Programming assignment 10: Dimensionality Reduction¶

In [40]:

```
import numpy as np
import matplotlib.pyplot as plt

%matplotlib inline
```

PCA Task¶

Given the data in the matrix X your tasks is to:

- Calculate the covariance matrix Σ.
- Calculate eigenvalues and eigenvectors of Σ.
- Plot the original data X and the eigenvectors to a single diagram. What do you observe? Which eigenvector corresponds to the smallest eigenvalue?
- Determine the smallest eigenvalue and remove its corresponding eigenvector. The remaining eigenvector is the basis of a new subspace.
- Transform all vectors in X in this new subspace by expressing all vectors in X in this new basis.

The given data X¶

In [41]:

```
X = np.array([(-3,-2),(-2,-1),(-1,0),(0,1),
              (1,2),(2,3),(-2,-2),(-1,-1),
              (0,0),(1,1),(2,2), (-2,-3),
              (-1,-2),(0,-1),(1,0), (2,1),(3,2)])
```

Task 1: Calculate the covariance matrix Σ¶

In [42]:

```
def get_covariance(X):
    """Calculates the covariance matrix of the input data.

    Parameters
    ----------
    X : array, shape [N, D]
        Data matrix.

    Returns
    -------
    Sigma : array, shape [D, D]
```

```
        Covariance matrix

    """
    # TODO
    return np.cov(X.transpose())
```

Task 2: Calculate eigenvalues and eigenvectors of Σ.¶

In [43]:

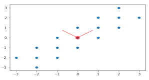
```
def get_eigen(S):
    """Calculates the eigenvalues and eigenvectors of the input matrix.

    Parameters
    ----------
    S : array, shape [D, D]
        Square symmetric positive definite matrix.

    Returns
    -------
    L : array, shape [D]
        Eigenvalues of S
    U : array, shape [D, D]
        Eigenvectors of S

    """
    # TODO
    return np.linalg.eigh(S)
```

Task 3: Plot the original data X and the eigenvectors to a single diagram.¶

In [44]:

```
# plot the original data
plt.scatter(X[:, 0], X[:, 1])

# plot the mean of the data
mean_d1, mean_d2 = X.mean(0)
plt.plot(mean_d1, mean_d2, 'o', markersize=10, color='red', alpha=0.5)

# calculate the covariance matrix
Sigma = get_covariance(X)

# calculate the eigenvector and eigenvalues of Sigma
```

```
L, U = get_eigen(Sigma)

plt.arrow(mean_d1, mean_d2, U[0, 0], U[0, 1], width=0.01, color='red', alpha=0.5)
plt.arrow(mean_d1, mean_d2, U[1, 0], U[1, 1], width=0.01, color='red', alpha=0.5);
```



What do you observe in the above plot? Which eigenvector corresponds to the smallest eigenvalue?

Write your answer here:

[YOUR ANSWER]

Task 4: Transform the data¶

Determine the smallest eigenvalue and remove its corresponding eigenvector. The remaining eigenvector is the basis of a new subspace. Transform all vectors in X in this new subspace by expressing all vectors in X in this new basis.

In [45]:

```
def transform(X, U, L):
    """Transforms the data in the new subspace spanned by the eigenvector
corresponding to the largest eigenvalue.

    Parameters
    ----------
    X : array, shape [N, D]
        Data matrix.
    L : array, shape [D]
        Eigenvalues of Sigma_X
    U : array, shape [D, D]
        Eigenvectors of Sigma_X

    Returns
    -------
    X_t : array, shape [N, 1]
        Transformed data

    """
    # TODO
    return None
```

In [46]:

```
X_t = transform(X, U, L)
```

Task SVD¶

Task 5: Given the matrix M find its SVD decomposition $M= U \cdot \Sigma \cdot V$ and reduce it to one dimension using the approach described in the lecture.¶

In [47]:

```
M = np.array([[1, 2], [6, 3],[0, 2]])
```

In [48]:

```
def reduce_to_one_dimension(M):
    """Reduces the input matrix to one dimension using its SVD decomposition.

    Parameters
    ----------
    M : array, shape [N, D]
        Input matrix.

    Returns
    -------
    M_t: array, shape [N, 1]
        Reduce matrix.

    """
    # TODO
    return None
```

In [49]:

```
M_t = reduce_to_one_dimension(M)
```