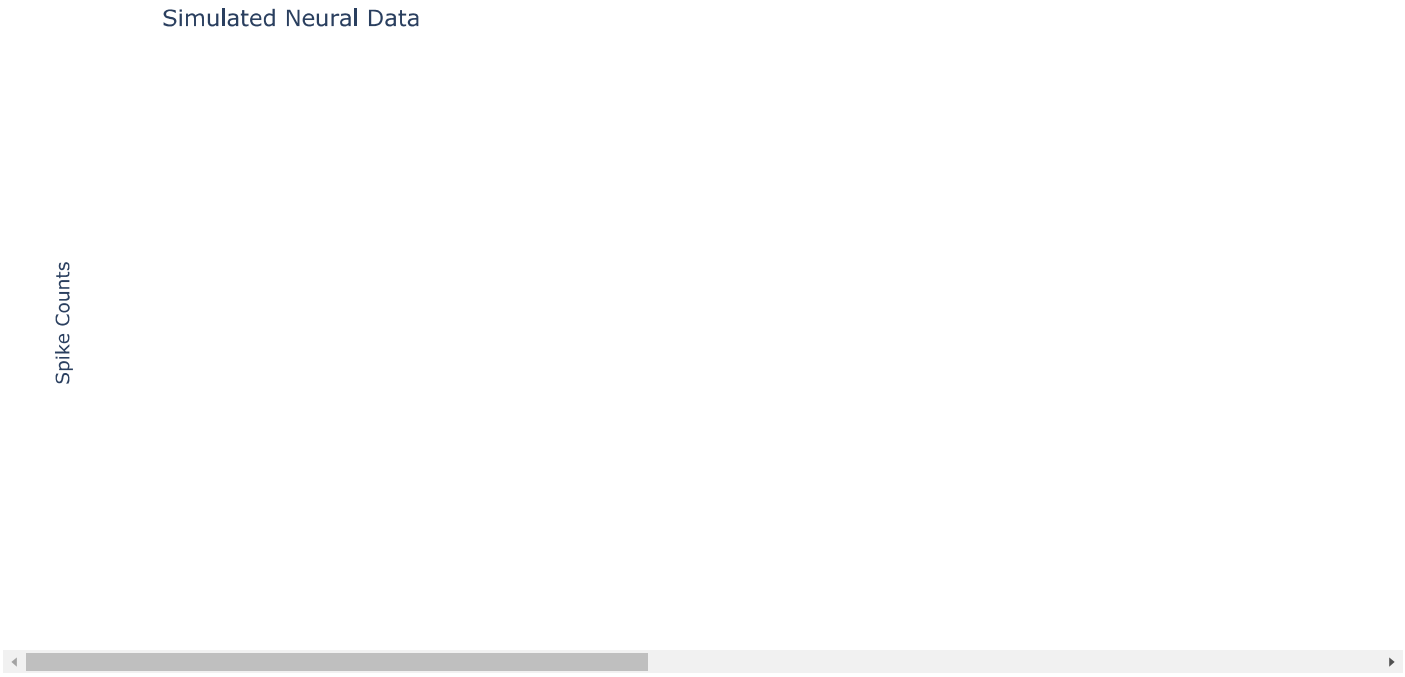
imports

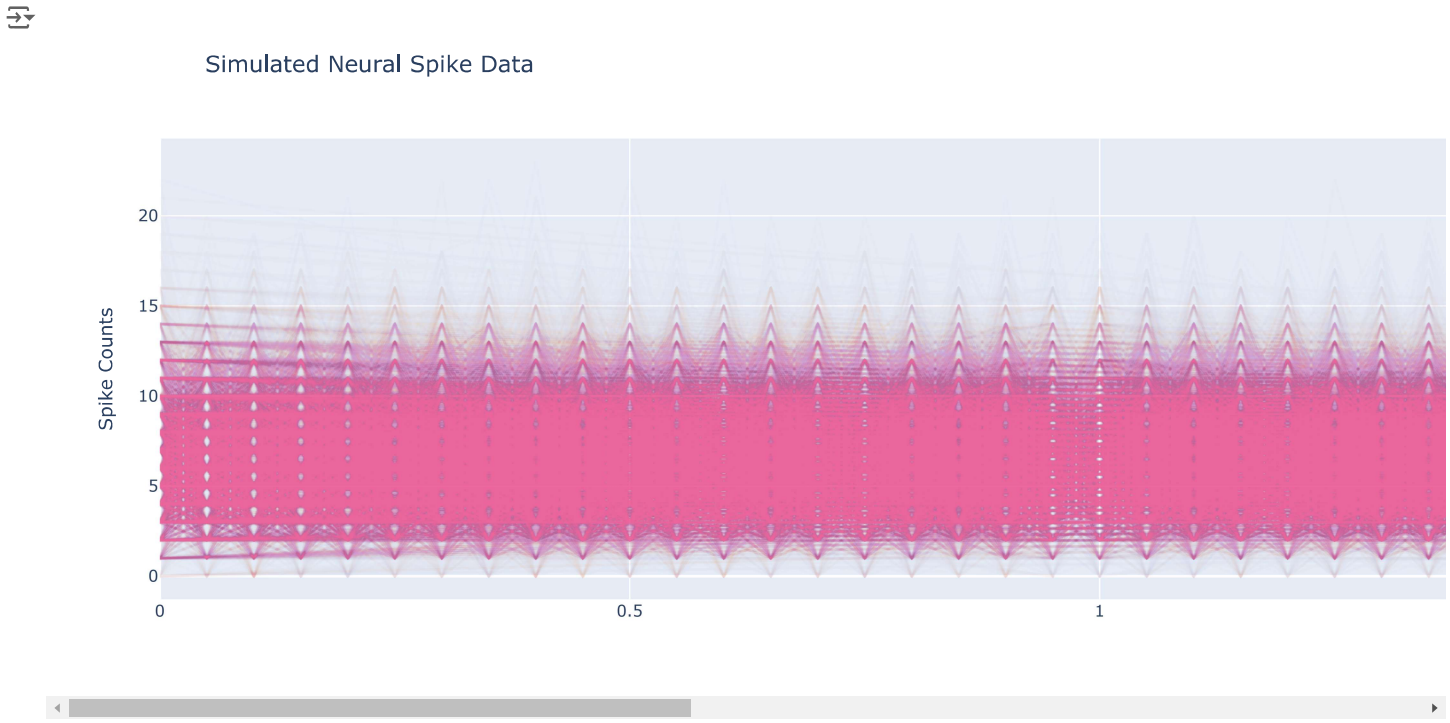
```
import numpy as np
import pandas as pd
import plotly.express as px
```

Simulate Neural Data

```
# Parameters for simulation
num_trials = 100  # Number of trials
num_neurons = 50  # Number of neurons
trial_duration = 3.0  # Trial duration in seconds
time_bin = 0.05  # Time bin size in seconds
num_bins = int(trial_duration / time_bin)

# Simulate spike trains for two conditions (left and right)
np.random.seed(42)
condition_1 = np.random.poisson(5, size=(num_trials, num_neurons, num_bins))  # Condition 1 (left)
condition_2 = np.random.poisson(8, size=(num_trials, num_neurons, num_bins))  # Condition 2 (right)

# Combine the conditions
data = np.concatenate([condition_1, condition_2], axis=0)  # Shape: (2*num_trials, num_neurons, num_bins)
labels = np.array([0] * num_trials + [1] * num_trials)  # 0 for left, 1 for right


# Assuming the code you provided is already executed and the variables
# num_trials, num_neurons, trial_duration, time_bin, num_bins, data, and labels
# are defined.

# Create time vector
time = np.arange(0, trial_duration, time_bin)

# Reshape data for plotting
data_reshaped = data.reshape(-1, num_bins)

# Create a DataFrame for plotting
df_plot = pd.DataFrame(data_reshaped)
df_plot['time'] = np.tile(time, 2 * num_trials)[:10000]
df_plot['trial'] = np.repeat(np.arange(2 * num_trials), num_bins)[:10000]
df_plot['condition'] = np.tile(labels, num_bins)[:10000]


# Plotting with Plotly Express
fig = px.line(df_plot, x='time', y=df_plot.columns[:-3], color='condition',
              labels={'value':'Spike Counts', 'variable':'Neuron', 'color':'Condition'},
              title='Simulated Neural Data')
fig.show()
```




```
import numpy as np
import pandas as pd
import plotly.express as px

# Create time vector
time = np.arange(0, trial_duration, time_bin)

# Reshape data for plotting
data_reshaped = data.reshape(-1, num_bins)

# Create a DataFrame for plotting
df_plot = pd.DataFrame(data_reshaped)
df_plot['time'] = np.tile(time, 2 * num_trials)[:10000]
df_plot['trial'] = np.repeat(np.arange(2 * num_trials), num_bins)[:10000]
df_plot['condition'] = np.tile(labels, num_bins)[:10000]

# Plotting with Plotly Express, adjusting opacity
fig = px.line(df_plot, x='time', y=df_plot.columns[:-3], color='condition',
              labels={'value':'Spike Counts', 'variable':'Neuron', 'color':'Condition'},
              title='Simulated Neural Spike Data')
# Update traces to set opacity
fig.update_traces(opacity=0.01) # Set opacity to 0.5 for both conditions after figure creation
fig.show()
```




```
from sklearn.mixture import GaussianMixture

# Flatten data across neurons and bins for clustering
data_flat = data.reshape(data.shape[0], -1)  # Shape: (2*num_trials, num_neurons*num_bins)

# Fit Gaussian Mixture Model (EM Algorithm)
gmm = GaussianMixture(n_components=2, random_state=42)
gmm.fit(data_flat)

# Predict cluster assignments
predicted_labels = gmm.predict(data_flat)

# Visualize the clustering
fig = px.scatter_3d(
    x=data_flat[:, 0], y=data_flat[:, 1], z=data_flat[:, 2],
    color=predicted_labels.astype(str),
    title="EM Clustering of Simulated Data",
    labels={"x": "Neuron 1", "y": "Neuron 2", "z": "Neuron 3"}
)
fig.show()
```

```
from sklearn.decomposition import FactorAnalysis

# Fit Factor Analysis
num_factors = 5  # Number of shared latent factors
fa = FactorAnalysis(n_components=num_factors, random_state=42)
fa.fit(data_flat)

# Transform data to obtain latent factors
latent_factors = fa.transform(data_flat)

# Visualize latent factors
latent_df = pd.DataFrame(latent_factors, columns=[f"Factor {i+1}" for i in range(num_factors)])
latent_df['Condition'] = labels

fig = px.scatter_matrix(
    latent_df,
    dimensions=[f"Factor {i+1}" for i in range(num_factors)],
    color='Condition',
    title="Latent Factors from GFCA",
    labels={"Condition": "Trial Condition"}
)
fig.show()
```