# Aufgabe 7

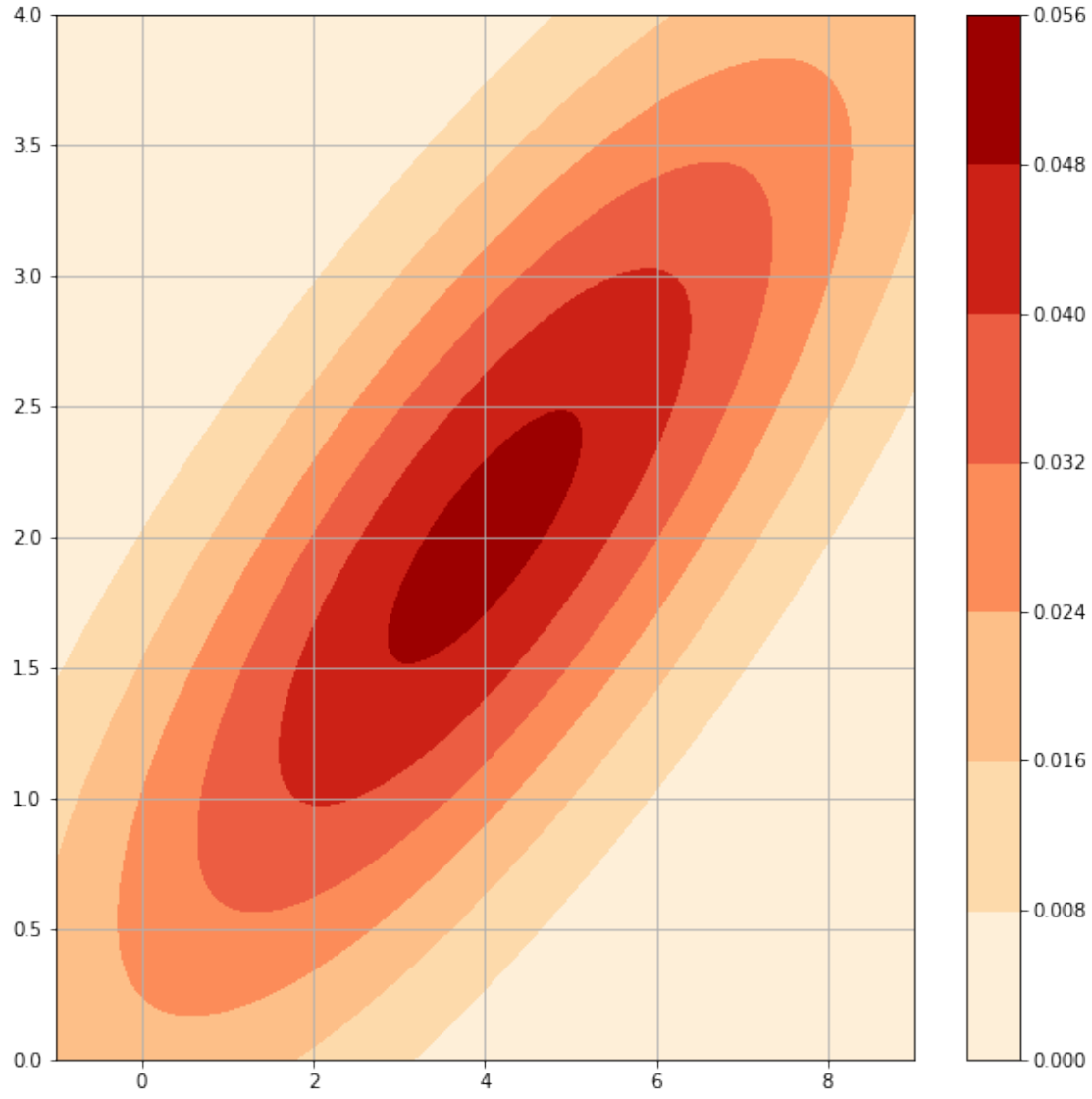
November 1, 2018

## 1 Teil b)

### 1.1 Bildliches Zeigen

kann auch Bildlich an den unterschiedlich gefärbten Ellipsen gesehen werden

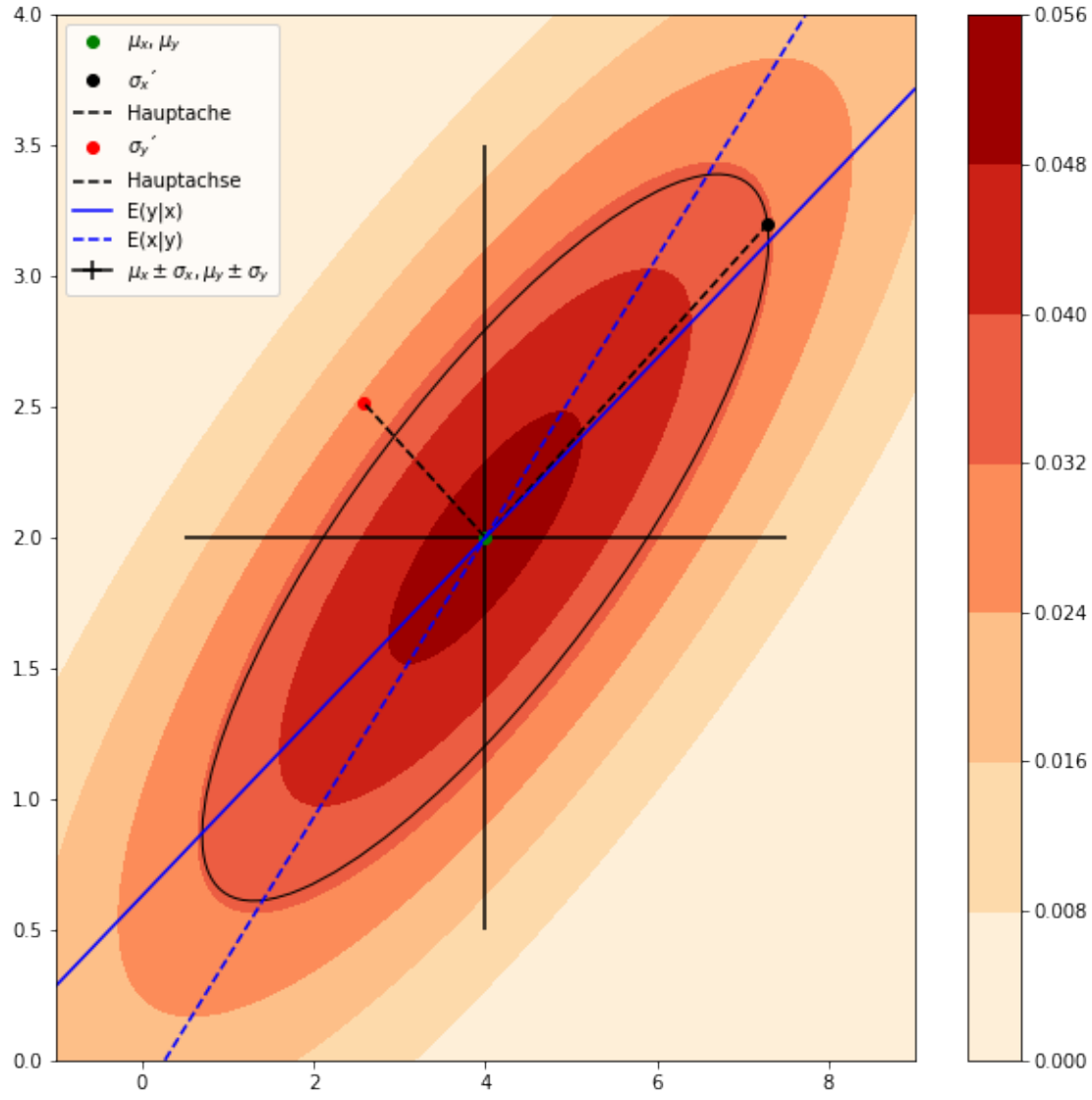
```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        from matplotlib.patches import Ellipse
        plt.rcParams['figure.figsize']=(10,10)
        nstd = 2
        mu_x = 4
        mu_y = 2
        rho = 0.8
        sigma_x = 3.5
        sigma_y = 1.5
        cov = 4.2
        def f(x, y, rho, mu_x, mu_y, sigma_x, sigma_y):
            return 1/(2*np.pi*sigma_x*sigma_y*np.sqrt(1-rho**2))*np.exp(
                -1/(2*(1 - rho**2))*((x - mu_x)**2 / (sigma_x**2)
                                     + (y - mu_y)**2 / (sigma_y**2) -
                                           2*rho*(x - mu_x)*(y - mu_y)/(sigma_x*sigma_y)))
        plt.grid()
        x = np.linspace(-1.0, 9.0,2000)
        y = np.linspace(0, 4.0,2000)
        X, Y = np.meshgrid(x, y)
        func = f(X, Y, rho, mu_x, mu_y, sigma_x, sigma_y)
        plt.contourf(X, Y, func, cmap='OrRd')
        #print(1/(np.sqrt(np.exp(1))*2*np.pi*sigma_x*sigma_y*np.sqrt(1-rho**2)))
        #print(1/(2*np.pi*sigma_x*sigma_y*np.sqrt(1-rho**2)))
        plt.colorbar()
        plt.show()
```




## 2 Teil c) - f) - Zeichnung

```
In [2]: ax = plt.subplot(111)
        theta=20.01512964
        ell = Ellipse(xy=(4, 2),
                      width=3.5*2, height=1.5,
                      angle=theta, color='black', label='1/\sqrt{e}')
        ell.set_facecolor('none')
        ax.add_artist(ell)
        plt.contourf(X, Y, func, cmap='OrRd')
        plt.errorbar(mu_x, mu_y, xerr=sigma_x, yerr=sigma_y, color='black',capthick=2,
                     label=r'$\mu_x \pm \sigma_x, \mu_y \pm \sigma_y$')
```

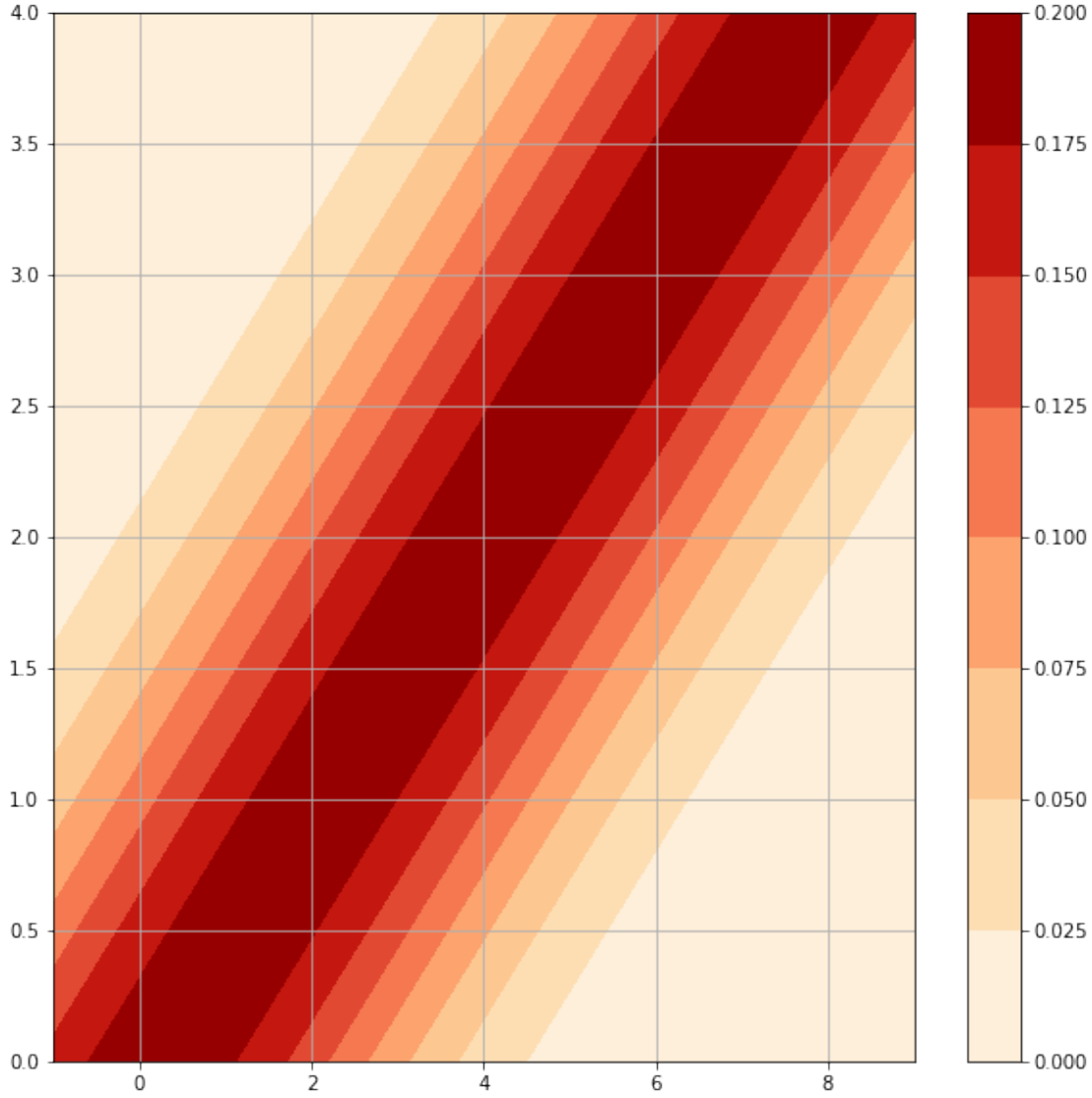
```
plt.plot(mu_x,mu_y,'go', label=r'$\mu_x, \mu_y$',linewidth=3)
sigma_x_x=[4,4+3.2886]
sigma_x_y=[2,2+1.1979]
plt.plot(4+3.2886, 2+1.1979,'ko',label=r"$\sigma_xť$")
plt.plot(sigma_x_x, sigma_x_y, 'k--', label='Hauptache')
sigma_y_x=[4,4-1.4095]
sigma_y_y=[2,2+0.513]
plt.plot(4-1.4095, 2+0.513,'ro', label=r"$\sigma_yť$")
plt.plot(sigma_y_x, sigma_y_y, 'k--', label='Hauptachse')
plt.plot(x, mu_y + rho*sigma_y*(x-mu_x)/sigma_x, 'b-', label='E(y|x)')
plt.plot( mu_x + rho*sigma_x*(y-mu_y)/sigma_y, y,  'b--', label='E(x|y)')

def f_x(x, y, rho, mu_x, mu_y, sigma_x, sigma_y):
    return 1/(sigma_x*np.sqrt(2*np.pi*(1-rho**2)))*np.exp(
        -1/(2*(1 - rho**2))*((x - mu_x) / (sigma_x)
                             - rho*(y - mu_y)**2 / (sigma_y**2))**2)

plt.colorbar()
plt.legend(loc='best')
plt.show()
```




Tut mir leid, weiSS nicht wie man die 3 Verteilungen in einen Plot bekommt.
Deshalb jetzt 3 Plots :(
f(x | y)

```
In [3]: def f_x(x, y, rho, mu_x, mu_y, sigma_x, sigma_y):
            return 1/(sigma_x*np.sqrt(2*np.pi*(1-rho**2)))*np.exp(
                -1/(2*(1 - rho**2))*((x - mu_x) / (sigma_x)
                                     - rho*(y - mu_y) / (sigma_y))**2)
        plt.clf()
        plt.grid()
        x = np.linspace(-1.0, 9.0,2000)
        y = np.linspace(0, 4.0,2000)
        X, Y = np.meshgrid(x, y)
        func = f_x(X, Y, rho, mu_x, mu_y, sigma_x, sigma_y)
```

```
    plt.contourf(X, Y, func, cmap='OrRd')
    plt.colorbar()
    plt.show()
```




f(y | x)

```
In [4]: def f_y(x, y, rho, mu_x, mu_y, sigma_x, sigma_y):
            return 1/(sigma_y*np.sqrt(2*np.pi*(1-rho**2)))*np.exp(
                -1/(2*(1 - rho**2))*((y - mu_y) / (sigma_y)
                                     - rho*(x - mu_x) / (sigma_x))**2)
        plt.clf()
        plt.grid()
        x = np.linspace(-1.0, 9.0,2000)
        y = np.linspace(0, 4.0,2000)
```

```
X, Y = np.meshgrid(x, y)
func = f_y(X, Y, rho, mu_x, mu_y, sigma_x, sigma_y)
plt.contourf(X, Y, func, cmap='OrRd')
plt.colorbar()
plt.show()
```