Standardizing a vector. If a vector x isn't constant (*i.e.*, at least two of its entries are different), we can standardize it, by subtracting its mean and dividing by its standard deviation. The resulting standardized vector has mean value zero and RMS value one. Its entries are called z-scores. We'll define a `standardize` function, and then check it with a random vector.

In [ ]:
```
def standardize(x):
    x_tilde = x - np.mean(x)
    return x_tilde/np.std(x_tilde)
x = np.random.random(100)
np.mean(x), np.std(x)
```

Out[ ]:
```
(0.568533078290501, 0.282467953801772)
```

In [ ]:
```
z = standardize(x)
np.mean(z), np.std(z)
```

Out[ ]:
```
(1.3322676295501878e-17, 1.0)
```

The mean or average value of the standardized vector `z` is very nearly zero.

3.4. Angle

Angle. Let's define a function that computes the angle between two vectors. We will call it `ang`.

In [ ]:
```
#Define angle function, which returns radians
ang = lambda x,y : np.arccos(x @ y /
↪  (np.linalg.norm(x)*np.linalg.norm(y)))
a = np.array([1,2,-1])
b = np.array([2,0,-3])
ang(a,b)
```

Out[ ]:
```
0.9689825515916383
```

In [ ]:
```
#Get angle in degrees
ang(a,b) * (360/(2*np.pi))
```

Out[ ]:
```
55.51861062801842
```

Correlation coefficient. The correlation coefficient between two vectors a and b (with nonzero standard deviation) is defined as

$$\rho = \frac{\tilde{a}^T \tilde{b}}{\|\tilde{a}\|\|\tilde{b}\|},$$

where $\tilde{a}$ and $\tilde{b}$ are the de-meaned versions of a and b, respectively. We can define our own function to compute the correlation. We calculate the correlation coefficients of the three pairs of vectors in Figure 3.8 in VMLS.

In [ ]:
```
def corr_coef(a,b):
    a_tilde = a - sum(a)/len(a)
    b_tilde = b - sum(b)/len(b)
    denom =  (np.linalg.norm(a_tilde) * np.linalg.norm(b_tilde))
    return (a_tilde @ b_tilde) /denom
a = np.array([4.4, 9.4, 15.4, 12.4, 10.4, 1.4, -4.6, -5.6, -0.6,
↪  7.4])
b = np.array([6.2, 11.2, 14.2, 14.2, 8.2, 2.2, -3.8, -4.8, -1.8,
↪  4.2])
corr_coef(a,b)
```

Out[ ]:
```
0.9678196342570432
```

In [ ]:
```
a = np.array([4.1, 10.1, 15.1, 13.1, 7.1, 2.1, -2.9, -5.9, 0.1,
↪  7.1])
b = np.array([5.5, -0.5, -4.5, -3.5, 1.5, 7.5, 13.5, 14.5, 11.5,
↪  4.5])
corr_coef(a,b)
```

Out[ ]:
```
-0.9875211120643734
```

In [ ]:
```
a = np.array([-5.0, 0.0, 5.0, 8.0, 13.0, 11.0, 1.0, 6.0, 4.0,
↪  7.0])
b = np.array([5.8, 0.8, 7.8, 9.8, 0.8, 11.8, 10.8, 5.8, -0.2,
↪  -3.2])
corr_coef(a,b)
```

Out[ ]:
```
0.004020976661367021
```

The correlation coefficients of the three pairs of vectors are 96.8%, −98.8%, and 0.4%.

3.5. Complexity

Let's check that the time to compute the correlation coefficient of two n-vectors is approximately linear in n.

In [ ]:
```
import time
x = np.random.random(10**6)
y = np.random.random(10**6)
start = time.time()
corr_coef(x,y)
end = time.time()
end - start
```

Out[ ]:
```
0.16412591934204102
```

In [ ]:
```
x = np.random.random(10**7)
y = np.random.random(10**7)
start = time.time()
corr_coef(x,y)
end = time.time()
end - start
```

Out[ ]:
```
1.6333978176116943
```

In [ ]:
```
x = np.random.random(10**8)
y = np.random.random(10**8)
start = time.time()
corr_coef(x,y)
end = time.time()
end - start
```

Out[ ]:
```
16.22579288482666
```