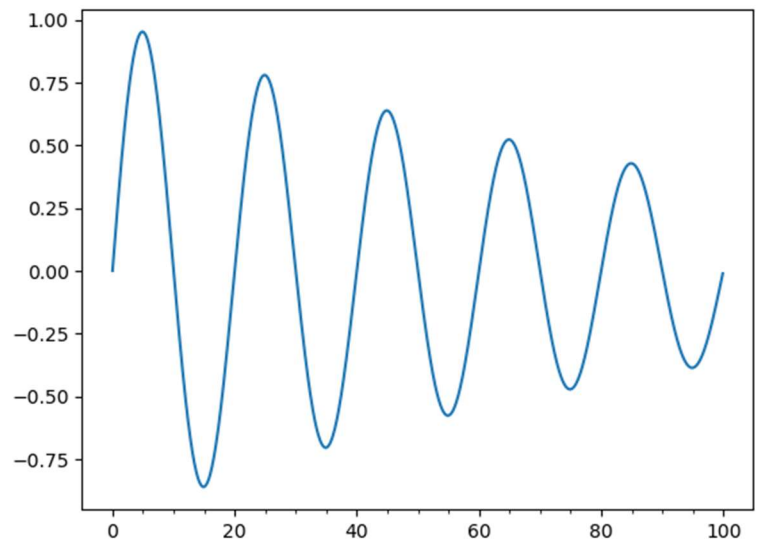
Task: Construct a line plot where the x-axis represents the sequence or time 't' and the y-axis represents the corresponding signal or measurement values 's'. The plot should display the data clearly allowing for easy observation of trends or patterns in 's' as 't' increases.

Style: Configure major and minor ticks on the x-axis to enhance readability. Major ticks should occur every 20 units, and minor ticks every 5 units within these intervals. The labels on the major ticks should display integer values, enhancing the distinction between consecutive ticks on the x-axis. The visual styling should be clear, with a straightforward line plot being sufficient for observing the data's behavior.

Generated



Ground truth

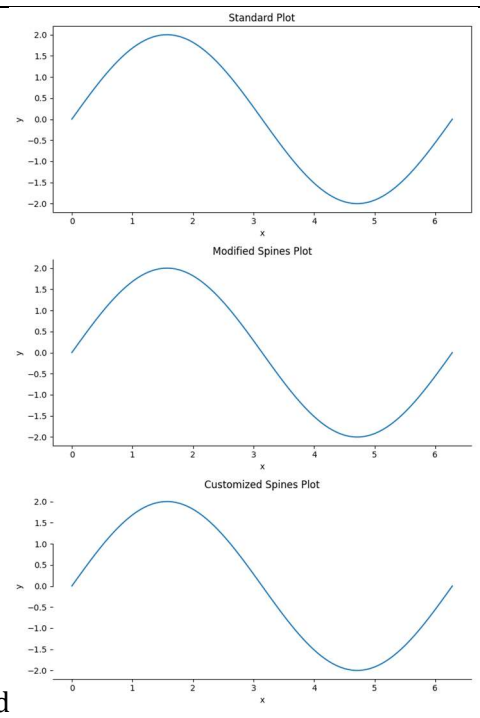


ID = 157
Score vis = 90
Score task = 100
Score human = 100

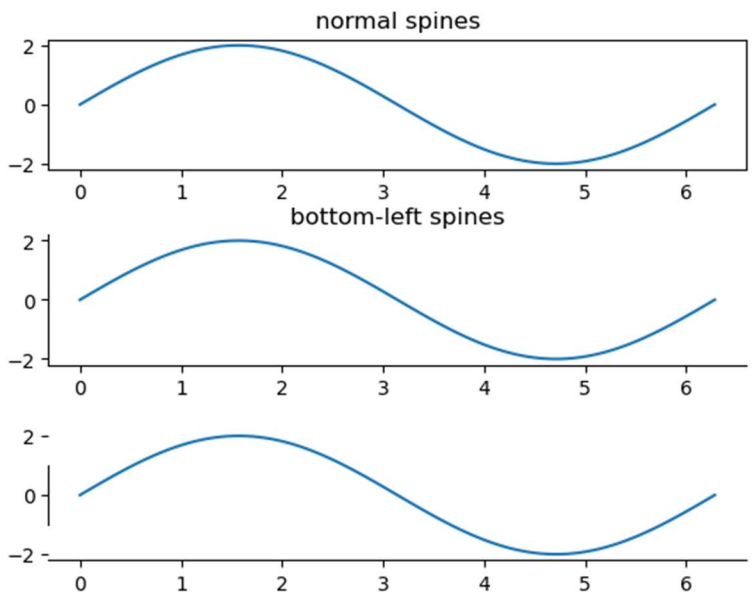
Task: Create three separate line plots stacked vertically. Each plot features the same set of data points, representing 'x' versus 'y'. The first plot will have a standard appearance. The second plot will have visible only the bottom and left spines, with specified modifications to the visibility of tick marks. The third plot will undergo further customization such that the left spine is only bound between specific y-values, asserting more control over the appearance of the plot.

Style: Employ distinct styling options for each subplot. The first maintains all four spines with all ticks displayed typically. On the second, hide the top and right spines and modify tick positions to be displayed only on the visible spines (bottom and left). In the third subplot, further manipulate the bounds of the visible left spine and hide the top and right spines, while maintaining tick positions like the second plot. Adjust the spacing between the subplots to ensure labels and titles do not overlap, maintaining clear visibility for plot interpretation.

Generated



Ground truth

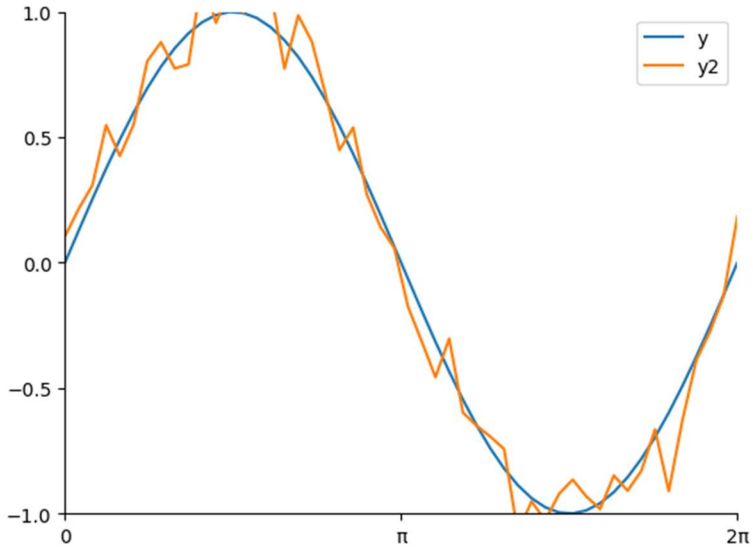


ID = 158
Score vis = 95
Score task = 95
Score human = 95

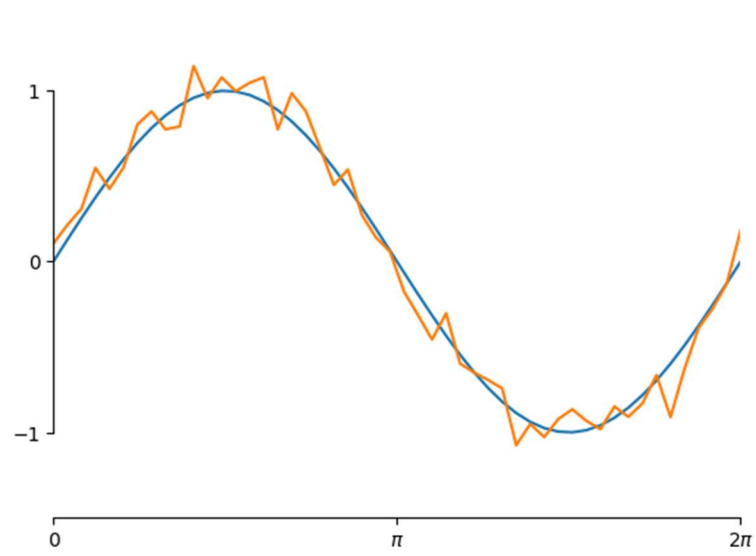
Task: Create a line plot using the values from the dataframe. The plot should include two lines: one representing the 'y' values against 'x' and another representing 'y2' values against 'x'. The x-axis should range from 0 to twice the value of pi, and the y-axis should display a range that accommodates the oscillations in 'y' and 'y2'.

Style: Adjust the plot to have a clear and simple presentation. Set specific ticks on the x-axis to represent 0, pi, and 2*pi, with appropriate labels for each. Configure the y-axis with ticks at meaningful intervals and set limits to focus on the primary range of data. Modify the plot spines for a clean look by hiding the top and right spines and setting bounds on the left spine. Ticks should appear only on the lower and left sides of the plot to maintain a neat appearance.

Generated



Ground truth

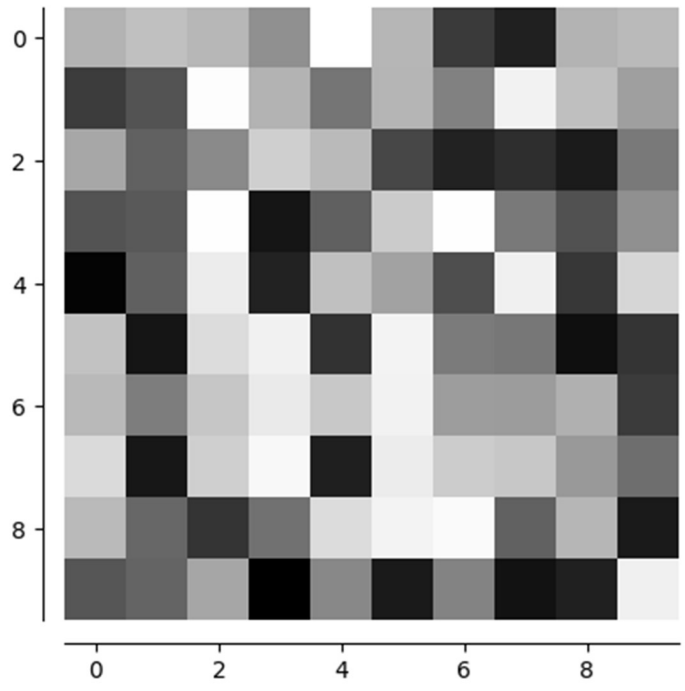


ID = 159
Score vis = 95
Score task = 95
Score human = 100

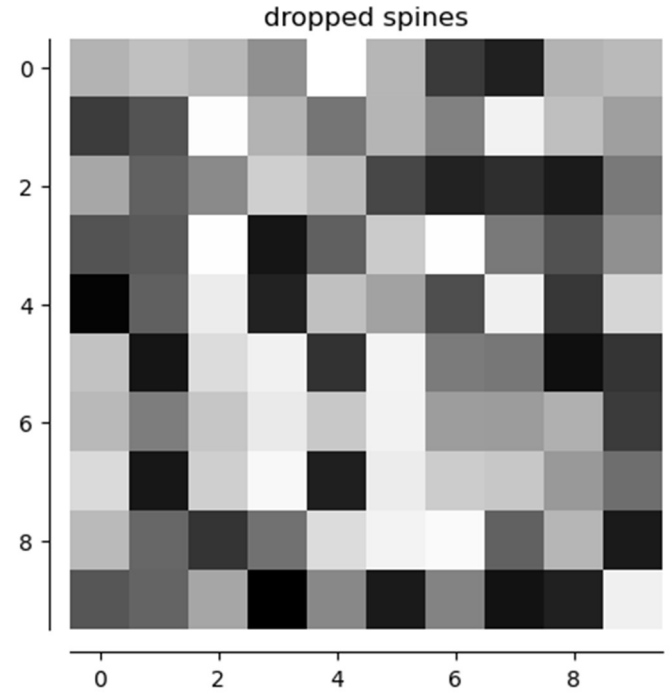
Task: Create a visual representation of the numerical data in the DataFrame by converting it into a grayscale image where each cell in the DataFrame corresponds to a pixel in the image. Each pixel color should vary based on the respective value it represents, with a spectrum ranging from black to white, indicating low to high values, respectively. Adjust the visual clarity of the plot by modifying the spine placements and tick positions.

Style: Adjust the plot's aesthetics by dropping the right and top spines making them invisible and moving the left and bottom spines outward. This gives the plot a clean and focused appearance on the data. Set ticks to only appear on the visible spines (left and bottom), enhancing the simplicity and readability. Use a grayscale colormap to map the DataFrame values to corresponding shades of gray.

Generated



Ground truth

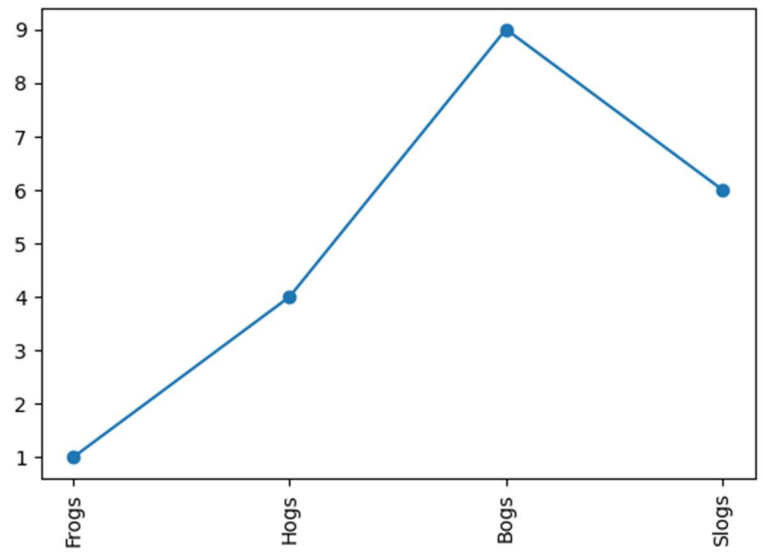


ID = 160
Score vis = 95
Score task = 90
Score human = 100

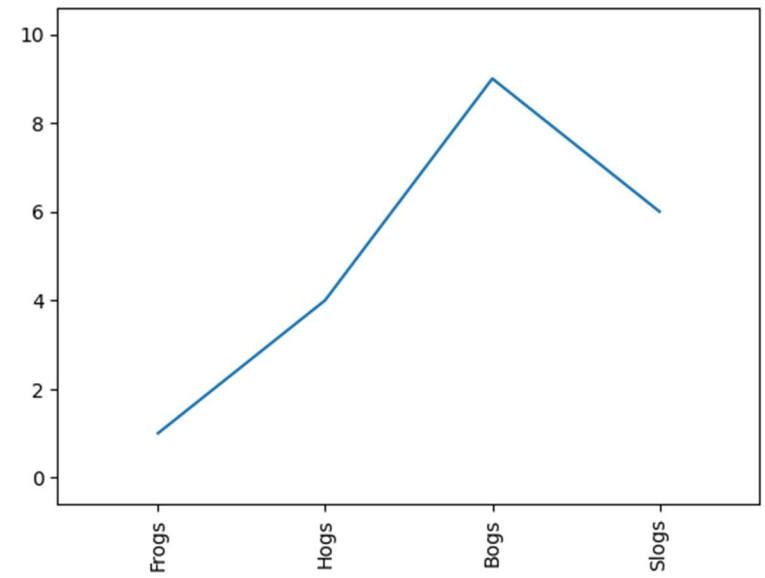
Task: description: Create a line plot with the "x" values plotted against the "y" values. Customize the x-axis tick marks to display the corresponding labels from the "labels" column and rotate these labels vertically for better readability.

Style: description: Adjust the margins of the plot to ensure all labels fit without overlapping. Ensure that the plot provides a clear view, including all necessary labels and avoiding any clipping of content.

Generated



Ground truth

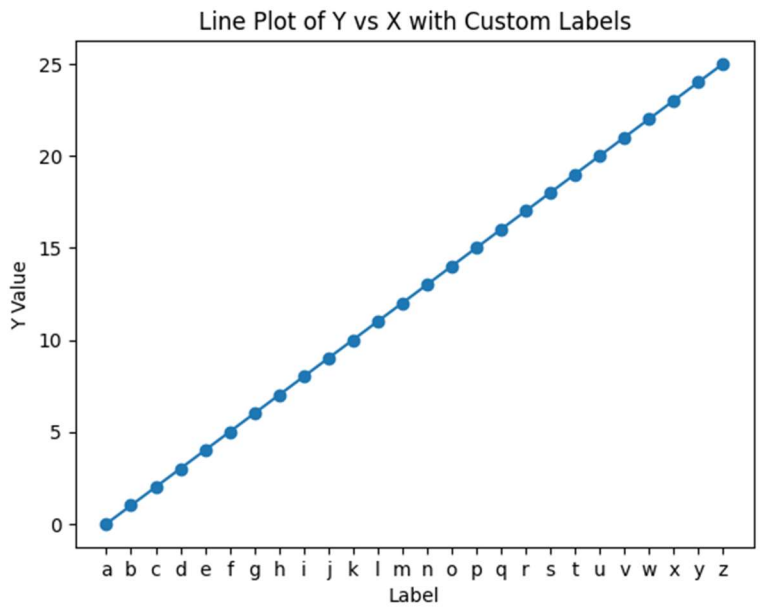


ID = 161
Score vis = 60
Score task = 100
Score human = 95

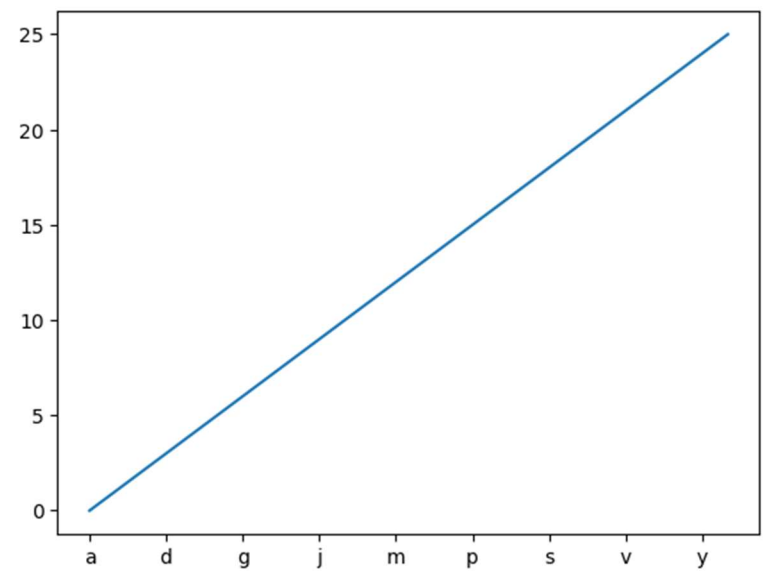
Task: Generate a line plot using the 'x' values for the horizontal axis and 'y' values for the vertical axis. Customize the x-axis to display string labels corresponding to each x-value from the 'label' column in the DataFrame.

Style: Customize the axes, ensuring that the x-axis labels are formatted to display strings from the 'label' column and ticks are placed at every integer x-value. Style the plot to provide clear visualization and differentiation of data points and labels.

Generated



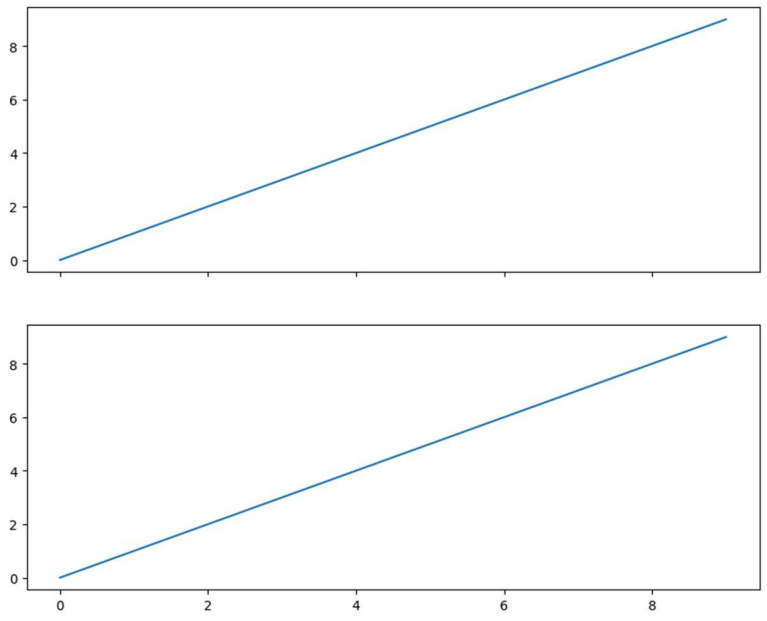
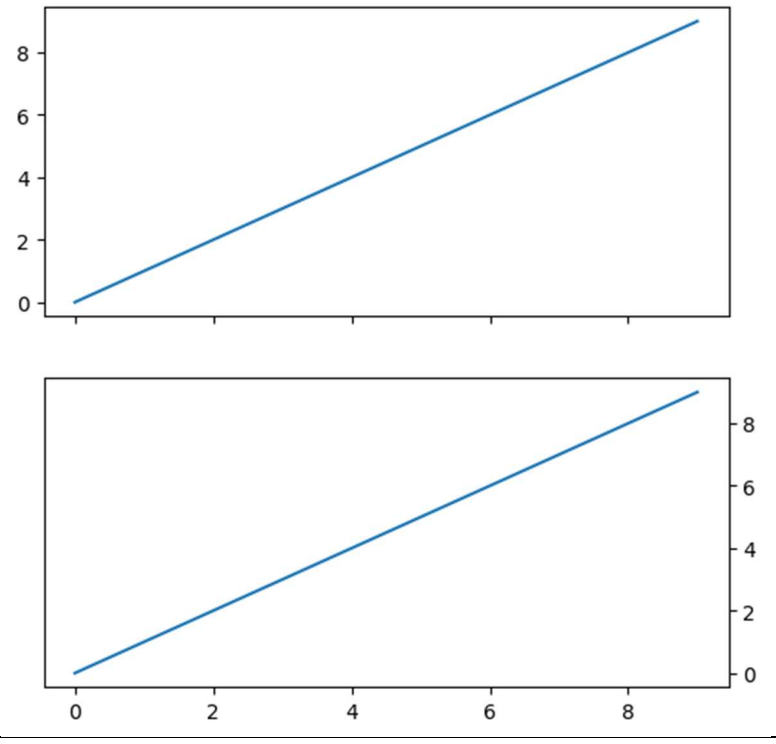
Ground truth



ID = 162
Score vis = 100
Score task = 30
Score human = 95

Task: Generate two subplots arranged vertically, sharing the same 'x' axis. The first subplot displays a line plot of 'y' values against 'x' values with customized y-axis tick marks appearing on the left side. The second subplot also displays a line plot of the same 'y' values against 'x' values, using standard axis settings.

Style: Adjust tick settings globally, specifying that y-axis ticks and labels should appear on the right side and not on the left. The figure size should be uniformly defined and suitable for viewing both subplots clearly. Ensure the x-axis is shared among the subplots to maintain visual coherence and axis alignment.

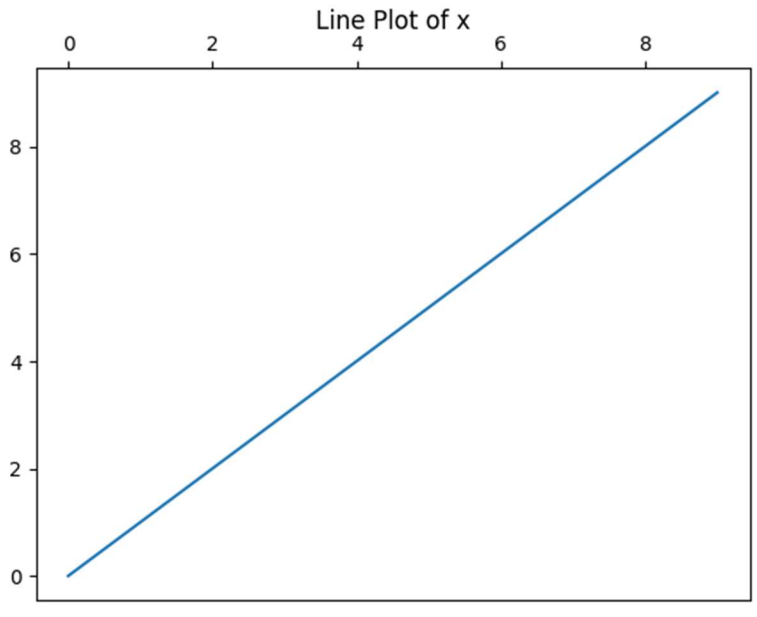
Generated	Ground truth
	

ID = 163
Score vis = 90
Score task = 90
Score human = 100

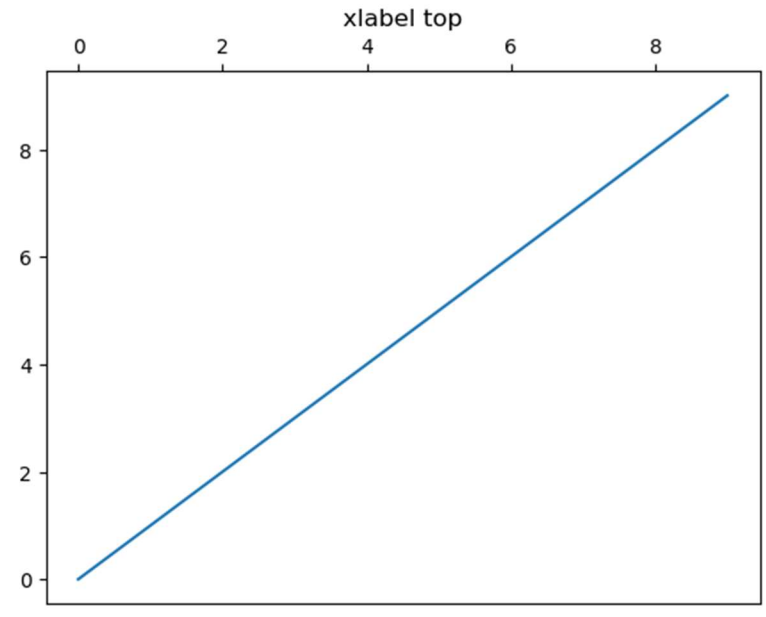
Task: Generate a line plot using the numerical data from the dataframe's 'x' column. Configure the plot to display the x-axis labels on top rather than the bottom. Include a title above the plot which should adjust its position relative to the top x-axis labels.

Style: Adjust the plot's style to modify the placement of the x-axis ticks and labels so that they appear on the top of the plot. Ensure that the graph maintains clarity and that all elements (including tick labels and the plot title) are unobstructed and easy to read.

Generated



Ground truth

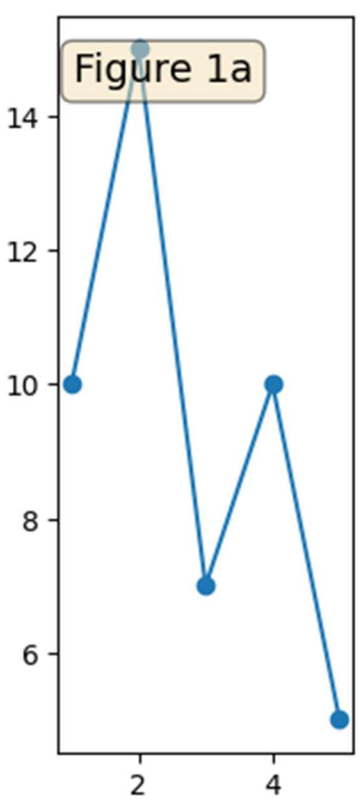


ID = 164
Score vis = 10
Score task = 90
Score human = 100

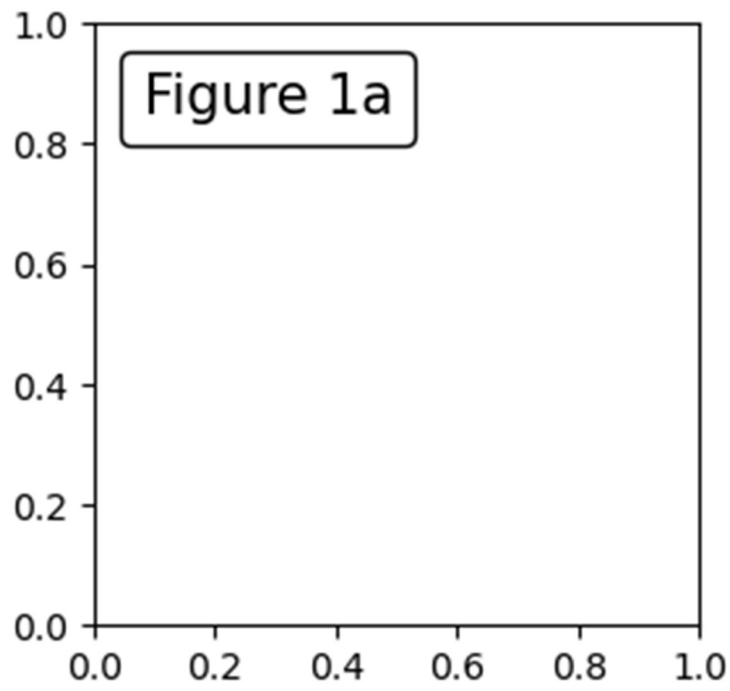
Task: The required plot is a simple figure with an embedded text box. The plot should be square in shape, and display a textual annotation labeled "Figure 1a" located at the upper left of the plot.

Style: The text within the plot should be distinctly styled: it should have a specific font size, be enclosed within a rounded frame, and have adjusted padding around the text. The numerical axes should be scaled proportionately and be clear and simple, focusing viewer attention on the textual annotation rather than on detailed data visualization.

Generated



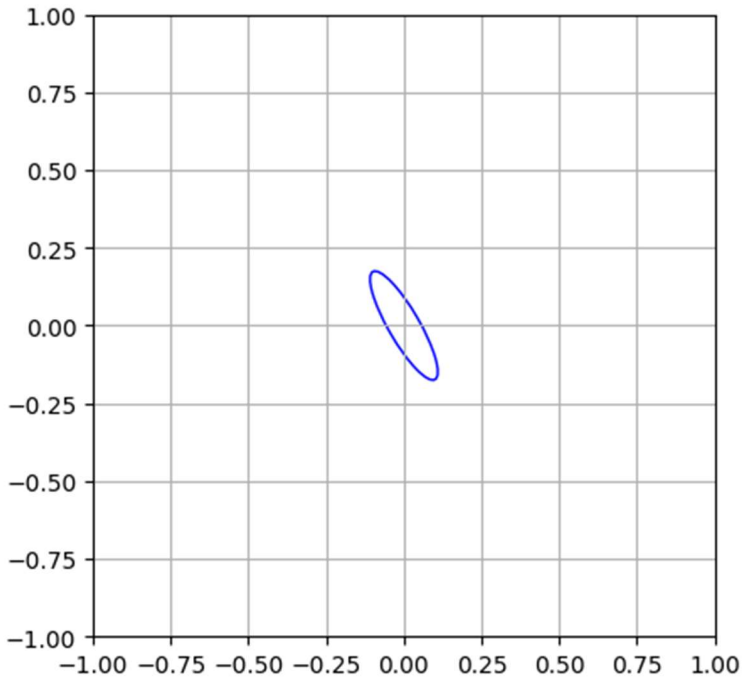
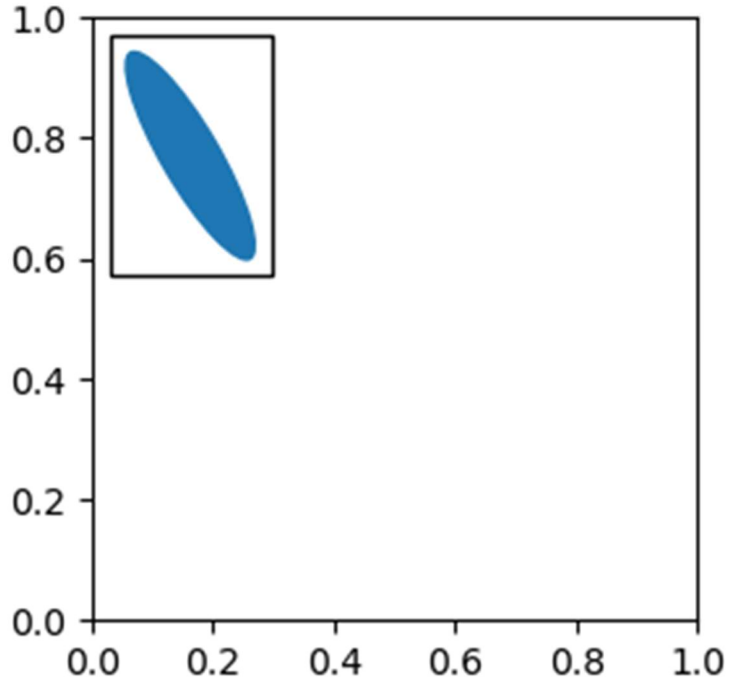
Ground truth



ID = 165
Score vis = 20
Score task = 90
Score human = 100

Task: description: Create a plot that involves iterating over rows in the dataframe to read the ellipse parameters of position, size, and orientation. Construct an ellipse accordingly, and place each ellipse within a bounding box located in a predefined part of the plot canvas (upper left in this case). Each ellipse should adhere to its specified attributes gathered from the dataframe.

Style: description: The plot should feature a clear and uncluttered design with a single bounding box containing the ellipse. This should be visually represented against a simple coordinate grid with both the x and y axes scaled appropriately to the range of the ellipse's parameters. The aesthetic should focus on clarity to emphasize the ellipse's orientation, size, and position as dictated by the dataframe.

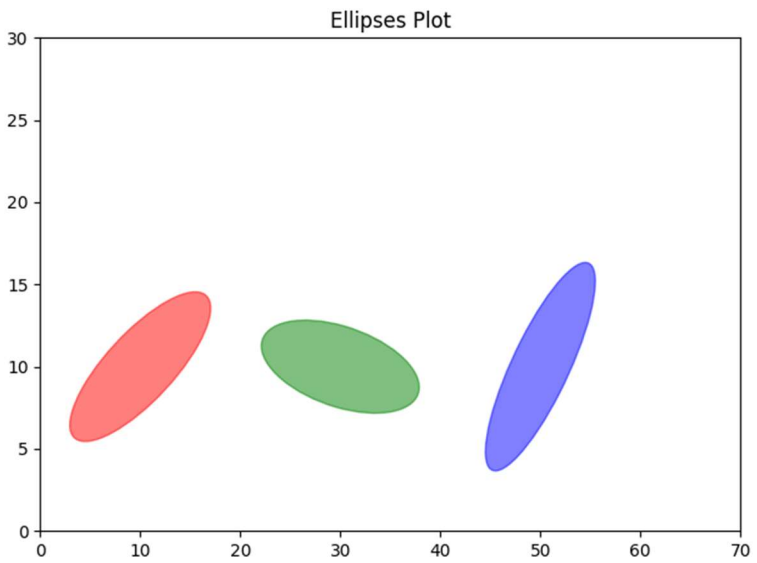
Generated	Ground truth
	

ID = 166
Score vis = 20
Score task = 90
Score human = 75

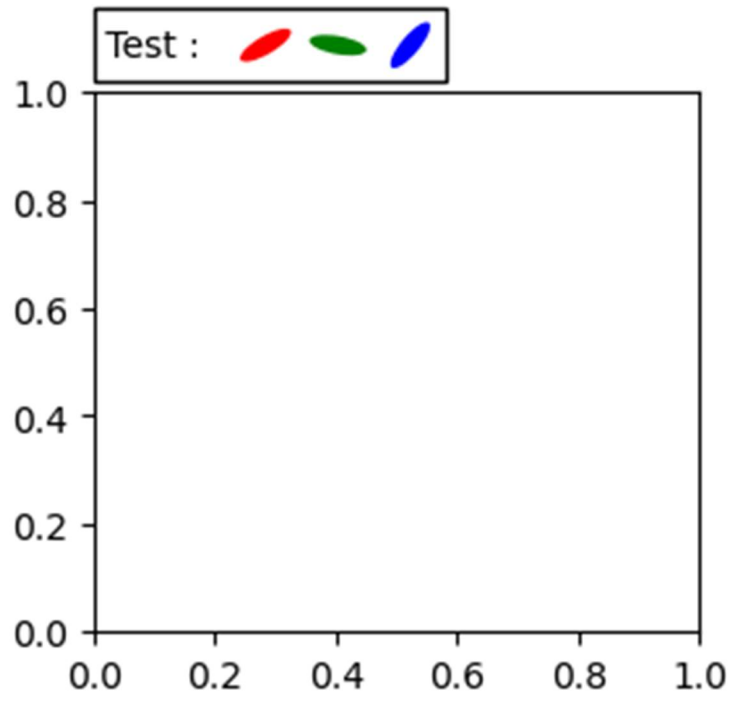
Task: Construct a plot that corresponds each row in the dataframe to an ellipse on a visual drawing space, using the specified attributes (center coordinates, width, height, angle) for each ellipse. First, create a plot area and integrate a textual area and a drawing area. Embed ellipses onto the drawing area. Group these into a box and anchor it to the plot to maintain relational positioning.

Style: Configure the layout and aesthetics of the plot to ensure proper visibility and styling. This includes adjustments to the plot dimensions and margins. Use colors as specified in the dataframe for each ellipse and align text and shapes appropriately for clear visual presentation. Ensure the plot is appropriately titled and axis limits are set for optimal visualization.

Generated



Ground truth

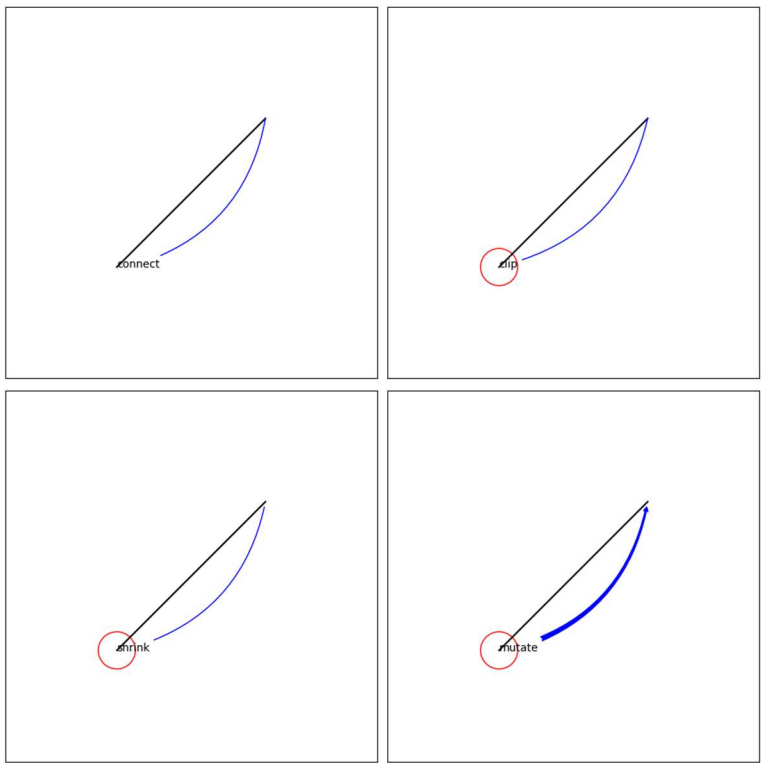


ID = 167
Score vis = 20
Score task = 100
Score human = 25

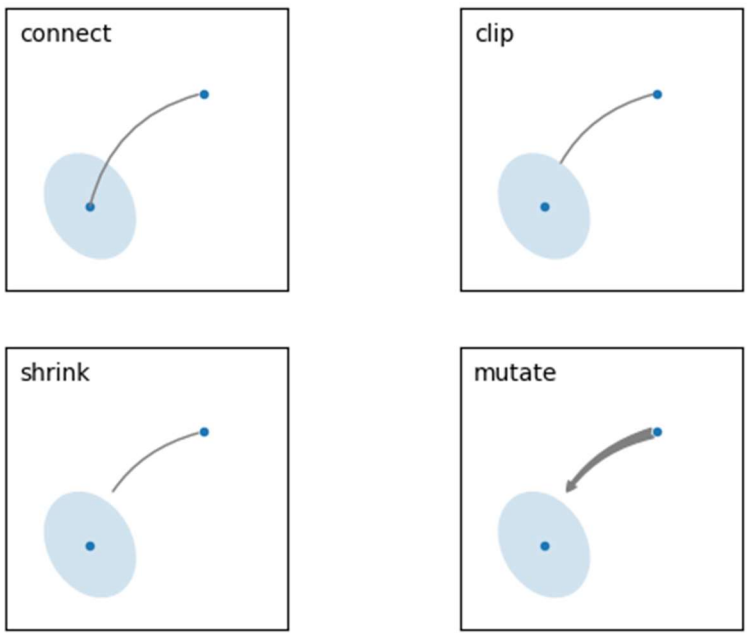
Task: Create a 2x2 subplot layout. For each subplot, a line will be drawn between two points defined by ('x1', 'y1') and ('x2', 'y2'). Around the starting point, an ellipse will be drawn. Add annotations between these points with directional arrows styled according to the dataframe's specifications. Each subplot also features custom text positioned at the top-left.

Style: Customize each subplot to limit its x and y ranges between 0 and 1 with no axis ticks displayed. Aspect ratio should be maintained at 1. Annotations involve adjustable attributes like arrow style, color, connection style, and possible linking to an elliptical patch based on conditions specified in the dataframe. The opacity and orientation of the ellipse, as well as text properties, are detailed within the plot commands.

Generated



Ground truth



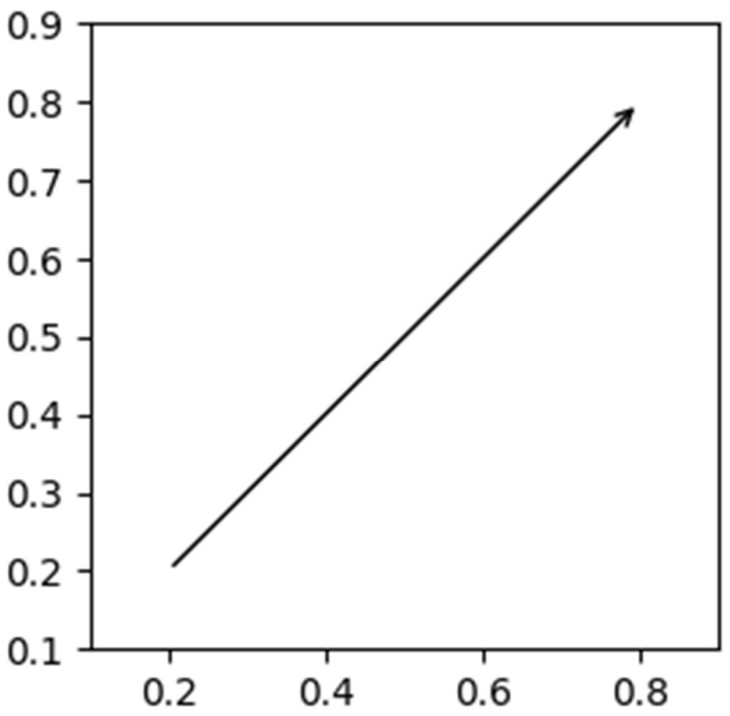
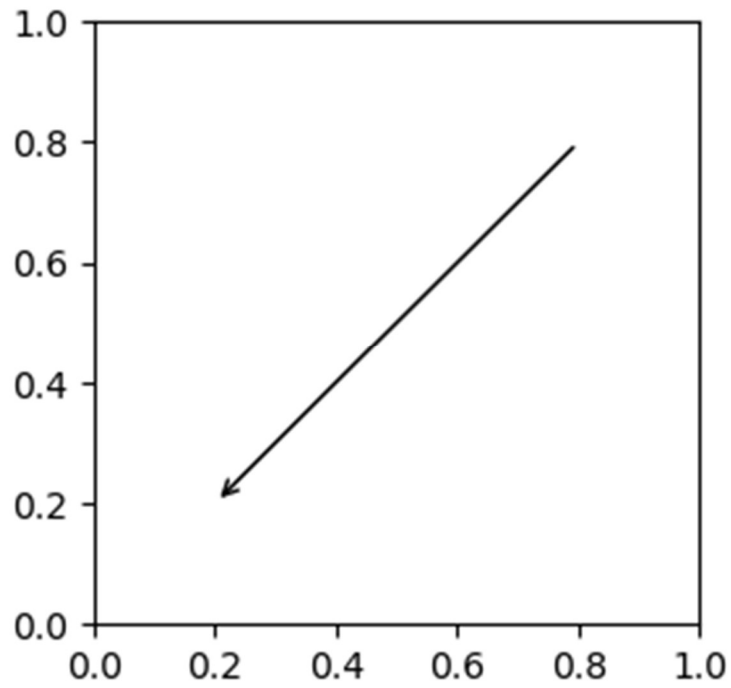
ID = 168
Score vis = 90
Score task = 100
Score human = 100

Task:

Create a plot illustrating a directed line segment (arrow) between two points specified in the DataFrame. The arrow originates at the first point and points towards the second point. The coordinates of the points are retrieved directly from the DataFrame.

Style:

The plot should be displayed in a square figure of size 3x3 inches. Utilize styling options for the arrow to show a simple arrowhead indicating direction, with a straight connection style between the points. The axes should be set to scale the plot appropriately based on the data values for clear visibility of the arrow within the plotting area.

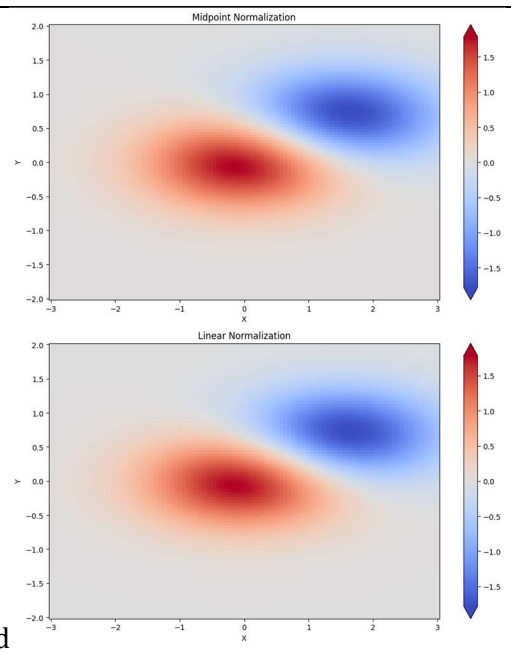
Generated	Ground truth
	

ID = 169
Score vis = 90
Score task = 95
Score human = 95

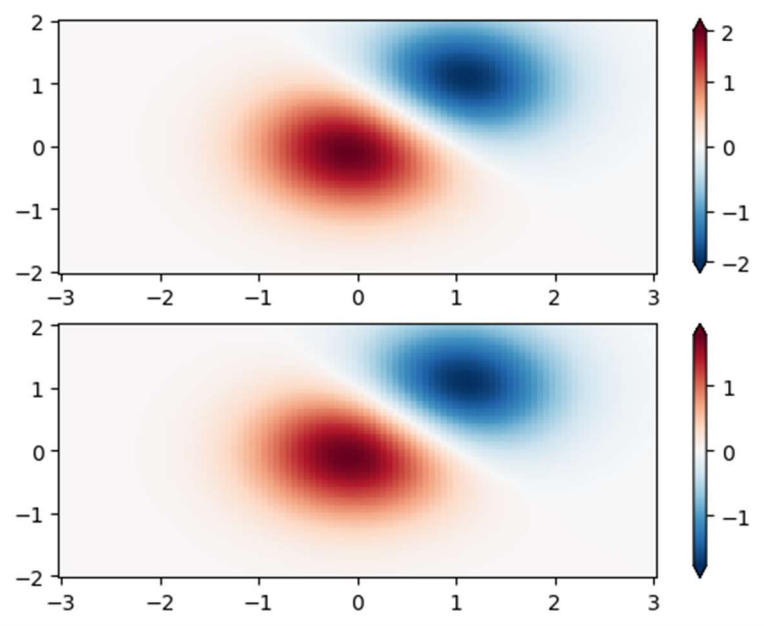
Task: description: The task will generate two subplots arranged vertically. Each subplot will display a color-coded mesh plot, where the mesh colors depict the 'Z' values over a grid defined by 'X' and 'Y' values. The first subplot will enhance visualization around a central 'Z' value (midpoint), normalized such that this midpoint appears neutral in color. The second subplot uses a color normalization based solely on the extremities of the 'Z' data range. Include a color bar beside each subplot to illustrate the range and correlation of the colors to the 'Z' values.

Style: description: Utilize a diverging color map (e.g., ranging from blue to red) to reflect negative through positive values effectively. Adjust the normalization of color maps in each plot to highlight different characteristics of the data: linear interpolation in one based on the maximum absolute 'Z' value and midpoint normalization in the other to demonstrate variations around zero. Each subplot should extend the color bar to denote values exceeding plotted ranges. Also, ensure the plots are clear and precisely annotate axes to correspond to the actual data values.

Generated



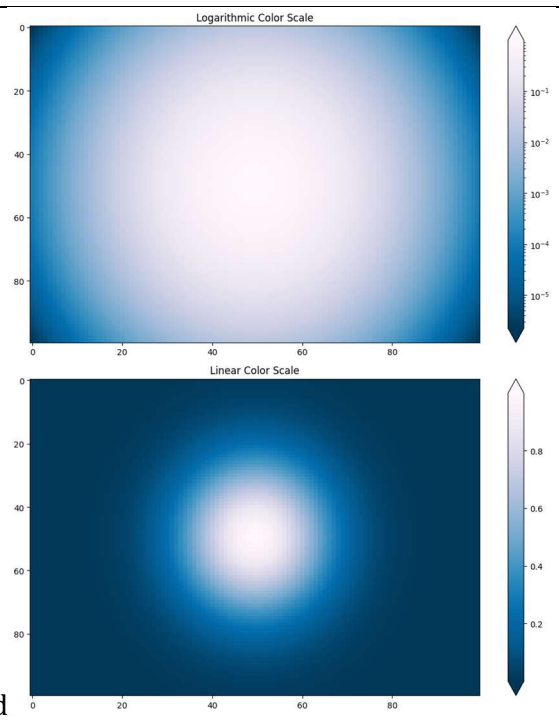
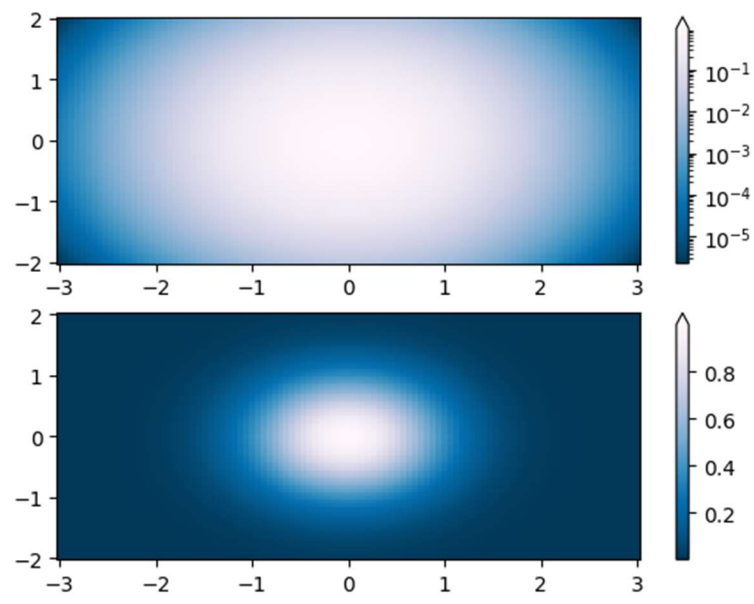
Ground truth



ID = 170
Score vis = 90
Score task = 100
Score human = 100

Task: The task requires generating two subplots in a single column layout. Each subplot should visualize data from the DataFrame using a grid representation with the 'X' and 'Y' values forming the grid coordinates, and the 'Z' values mapping to color intensities on this grid. The top subplot utilizes a logarithmic color scale to enhance the contrast in data visualization, particularly useful for highlighting a range of data with exponential differences. Both subplots include color bars to indicate the scale of values, with extensions to show the range beyond the color map limits.

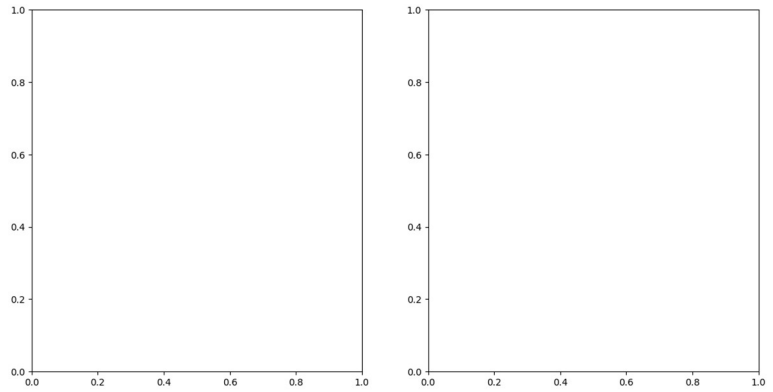
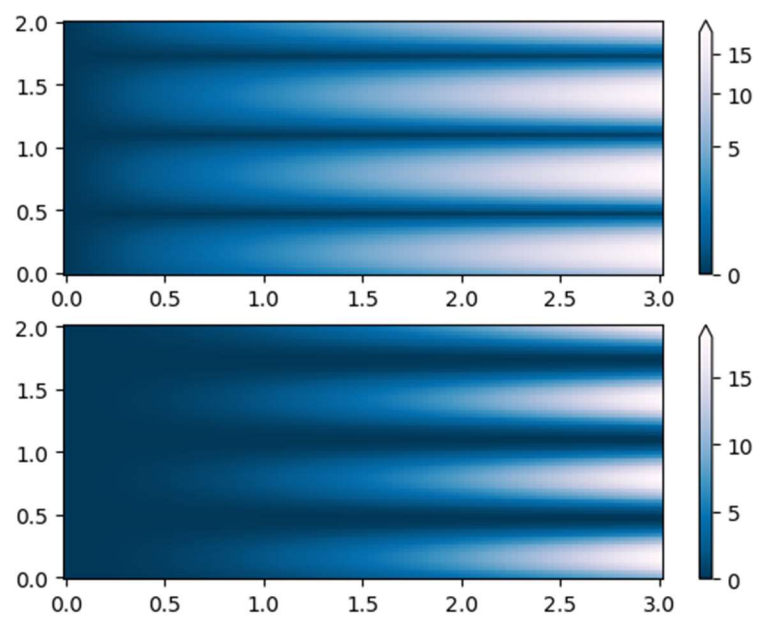
Style: Both plots should use the same color map, 'PuBu_r', providing a sequential palette that scales from light to dark, suitable for showing varying intensities. The top plot needs to have a logarithmic normalization to address wide-ranging data values, whereas the bottom plot uses a regular, linear scale. Both plots should include color bars at their right side, equipped to display their respective value ranges effectively, and clearly annotate their scales for better readability and data interpretation.

<table>
<tr>
<td>Generated
 
</td>
<td>Ground truth
  </td>
</tr>
</table>

ID = 171
Score vis = 0
Score task = 0
Score human = 5

Task: The code is designed to generate two subplot visualizations using data from the DataFrame. Each subplot represents a form of intensity mapping or heatmapping where the 'X' and 'Y' columns determine position and the 'Z1' column determines intensity or value at each point. Each plot uses a grid interpolation from scattered data points, creating a smooth, continuous representation of the variable 'Z1' across a defined spatial area. Both subplots use a similar type of visualization but differ slightly in their color normalization methods to represent data intensities.

Style: Both plots utilize a color gradient to visually represent data values, with a specific color map chosen for its visual presentation of numerical gradients. The first plot employs a color normalization technique designed to enhance the visual perception of differences in lower values, altered using a gamma adjustment. The second plot presents a straightforward approach to color gradient mapping. Each plot includes a color bar indicating the range of values in the plot, which provides a clear indicator of what the colors represent in terms of 'Z1' data values.

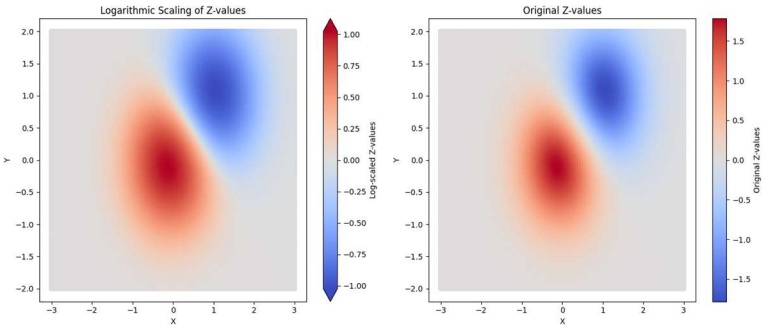
Generated	Ground truth
	

ID = 172
Score vis = 90
Score task = 100
Score human = 100

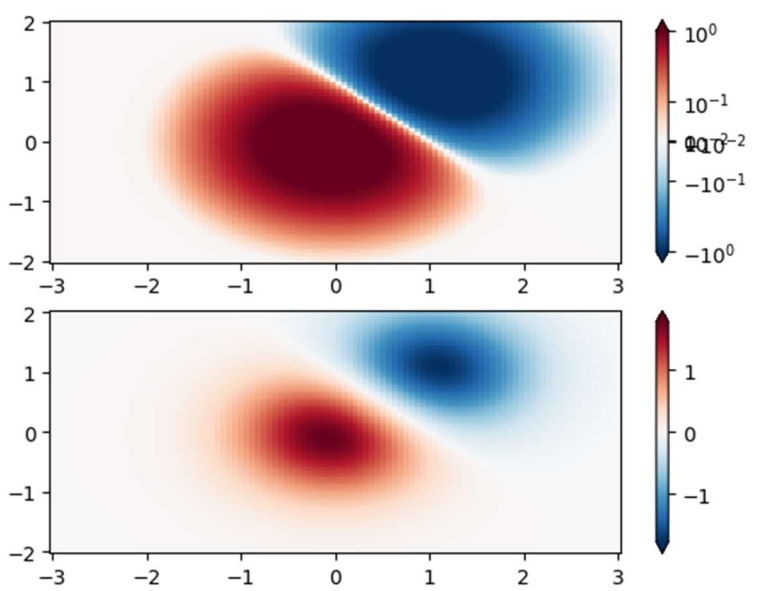
Task: Create two subplot visualizations of the Z-values over the X and Y grid. The first plot must transform the data display using a logarithmic scaling for Z-values around zero to enhance visual contrast in regions of low absolute value. The second plot will show the data without this transformation to provide a direct view of the data distribution.

Style: Both plots should use a diverging color map to represent the range and sign of Z-values effectively. The inclusion of a color bar for each plot will guide interpretation of the values. To accommodate extremes and enhance readability, the color bar in the first plot should adapt to include extensions at both ends.

Generated



Ground truth

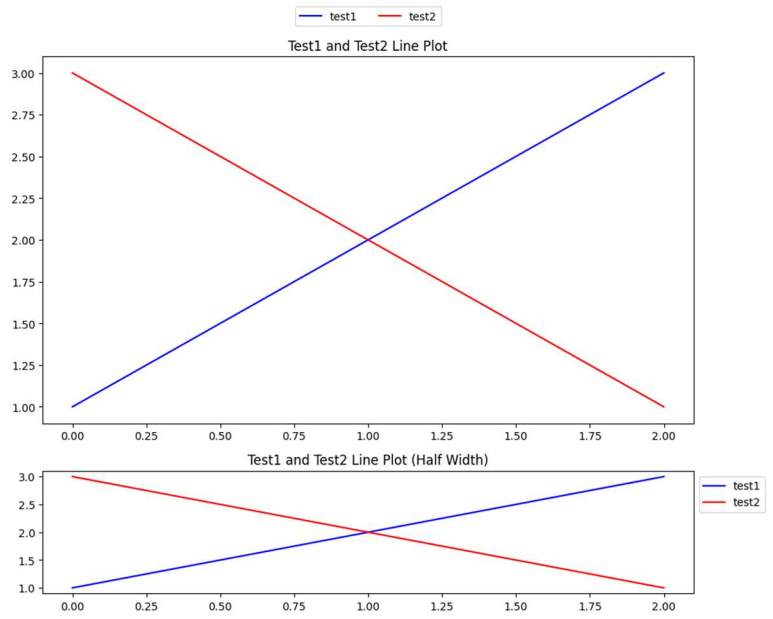


ID = 173
Score vis = 80
Score task = 100
Score human = 70

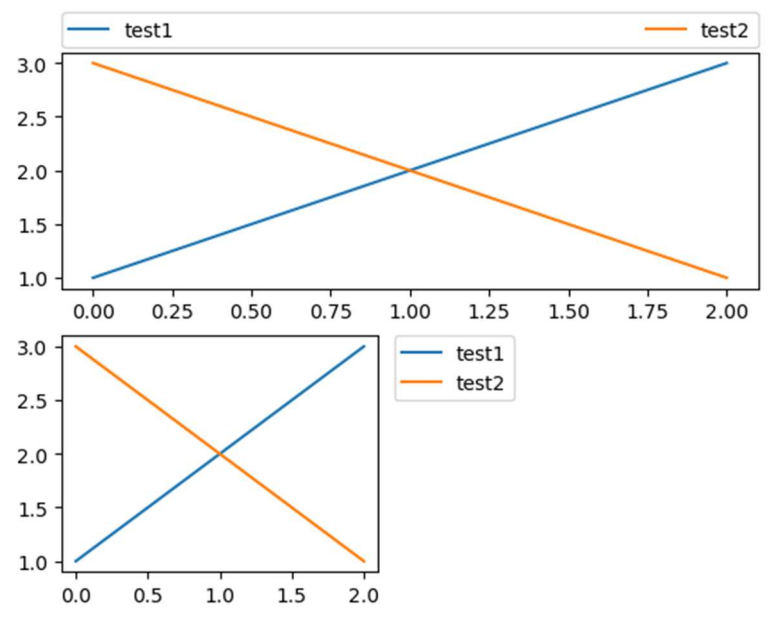
Task: Create a figure consisting of two subplots. The first subplot should display both 'test1' and 'test2' as line plots on the top section of the figure, spanning the full width. The second subplot, located below the first on the left side of the figure and covering half the width, should again plot the same two lines. Both plots should include legends displaying the labels of the respective lines.

Style: In the first subplot, place the legend above the plot spanning across the plot width with a particular alignment at the bottom center. For the second subplot, position the legend outside the plot on the upper left side. Use distinctive colors for each dataset to clearly differentiate between 'test1' and 'test2' in the visual representation.

Generated



Ground truth

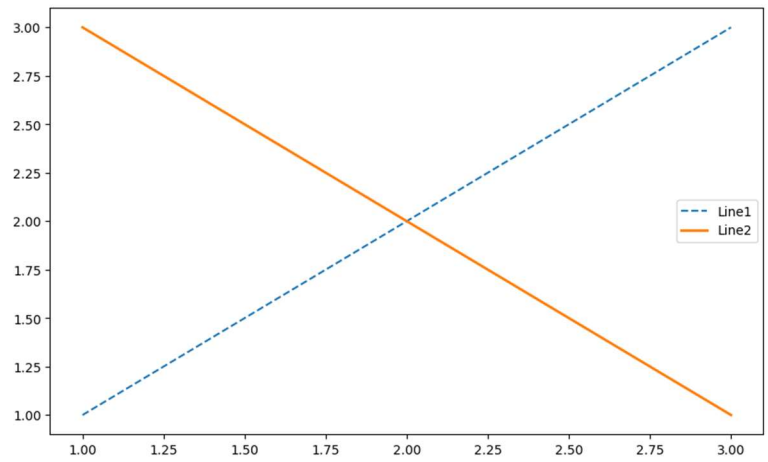


ID = 174
Score vis = 95
Score task = 90
Score human = 100

Task: Create a plot with two lines representing the two data series from the DataFrame. Each line should use the x-axis data for plotting. Apply a distinct line style to each series to differentiate between them. Incorporate legends for each data series, positioned thoughtfully on the plot to avoid overlapping and ensure clarity.

Style: he plot by customizing line styles and thickness. For the first data series, use a dashed line style. For the second series, use a solid line with increased thickness. Both lines should be clearly distinguishable from each other, not only by their positioning and intersection but also by their styling. Include legends for each series with appropriate placement to ensure both guides are visible and do not overlap.

Generated



Ground truth