Scatterplot Matrix in Python

How to make scatterplot matrices or sploms natively in Python with Plotly.

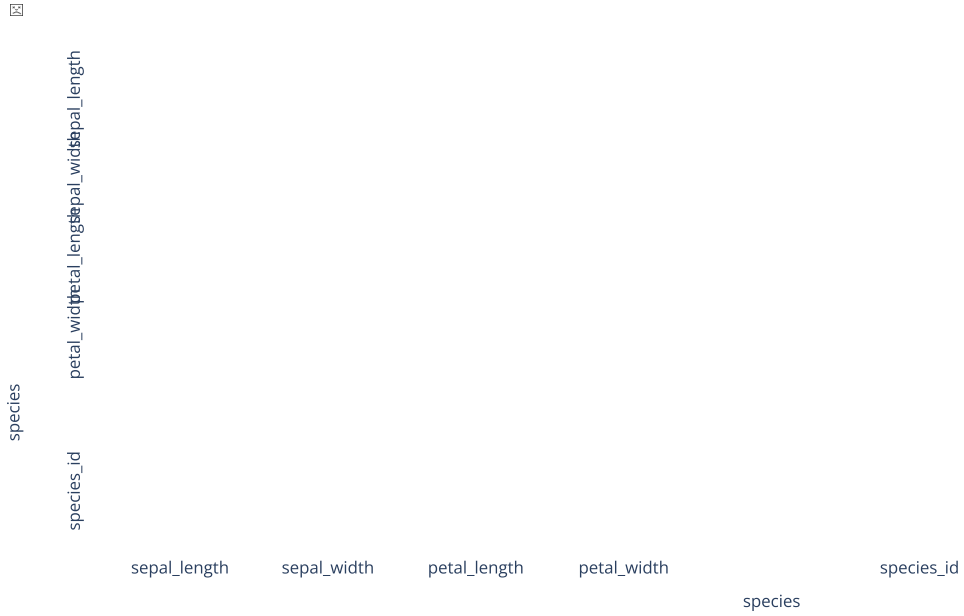
Scatter matrix with Plotly Express

A scatterplot matrix is a matrix associated to n numerical arrays (data variables), $X_1,X_2,...,X_n$, of the same length. The cell (i,j) of such a matrix displays the scatter plot of the variable Xi versus Xj.

Here we show the Plotly Express function `px.scatter_matrix` to plot the scatter matrix for the columns of the dataframe. By default, all columns are considered.

Plotly Express (/python/plotly-express/) is the easy-to-use, high-level interface to Plotly, which operates on a variety of types of data (/python/px-arguments/) and produces easy-to-style figures (/python/styling-plotly-express/).

```
import plotly.express as px
df = px.data.iris()
fig = px.scatter_matrix(df)
fig.show()
```



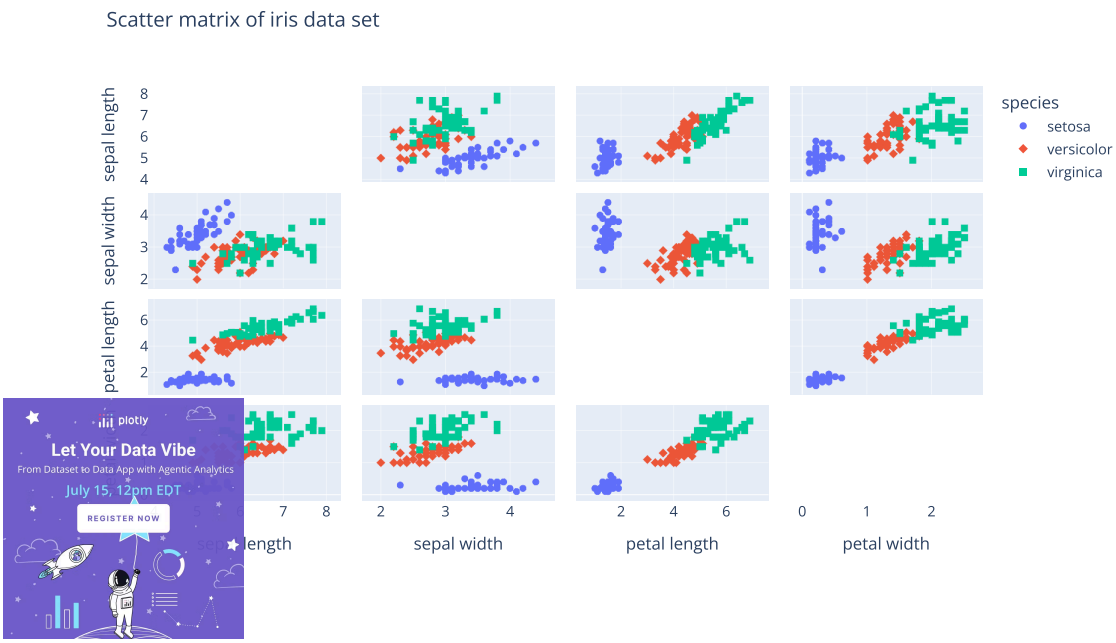
Specify the columns to be represented with the `dimensions` argument, and set colors using a column of the dataframe:

```
import plotly.express as px
df = px.data.iris()
fig = px.scatter_matrix(df,
    dimensions=["sepal_length", "sepal_width", "petal_length", "petal_width"],
    color="species")
fig.show()
```

Styled Scatter Matrix with Plotly Express

The scatter matrix plot can be configured thanks to the parameters of px.scatter_matrix, but also thanks to fig.update_traces for fine tuning (see the next section to learn more about the options).

```
import plotly.express as px
df = px.data.iris()
fig = px.scatter_matrix(df,
    dimensions=["sepal_length", "sepal_width", "petal_length", "petal_width"],
    color="species", symbol="species",
    title="Scatter matrix of iris data set",
    labels={col:col.replace('_', ' ') for col in df.columns}) # remove underscore
fig.update_traces(diagonal_visible=False)
fig.show()
```



Scatter matrix (splom) with go.Splom

If Plotly Express does not provide a good starting point, it is possible to use the more generic go.Splom class from plotly.graph_objects (/python/graph-objects/). All its parameters are documented in the reference page https://plotly.com/python/reference/splom/ (https://plotly.com/python/reference/splom/).

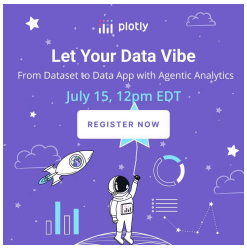
The Plotly splom trace implementation for the scatterplot matrix does not require to set $x=Xi$, and $y=Xj$, for each scatter plot. All arrays, $X_1,X_2,...,X_n$, are passed once, through a list of dicts called dimensions, i.e. each array/variable represents a dimension.

A trace of type splom is defined as follows:

```
trace=go.Splom(dimensions=[dict(label='string-1',
                                values=X1),
                           dict(label='string-2',
                                values=X2),
                           .
                           .
                           .
                           dict(label='string-n',
                                values=Xn)],
                           ....
               )
```

The label in each dimension is assigned to the axes titles of the corresponding matrix cell.

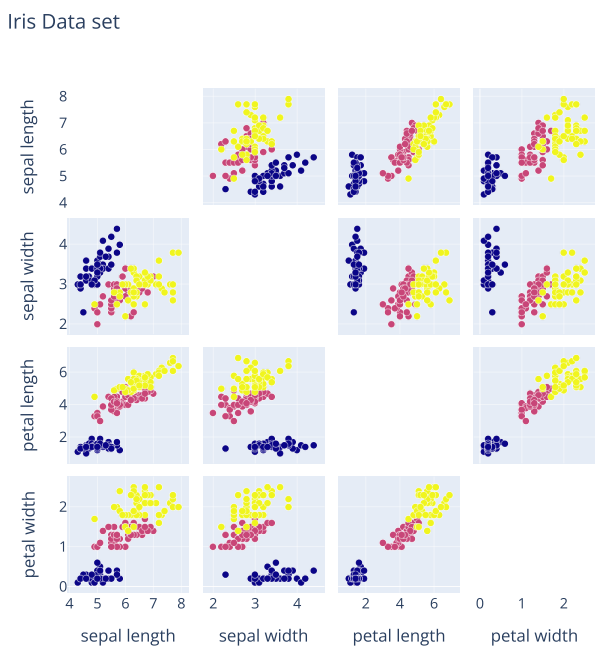
Splom of the Iris data set



```
import plotly.graph_objects as go
import pandas as pd

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/iris-data.csv')

# The Iris dataset contains four data variables, sepal length, sepal width, petal length,
# petal width, for 150 iris flowers. The flowers are labeled as `Iris-setosa`,
# `Iris-versicolor`, `Iris-virginica`.

# Define indices corresponding to flower categories, using pandas label encoding
index_vals = df['class'].astype('category').cat.codes

fig = go.Figure(data=go.Splom(
                dimensions=[dict(label='sepal length',
                                 values=df['sepal length']),
                            dict(label='sepal width',
                                 values=df['sepal width']),
                            dict(label='petal length',
                                 values=df['petal length']),
                            dict(label='petal width',
                                 values=df['petal width'])],
                text=df['class'],
                marker=dict(color=index_vals,
                            showscale=False, # colors encode categorical variables
                            line_color='white', line_width=0.5)
                ))


fig.update_layout(
    title=dict(text='Iris Data set'),
    dragmode='select',
    width=600,
    height=600,
    hovermode='closest',
)

fig.show()
```

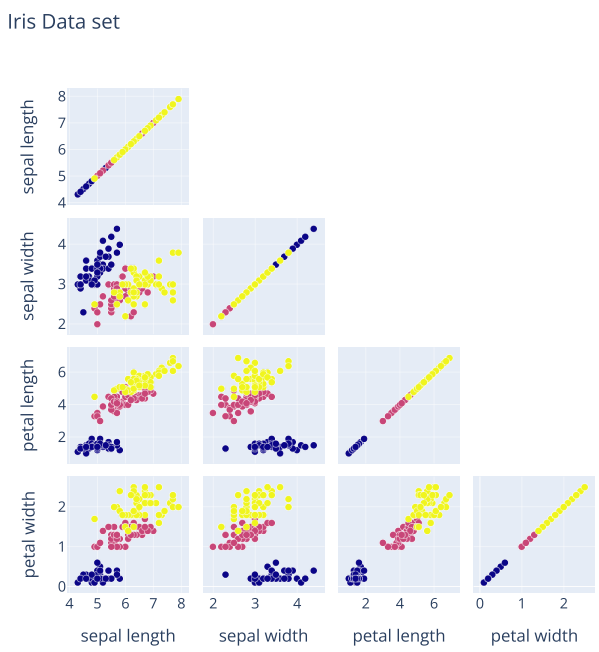
The scatter plots on the principal diagonal can be removed by setting diagonal_visible=False:

```
import plotly.graph_objects as go
import pandas as pd

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/iris-data.csv')
index_vals = df['class'].astype('category').cat.codes

fig = go.Figure(data=go.Splom(
                dimensions=[dict(label='sepal length',
                                 values=df['sepal length']),
                            dict(label='sepal width',
                                 values=df['sepal width']),
                            dict(label='petal length',
                                 values=df['petal length']),
                            dict(label='petal width',
                                 values=df['petal width'])],
                diagonal_visible=False, # remove plots on diagonal
                text=df['class'],
                marker=dict(color=index_vals,
                            showscale=False, # colors encode categorical variables
                            line_color='white', line_width=0.5)
                ))


fig.update_layout(
    title=dict(text='Iris Data set'),
    width=600,
    height=600,
)

fig.show()
```

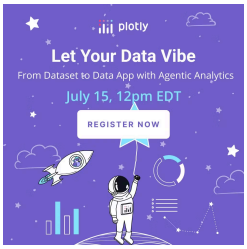


To plot only the lower/upper half of the splom we switch the default showlowerhalf=True/showupperhalf=True to False:

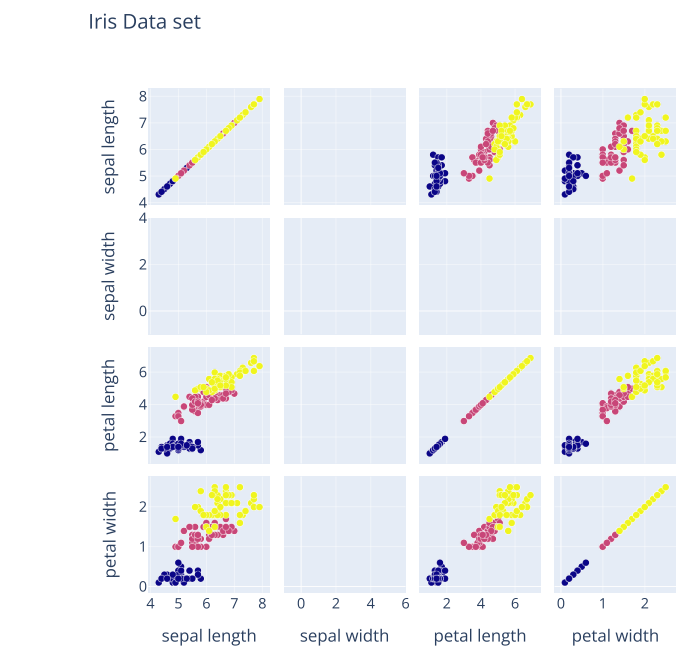
```
import plotly.graph_objects as go
import pandas as pd

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/iris-data.csv')
index_vals = df['class'].astype('category').cat.codes

fig = go.Figure(data=go.Splom(
                dimensions=[dict(label='sepal length',
                                 values=df['sepal length']),
                            dict(label='sepal width',
                                 values=df['sepal width']),
                            dict(label='petal length',
                                 values=df['petal length']),
                            dict(label='petal width',
                                 values=df['petal width'])],
                showupperhalf=False, # remove plots on diagonal
                text=df['class'],
                marker=dict(color=index_vals,
                            showscale=False, # colors encode categorical variables
                            line_color='white', line_width=0.5)
                ))


fig.update_layout(
    title=dict(text='Iris Data set'),
    width=600,
    height=600,
)

fig.show()
```



Each dict in the list dimensions has a key, visible, set by default on True. We can choose to remove a variable from splom, by setting visible=False in its corresponding dimension. In this case the default grid associated to the scatterplot matrix keeps its number of cells, but the cells in the row and column corresponding to the visible false dimension are empty:

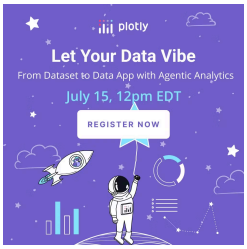


```
import plotly.graph_objects as go
import pandas as pd

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/iris-data.csv')
index_vals = df['class'].astype('category').cat.codes

fig = go.Figure(data=go.Splom(
                dimensions=[dict(label='sepal length',
                                 values=df['sepal length']),
                            dict(label='sepal width',
                                 values=df['sepal width'],
                                 visible=False),
                            dict(label='petal length',
                                 values=df['petal length']),
                            dict(label='petal width',
                                 values=df['petal width'])],
                text=df['class'],
                marker=dict(color=index_vals,
                            showscale=False, # colors encode categorical variables
                            line_color='white', line_width=0.5)
                ))


fig.update_layout(
    title=dict(text='Iris Data set'),
    width=600,
    height=600,
)

fig.show()
```

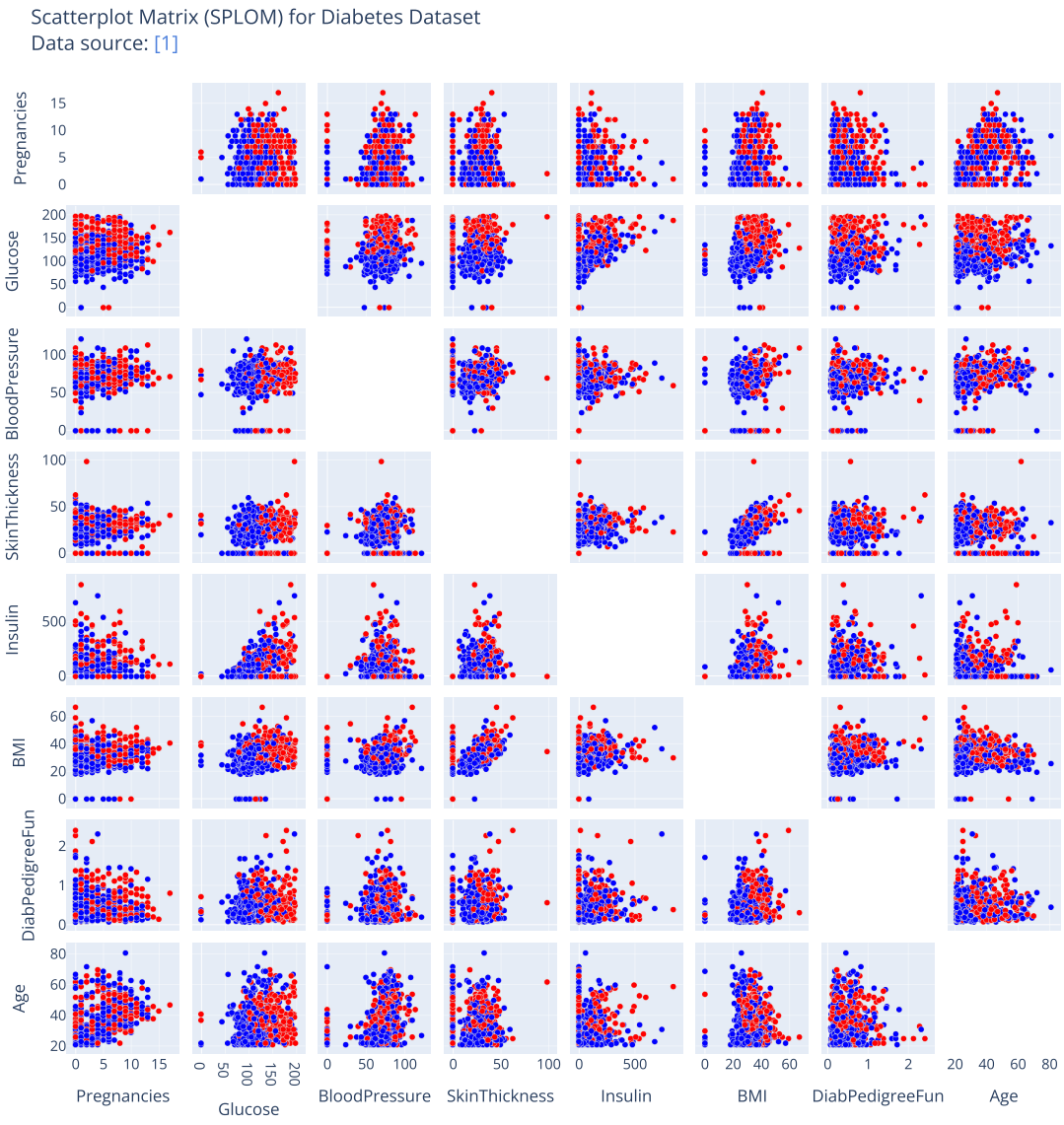


Splom for the diabetes dataset

Diabetes dataset is downloaded from kaggle (https://www.kaggle.com/uciml/pima-indians-diabetes-database/data). It is used to predict the onset of diabetes based on 8 diagnostic measures. The diabetes file contains the diagnostic measures for 768 patients, that are labeled as non-diabetic (Outcome=0), respectively diabetic (Outcome=1). The splom associated to the 8 variables can illustrate the strength of the relationship between pairs of measures for diabetic/nondiabetic patients.



```
import plotly.graph_objs as go
import pandas as pd

dfd = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/diabetes.csv')
textd = ['non-diabetic' if cl==0 else 'diabetic' for cl in dfd['Outcome']]

fig = go.Figure(data=go.Splom(
                  dimensions=[dict(label='Pregnancies', values=dfd['Pregnancies']),
                              dict(label='Glucose', values=dfd['Glucose']),
                              dict(label='BloodPressure', values=dfd['BloodPressure']),
                              dict(label='SkinThickness', values=dfd['SkinThickness']),
                              dict(label='Insulin', values=dfd['Insulin']),
                              dict(label='BMI', values=dfd['BMI']),
                              dict(label='DiabPedigreeFun', values=dfd['DiabetesPedigreeFunction']),
                              dict(label='Age', values=dfd['Age'])],
                  marker=dict(color=dfd['Outcome'],
                              size=5,
                              colorscale='Bluered',
                              line=dict(width=0.5,
                                        color='rgb(230,230,230)')),
                  text=textd,
                  diagonal=dict(visible=False)))

title = "Scatterplot Matrix (SPLOM) for Diabetes Dataset<br>Data source:"+\
        " <a href='https://www.kaggle.com/uciml/pima-indians-diabetes-database/data'>[1]</a>"
fig.update_layout(title=title,
                  dragmode='select',
                  width=1000,
                  height=1000,
                  hovermode='closest')

fig.show()
```

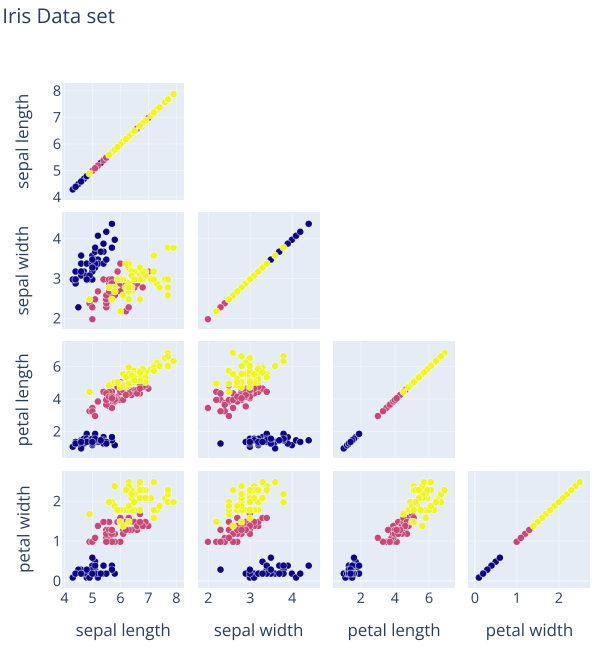


Hover Effects

New in 5.21

Set `hoversubplots='axis'` with `hovermode` set to `x`, `x unified`, `y`, or `y unified` for hover effects to appear across a column or row. For more on hover effects, see the Hover Text and Formatting (/python/hover-text-and-formatting/) page.

```
import plotly.graph_objects as go
import pandas as pd

df = pd.read_csv(
    "https://raw.githubusercontent.com/plotly/datasets/master/iris-data.csv"
)
index_vals = df["class"].astype("category").cat.codes

fig = go.Figure(
    data=go.Splom(
        dimensions=[
            dict(label="sepal length", values=df["sepal length"]),
            dict(label="sepal width", values=df["sepal width"]),
            dict(label="petal length", values=df["petal length"]),
            dict(label="petal width", values=df["petal width"]),
        ],
        showupperhalf=False,
        text=df["class"],
        marker=dict(
            color=index_vals,
            showscale=False,
            line_color="white",
            line_width=0.5,
        ),
    )
)


fig.update_layout(
    title=dict(text="Iris Data set"),
    hoversubplots="axis",
    width=600,
    height=600,
    hovermode="x",
)

fig.show()
```



Reference

or px.scatter_matrix() (https://plotly.com/python-api-reference/generated/plotly.express.scatter_matrix) or
n/reference/splom/ (https://plotly.com/python/reference/splom/) for more information and chart attribute options!

What About Dash?

Dash (https://dash.plot.ly/) is an open-source framework for building analytical applications, with no Javascript required, and it is tightly integrated with the Plotly graphing library.

Learn about how to install Dash at https://dash.plot.ly/installation (https://dash.plot.ly/installation).

Everywhere in this page that you see fig.show(), you can display the same figure in a Dash application by passing it to the figure argument of the Graph component (https://dash.plot.ly/dash-core-components/graph) from the built-in dash_core_components package like this:

```
import plotly.graph_objects as go # or plotly.express as px
fig = go.Figure() # or any Plotly Express function e.g. px.bar(...)
# fig.add_trace( ... )
# fig.update_layout( ... )

from dash import Dash, dcc, html

app = Dash()
app.layout = html.Div([
    dcc.Graph(figure=fig)
])

app.run(debug=True, use_reloader=False)  # Turn off reloader if inside Jupyter
```