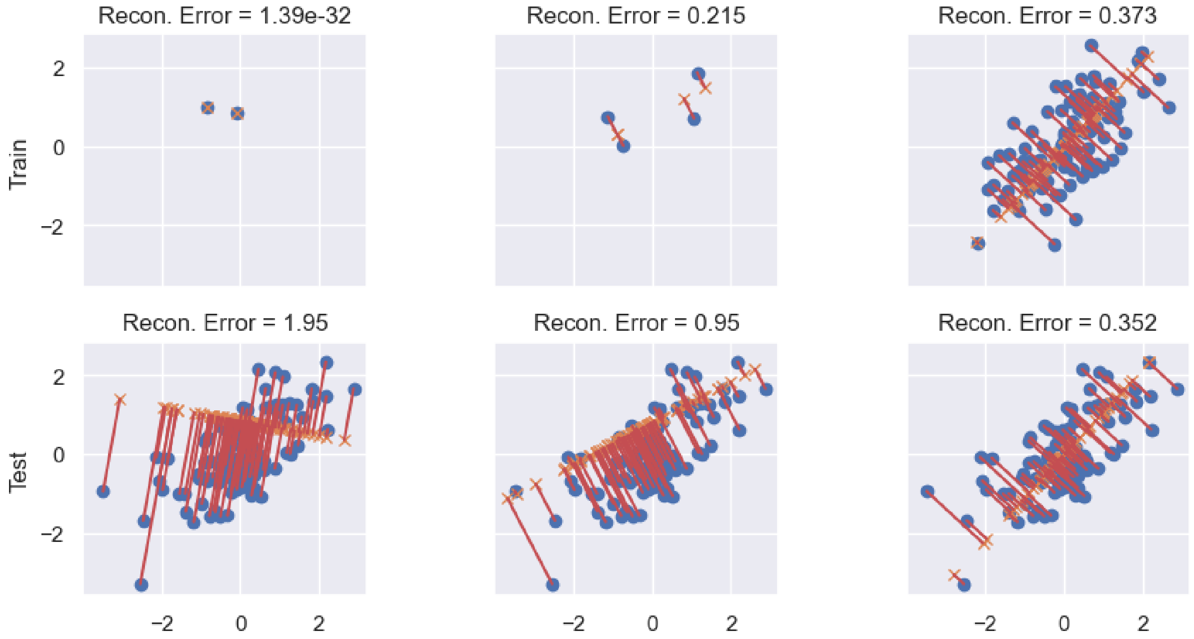
In [1]:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
from sklearn.decomposition import PCA
```

In [2]:

```
rng = np.random.RandomState(6)
n_features = 2

def generate_data(n):
    A = np.array([[1,0.4],[0.4,1]])
    #A = np.eye(n_features)
    return rng.randn(n, n_features) @ A

def plot_recon(X, X_recon, ax):
    squared_fro_norm = 1/X.shape[0] * np.linalg.norm(X-X_recon, ord='fro')

    ax.plot(*X.T, 'o')
    ax.plot(*X_recon.T, 'x')
    for x, xr in zip(X, X_recon):
        ax.plot(*np.array([x, xr]).T, '-r')
    ax.set_title(f'Recon. Error = {squared_fro_norm:.3g}')
    ax.set(adjustable='box', aspect='equal') # Make

n_test = 100
X_test = generate_data(n_test)

n_train_arr = [2, 4, 100]
fig, axes = plt.subplots(2, len(n_train_arr), figsize=np.array([4*3,3*2])*0
for i, (n_train, ax) in enumerate(zip(n_train_arr, axes.T)):
    X_train = generate_data(n_train)
    pca = PCA(n_components=1)
    pca.fit(X_train)
    X_train_recon = pca.inverse_transform(pca.transform(X_train))

    X_test_recon = pca.inverse_transform(pca.transform(X_test))

    plot_recon(X_train, X_train_recon, ax[0])
    plot_recon(X_test, X_test_recon, ax[1])
    if i == 0:
        ax[0].set_ylabel('Train')
        ax[1].set_ylabel('Test')

fig.tight_layout()
```


Recon. Error = 1.39e-32
Recon. Error = 0.215
Recon. Error = 0.373
Recon. Error = 1.95
Recon. Error = 0.95
Recon. Error = 0.352
Train
Test
2
0
−2
2
0
−2
−2
0
2
−2
0
2
−2
0
2