

## SMD-Abgabe

# 4. Übungsblatt

Lars Kolk

[lars.kolk@tu-dortmund.de](mailto:lars.kolk@tu-dortmund.de)

Julia Sobolewski

[julia.sobolewski@tu-dortmund.de](mailto:julia.sobolewski@tu-dortmund.de)

Jannine Salewski

[jannine.salewski@tu-dortmund.de](mailto:jannine.salewski@tu-dortmund.de)

Abgabe: 15.11.2018

18,5/20

TU Dortmund – Fakultät Physik

# 1 Aufgabe 10

Gegeben ist die Population  $P_0$  mit den Werten:

$$\mu_x = 0 \quad (1)$$

$$\mu_y = 3 \quad (2)$$

$$\sigma_x = 3,5 \quad (3)$$

$$\sigma_y = 2,6 \quad (4)$$

$$\rho = 0,9. \quad (5)$$

Ebenso ist eine Population  $P_1$  gegeben. Bei dieser sind die Werte in  $x$  gegeben durch:

$$\mu_x = 6 \quad (6)$$

$$\sigma_x = 3,5. \quad (7)$$

Für die Werte in  $y$  sind folgende Zusammenhänge gegeben:

$$E(y|x) = \mu_{y|x} = a + bx \text{ mit } a = -0,5 \text{ und } b = 0,6 \quad (8)$$

$$\text{Var}(y|x) = \sigma_{y|x}^2 = 1 \quad (9)$$

## 1.1 Teilaufgabe a)

### 1.1.1 Zeigen der 2D-Gaußverteilung

Die Formel für die bedingte Wahrscheinlichkeit der bivariaten Normalverteilung lautet:

$$f(y|x) = \frac{1}{\sqrt{2\pi}\sigma_y\sqrt{1-\rho^2}} \cdot \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{\tilde{y}}{\sigma_y} - \rho\frac{\tilde{x}}{\sigma_x}\right]^2\right). \quad (10)$$

wobei  $\tilde{x} = x - \mu_x$  und  $\tilde{y} = y - \mu_y$

Für die Wahrscheinlichkeitsdichtefunktion einer bivariaten Normalverteilung gilt dagegen:

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_x)^2}{\sigma_x^2} + \frac{(y-\mu_y)^2}{\sigma_y^2} - \frac{2\rho(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y}\right]\right), \quad (11)$$

Mit  $f(x,y) = g(x)f(y|x)$  folgt mit

$$g(x) = \frac{\sqrt{2(1-\rho^2)}}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}}\sqrt{\pi}\sigma_y \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2\right) \quad (12)$$

der Zusammenhang

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2 - \frac{1}{2(1-\rho^2)}\left(\frac{y-\mu_y}{\sigma_y} - \rho\frac{x-\mu_x}{\sigma_x}\right)^2\right). \quad (13)$$

Durch Umformungen erhält man die Gleichung (11), da:

$$-\frac{(x - \mu_x)^2}{\sigma_x^2} \frac{\rho^2}{2(1 - \rho^2)} - \frac{(x - \mu_x)^2}{2\sigma_x^2} = -\frac{(x - \mu_x)^2}{\sigma_x^2} \frac{1}{2(1 - \rho^2)} \quad (14)$$

### 1.1.2 Bestimmen der Werte

Für die Berechnung der Werte werden folgende Gleichungen genutzt:

$$\sigma_{y|x}^2 = (1 - \rho^2)\sigma_y^2 \quad (15)$$

$$E(y|x) = \rho \frac{\sigma_y}{\sigma_x} (x - \mu_x) + \mu_y \quad (16)$$

(17)

Bei einem Koeffizientenvergleich der Gleichung (16) mit (8) ergeben sich folgende Zusammenhänge:

$$b = \frac{\sigma_y}{\sigma_x} \cdot \rho \quad (18)$$

$$a = -\rho \frac{\sigma_y}{\sigma_x} \mu_x + \mu_y \quad (19)$$

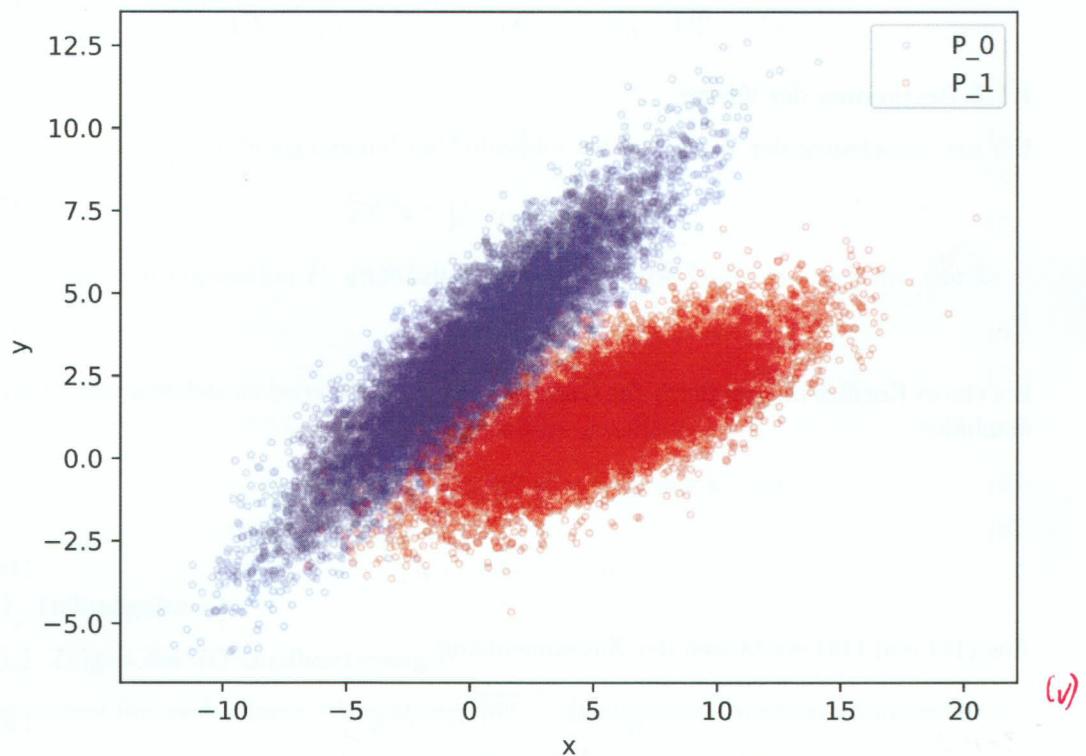
Aus (18) und (15) ergibt sich der Zusammenhang

$$p = \sqrt{\frac{1}{1 + \frac{\sigma_{y|x}^2}{b^2 \sigma_x^2}}} \approx 0,75 \quad (20)$$

womit sich  $\sigma_y = \sqrt{\frac{\sigma_{y|x}^2}{1-p^2}} \approx 1,51$  und  $\mu_y = \alpha + \frac{\rho \mu_x}{\sigma_x} \cdot \frac{\sigma_{y|x}}{\sqrt{1-p^2}} \approx 1,44$  ergeben.

### 1.2 Teilaufgabe b)

Es ergibt sich folgender Scatter-Plot:



**Abbildung 1:** Scatter-Plot

### 1.3 Teilaufgabe c)

es ergeben sich folgende Ergebnisse:

Ausgabe des Programms: Population 0

Mittelwerte von P0: [0.00781587 3.00831114]

Varianz von x0: 12.14051138946726

Varianz von y0: 6.654329986905046

Kovarianz cov(x, y): 8.095175577329087

Korrelation rho: 0.9006490919236851

Ausgabe des Programms: Population 1

Mittelwerte von P0: [6.00106294 1.41078796]  
Varianz von x0: 12.258004437206726  
Varianz von y0: 2.2870875071918366  
Kovarianz cov(x, y): 3.9966098123833347  
Korrelation rho: 0.7548149136498276

Ausgabe des Programms: Population 0 + Population 1

Mittelwerte von P0: [2.99233451 2.20887921]  
Varianz von x0: 21.37175112487808  
Varianz von y0: 5.207492960450249  
Kovarianz cov(x, y): 3.8264610565821275  
Korrelation rho: 0.3627128132883111

(✓)

a) fehlt

3,5/5

SMD - Blatt 4

Aufgabe 11

Population 0:  $\left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} 2 \\ 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} 1,5 \\ 2 \end{smallmatrix}\right), \left(\begin{smallmatrix} 2 \\ 2 \end{smallmatrix}\right), \left(\begin{smallmatrix} 2 \\ 3 \end{smallmatrix}\right), \left(\begin{smallmatrix} 3 \\ 3 \end{smallmatrix}\right)$

Population 1:  $\left(\begin{smallmatrix} 1,5 \\ 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} 2,5 \\ 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} 3,5 \\ 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} 2,5 \\ 2 \end{smallmatrix}\right), \left(\begin{smallmatrix} 3,5 \\ 2 \end{smallmatrix}\right), \left(\begin{smallmatrix} 4,5 \\ 2 \end{smallmatrix}\right)$

$$a) \vec{\mu} = \frac{1}{N} \begin{pmatrix} \sum_{i=1}^N x_i \\ \sum_{i=1}^N y_i \end{pmatrix}$$

$$\Rightarrow \vec{\mu}_0 = \begin{pmatrix} 23/12 \\ 2 \end{pmatrix}, \vec{\mu}_1 = \begin{pmatrix} 3 \\ 3/2 \end{pmatrix}$$

$$S_0 = \sum_{j=1}^{n_i} (\vec{x}_j - \vec{\mu}_0)(\vec{x}_j - \vec{\mu}_0)^T$$

$$S_0 = \begin{pmatrix} 42/144 & 1/12 \\ 1/12 & 1 \end{pmatrix} + \begin{pmatrix} 1/144 & -1/12 \\ -1/12 & 1 \end{pmatrix} + \begin{pmatrix} 25/144 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 3/144 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 1/144 & 1/12 \\ 1/12 & 1 \end{pmatrix} + \begin{pmatrix} 169/144 & 13/12 \\ 13/12 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 23/144 & 2 \\ 2 & 1 \end{pmatrix}$$

$$S_1 = \begin{pmatrix} 9/4 & 3/4 \\ 3/4 & 3/4 \end{pmatrix} + \begin{pmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{pmatrix} + \begin{pmatrix} 1/4 & -1/4 \\ -1/4 & 1/4 \end{pmatrix} + \begin{pmatrix} 1/4 & -1/4 \\ -1/4 & 1/4 \end{pmatrix} + \begin{pmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{pmatrix} + \begin{pmatrix} 9/4 & 3/4 \\ 3/4 & 1/4 \end{pmatrix}$$

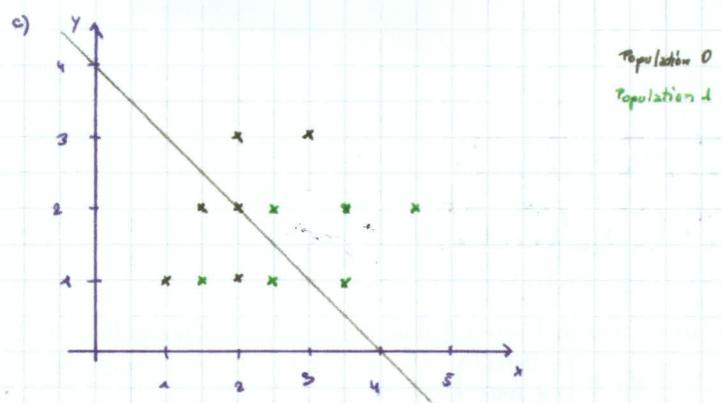
$$= \begin{pmatrix} 17/2 & 3/2 \\ 3/2 & 3/2 \end{pmatrix}$$

$$S_{10} = S_0 + S_1 = \begin{pmatrix} 185/144 & 3/2 \\ 3/2 & 17/2 \end{pmatrix} \Rightarrow S_{10}^{-1} = \begin{pmatrix} 0,1282 & -0,1216 \\ -0,1216 & 0,256 \end{pmatrix}$$

$$S_0 = (\vec{\mu}_0 - \vec{\mu}_1)(\vec{\mu}_0 - \vec{\mu}_1)^T = \begin{pmatrix} 169/144 & -13/12 \\ -13/12 & 1/4 \end{pmatrix} \quad \checkmark$$

$$b) \vec{\lambda} = S_{10}^{-1}(\vec{\mu}_0 - \vec{\mu}_1) = \begin{pmatrix} -370/1447 \\ 362/1447 \end{pmatrix} = \begin{pmatrix} -0,256 \\ 0,254 \end{pmatrix}$$

$$\vec{\lambda} = \lambda \vec{e}_2 = 0,360 \begin{pmatrix} -0,256 \\ 0,254 \end{pmatrix}$$



ii)  $\text{Projection} = \vec{\lambda}^T \vec{x}$

Population 0:  $-0,005; -0,716; 0,343; -0,042; 0,693; -0,042$

Population 1:  $-0,361; \text{[redacted]} -1,071; -1,781; -0,367; -1,076; -1,786$

c)  $\lambda_{\text{cut}} = -0,364 \rightarrow$  gewählt, weil Verhältnis zw. Effizienz und Reinheit 1 ist.

$$\begin{aligned} \rightarrow t_p &= 5 & \text{Reinheit: } \frac{t_p}{t_p + f_p} &= 0,83 \\ t_n &= 1 & \text{Effizienz: } \frac{t_p}{t_p + f_n} &= 0,93 \\ f_p &= 1 & & \\ f_n &= 5 & & \end{aligned}$$

9/5

## 2 Aufgabe 12

### 2.1 a)

Mittelwerte:

$$\vec{\mu}_0 = \begin{pmatrix} -0,027 \\ 2,980 \end{pmatrix}$$

$$\vec{\mu}_1 = \begin{pmatrix} 5,986 \\ 3,085 \end{pmatrix}$$

### 2.2 b)

Kovarianzmatrizen:

$$V_0 = \begin{pmatrix} 12,209 & 8,158 \\ 8,158 & 6,7223 \end{pmatrix}$$

$$V_1 = \begin{pmatrix} 12,352 & 7,411 \\ 7,411 & 5,477 \end{pmatrix}$$

Kombinierte Kovarianzmatrix:

$$V_{0,1} = \begin{pmatrix} 21,322 & 7,943 \\ 7,943 & 6,103 \end{pmatrix}$$

### 2.3 c)

Zu Berechnung der Fisher-Diskriminante, müssen zunächst die Streumatrizen berechnet werden:

$$S_0 = \begin{pmatrix} 122077,077 & 81575,940 \\ 81575,940 & 67221,910 \end{pmatrix}$$

$$S_1 = \begin{pmatrix} 123509.502 & 74100.151 \\ 74100.151 & 54767.673 \end{pmatrix}$$

Daraus wird die Gesamtstreuung  $S_W$  berechnet

$$S_W = S_0 + S_1 = \begin{pmatrix} 245586,579 & 155676,091 \\ 155676,091 & 121989,583 \end{pmatrix}$$

Die Fisher-Diskriminante lässt sich nun mit Hilfe der Formel

$$\vec{\lambda} = S_W^{-1}(\vec{\mu}_0 - \vec{\mu}_1) \quad (21)$$

berechnen:

$$\vec{\lambda} = \begin{pmatrix} -0,0001253 \\ 0,00015904 \end{pmatrix}$$

Diese lässt sich als Geradengleichung der Form  $\vec{\lambda} = \lambda \cdot \vec{e}_\lambda$  darstellen:, mit

$$\lambda = 0,00020247$$

und

$$\vec{e}_\lambda = \begin{pmatrix} -0,619 \\ 0,785 \end{pmatrix} \quad (22)$$

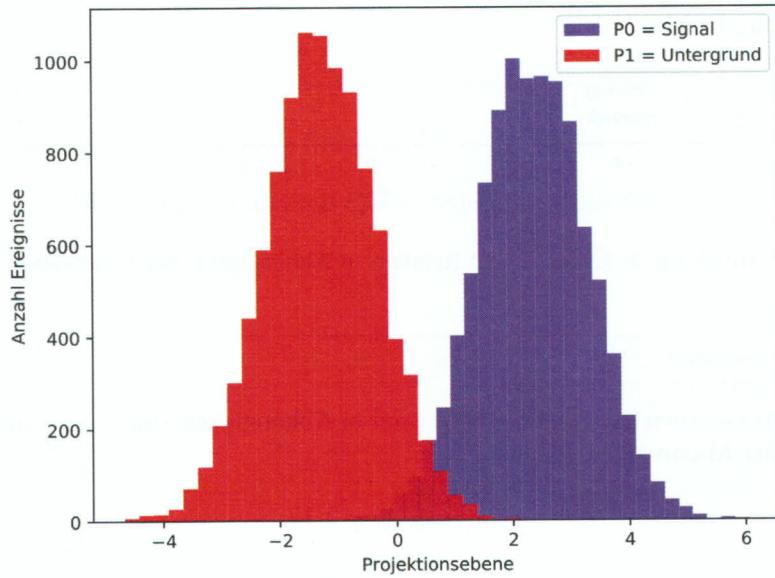
Der Einheitsvektor (22) lässt sich für die Projektion in den nächsten Aufgabenteilen verwenden.

## 2.4 d)

Zu Projektion der einzelnen Punkte auf die Gerade  $\vec{\lambda}$  wird folgende Formel verwendet:

$$P_\lambda(\vec{x}) = (\vec{x} \cdot \vec{e}_\lambda) \cdot \vec{e}_\lambda$$

wobei für die eindimensionale Verteilung nur der Betrag genommen wird, also nur  $(\vec{x} \cdot \vec{e}_\lambda)$ . Die eindimensionale Verteilung auf der Geraden ist in Abbildung 2 zu sehen.



**Abbildung 2:** Projektion auf die Gerade.

## 2.5 e)

Die Effizienz wird mit der Formel:

$$\text{Effizienz} = \frac{t_p}{t_p + f_p}$$

berechnet und die Reinheit mir:

$$\text{Reinheit} = \frac{t_p}{t_p + f_n} \quad (23)$$

Die Effizienz und die Reinheit in Abhängigkeit der Cut-Stelle  $\lambda_{cut}$  ist in Abbildung 3 dargestellt.

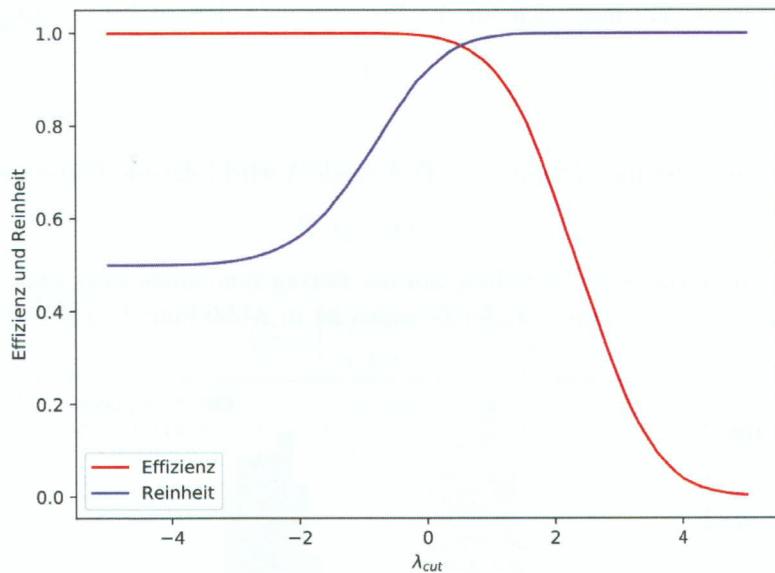


Abbildung 3: Effizienz und Reinheit in Abhängigkeit der Cut-Stelle.

## 2.6 f)

Das Verhältnis zwischen Signal und Untergrund in Abhängigkeit von  $\lambda_{cut}$  ist in Abbildung 4 zu sehen. Der Maximalwert liegt hier bei

$$\lambda_{cut, \text{Verhlt}} \approx 2,17$$

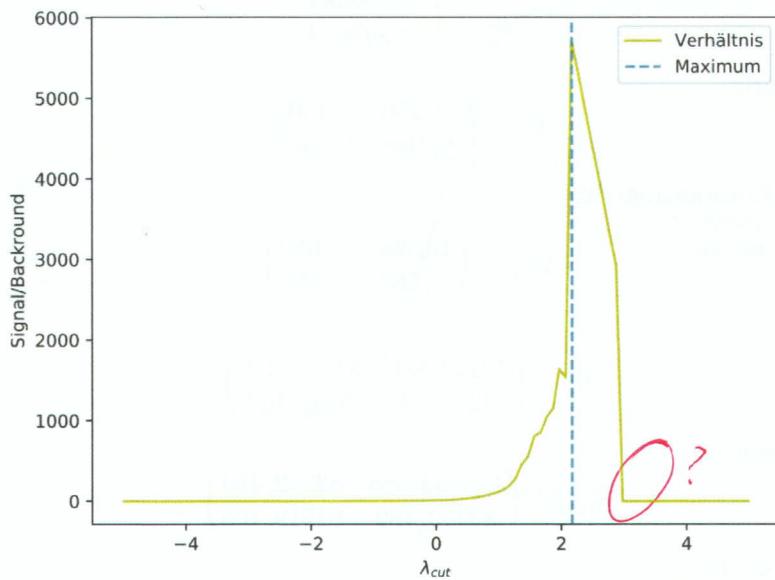
## 2.7 g)

Die Signifikanz  $\frac{S}{\sqrt{S+B}}$  in Abhängigkeit von  $\lambda_{cut}$  ist in Abbildung 5 dargestellt. Der Maximalwert liegt hier bei:

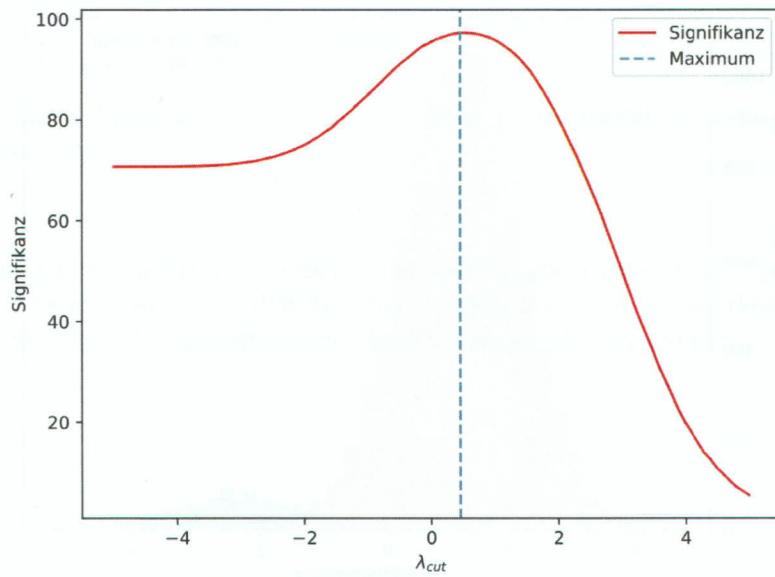
$$\lambda_{cut, \text{Signifikanz}} \approx 0,45$$

## 2.8 f)

Das ganze soll nun mit einem anderen Signal erneut untersucht werden:



**Abbildung 4:** Verhältnis S/B in Abhängigkeit der Cut-Stelle.



**Abbildung 5:** Signifikanz in Abhängigkeit der Cut-Stelle.

Mittelwert:

$$\vec{\mu}_2 = \begin{pmatrix} -0,0958 \\ 2,8788 \end{pmatrix}$$

Kovarianzmatrix:

$$V_2 = \begin{pmatrix} 12,236 & 8,160 \\ 8,160 & 6,758 \end{pmatrix}$$

Kombinierte Kovarianzmatrix:

$$V_{2,1} = \begin{pmatrix} 15,398 & 7,582 \\ 7,582 & 5,597 \end{pmatrix}$$

Streumatrix:

$$S_2 = \begin{pmatrix} 12223,886 & 81252,338 \\ 81252,338 & 6751,132 \end{pmatrix}$$

Gesamtstreuung:

$$S_{W2} = \begin{pmatrix} 135733,388 & 82252,489 \\ 82252,489 & 61519,105 \end{pmatrix}$$

Fisher-Diskriminante:

$$\vec{\lambda}_2 = \begin{pmatrix} -0,000254 \\ 0,000298 \end{pmatrix} = 0,374 \cdot 10^{-3} \begin{pmatrix} -0,603 \\ 0,797 \end{pmatrix} = \lambda_2 \cdot \vec{e}_{\lambda_2}$$

Die eindimensionale Verteilung ist in Abbildung 6 dargestellt.

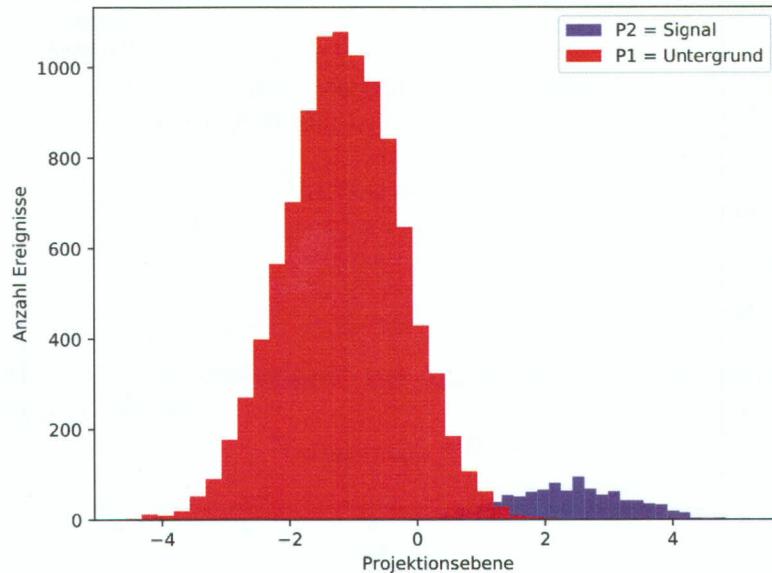
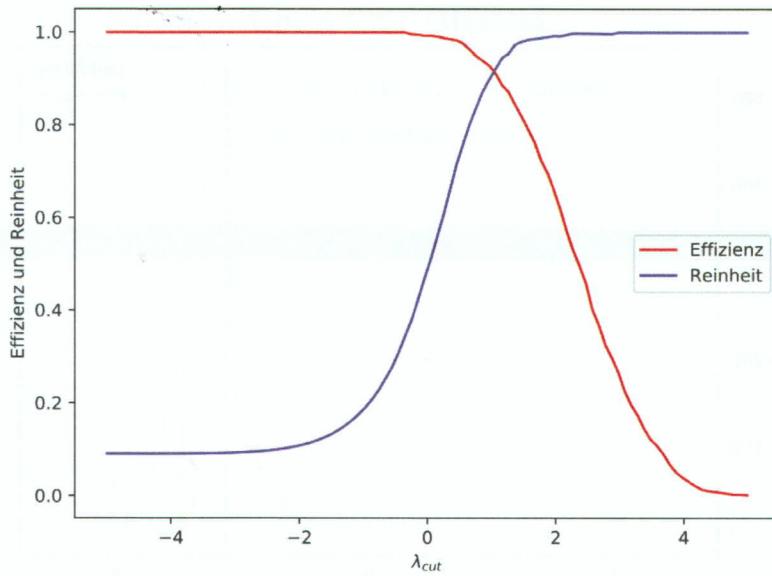


Abbildung 6: Eindimensionale Verteilung mit P2.



**Abbildung 7:** Effizienz und Reinheit der zweiten Verteilung.

Die Effizienz und die Reinheit sind in Abbildung 7 zu sehen.

Das Verhältnis  $S/B$  in Abhängigkeit von der Cut-Stelle ist in Abbildung 8 dargestellt.

Das Maximum liegt bei:

$$\lambda_{\text{cut}, \text{Verhaeltnis2}} \approx 2,27$$

Die Signifikanzin Abhängigkeit von der Cut-Stelle ist in Abbildung 9 dargestellt. Das Maximum liegt bei:

$$\lambda_{\text{cut}, \text{Signifikanz2}} \approx 1,06$$

Die Trennung funktioniert für kleinere oder gleichgroße Untergründe, im Bezug auf das Signal, besser. Dies ist an den Maxima der Signifikanzkurve zu erkennen, da das Maximum der ersten Verteilung deutlich höher liegt, als die der zweiten.

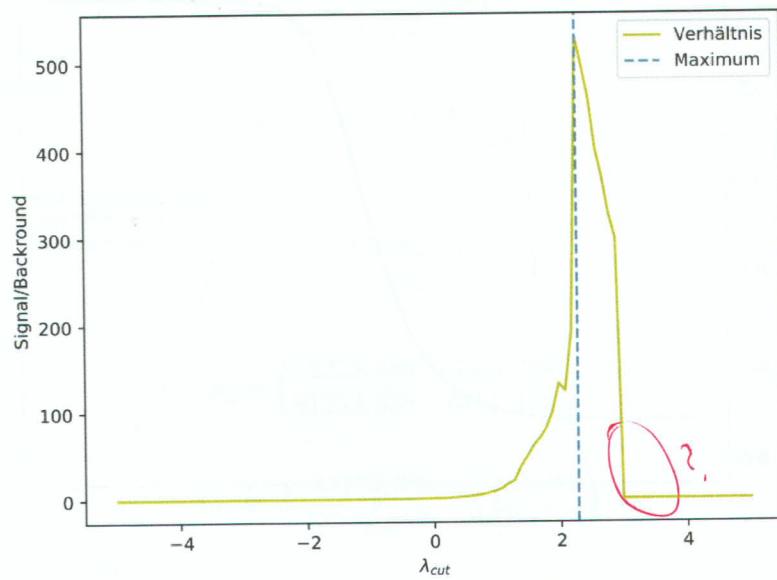


Abbildung 8: Verhältnis in Abhängigkeit der Cut-Stelle der zweiten Verteilung.

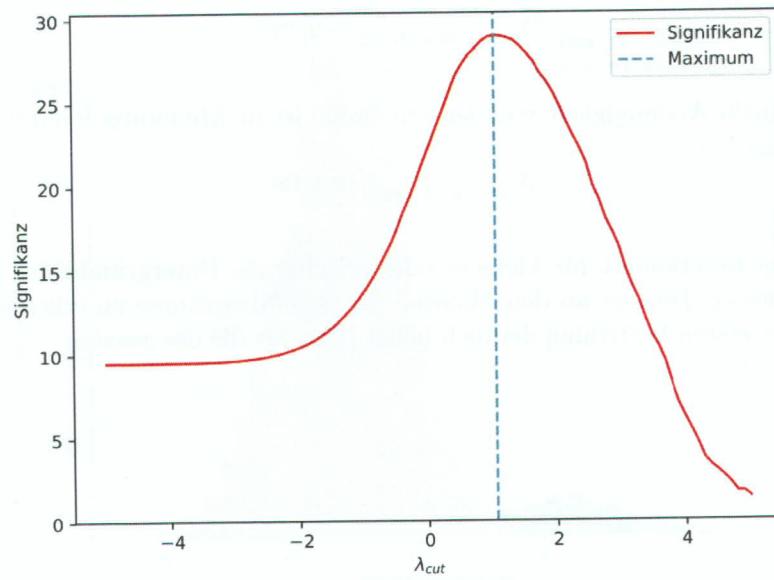


Abbildung 9: Signifikanz in Abhängigkeit der Cut-Stelle der zweiten Verteilung.

## Code fuer Blatt04

Kolk, Sobolewski, Salewski

16. November 2018

```
./B/7/Blatt04_Kolk_Sobolewski_Salewski/Aufgabe10.py
```

1 import numpy as np  
2 import matplotlib.pyplot as plt  
3 np.random.seed = 42  
4  
5 def shift(list, n):  
6 n = n % len(list)  
7 return list[n:] + list[:n]  
8  
9 #Funktion zum berechnen des Mittelwerts  
10 def mean(numbers):  
11 return float(sum(numbers)) / max(len(numbers), 1) ? np.mean()  
12  
13 #a)  
14 covP0 = [[3.5\*\*2, 0.9\*3.5\*2.6], [0.9\*3.5\*2.6, 2.6\*\*2]]  
15 P0 = np.random.multivariate\_normal([0, 3], covP0, size=10000)  
16 x0, y0 = zip(\*P0)  
17 plt.plot(x0, y0, 'b.', label='P\_0', alpha=0.07)  
18 covP1 = [[3.5\*\*2, 0.75\*3.5\*1.51], [0.75\*3.5\*1.51, 1.51\*\*2]]  
19 P1 = np.random.multivariate\_normal([6, 1.44], covP1, size=10000)  
20 x1, y1 = zip(\*P1)  
21 plt.plot(x1, y1, 'r.', label='P\_1', alpha=0.1)  
22 plt.legend()  
23 plt.xlabel('x')  
24 plt.ylabel('y')  
25 plt.savefig("Populationen.pdf")  
26  
27 #b)  
28 print("Population 0")  
29 m0=np.array([np.mean(x0), np.mean(y0)])  
30 a, b=zip(\*P0)  
31 print("Mittelwerte von P0: ", m0)  
32 print("Varianz von x0: ", np.var(x0))  
33 print("Varianz von y0: ", np.var(y0)) default  
34 print("Kovarianz cov(x, y): ", np.cov(a,b, ddof=1)[0][1])  
35 print("Korrelation rho: ", np.cov(a,b, ddof=1)[0][1]/np.sqrt(np.var(x0)\*np.var(y0)), "\n \n")  
36  
37 print("Population 1")  
38 m1=np.array([np.mean(x1), np.mean(y1)])  
39 a, b=zip(\*P1)  
40 print("Mittelwerte von P0: ", m1)  
41 print("Varianz von x0: ", np.var(x1))  
42 print("Varianz von y0: ", np.var(y1))  
43 print("Kovarianz cov(x, y): ", np.cov(a,b, ddof=1)[0][1])  
44 print("Korrelation rho: ", np.cov(a,b, ddof=1)[0][1]/np.sqrt(np.var(x1)\*np.var(y1)), "\n \n")  
45  
46 print("Population 0 + Population 1")  
47 P= np.vstack((P0,P1))  
48 x, y = zip(\*P)  
49 m2=np.array([np.mean(x), np.mean(y)])  
50 print("Mittelwerte von P0: ", m2)  
51 print("Varianz von x0: ", np.var(x))  
52 print("Varianz von y0: ", np.var(y))  
53 print("Kovarianz cov(x, y): ", np.cov(x,y, ddof=1)[0][1])  
54 print("Korrelation rho: ", np.cov(x,y, ddof=1)[0][1]/np.sqrt(np.var(x)\*np.var(y)), "\n \n")  
55

50, 50, 50,

```
./B/7/Blatt04_Kolk_Sobolewski_Salewski/Aufgabel2.py
```

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 import pandas as pd
4 from scipy.optimize import curve_fit
5 import uncertainties.unumpy as unp
6 from uncertainties import ufloat
7 from scipy.stats import stats
8 from scipy import linalg
9
10 P0_p = pd.read_hdf('zwei_populationen.h5', key='P_0_10000')
11 P1_p = pd.read_hdf('zwei_populationen.h5', key='P_1')
12
13
14 # Aufgabenteil a) #####
15 # Mittelwerte bilden
16 mue0=np.array([P0_p.x.mean(), P0_p.y.mean()])
17 mue1=np.array([P1_p.x.mean(), P1_p.y.mean()])
18
19 print('Mittelwerte:')
20 print('mue0 = ', mue0)
21 print('mue1 = ', mue1)
22 print()
23
24 # Aufgabenteil b) #####
25 # Kovarianzmatrizen berechnen
26 V_P0 = P0_p.cov()
27 V_P1 = P1_p.cov()
28 V_P01 = P0_p.append(P1_p).cov()
29 print('Kovarianzmatrizen:')
30 print('V_P0 = ')
31 print(V_P0)
32 print('V_P1 = ')
33 print(V_P1)
34 print('V_P01 = ')
35 print(V_P01)
36 print()
37
38 # Aufgabenteil c) #####
39 # Lineare Fisher-Diskriminante L
40
41 # Definition des Vektorprodukts
42 def vprod(x):
43     y = np.atleast_2d(x)
44     return np.dot(y.T, y)
45
46 S0 = np.zeros(2)
47 for index, row in (P0_p - mue0).iterrows():
48     S0 = S0 + vprod(row)
49
50 S1 = np.zeros(2)
51 for index, row in (P1_p - mue1).iterrows():
52     S1 = S1 + vprod(row)
53
54 # Addition der Matritzen
55 SW = S0 + S1
56 print('S0 = ', S0)
57 print('S1 = ', S1)
58 print('SW = ', SW)
59 print()
60
61 L = np.dot(np.linalg.inv(SW), (mue0-mue1))
62 print('L = ', L)
63 # Geradengleichung noch ausrechnen!!!
64 norm = np.linalg.norm(L)
65 print("Normierung: ", norm)
66 L_norm = L/norm
67 print('L_norm = ', L_norm)
68
69 # Aufgabenteil d) #####
70 Projektion_0 = np.array([])
71 for index, row in (P0_p).iterrows():

```

```

72     Projektion_0 = np.append(Projektion_0, np.vdot(row, L_norm))
73
74 Projektion_1 = np.array([])
75 for index, row in (P1_p).iterrows():
76     Projektion_1 = np.append(Projektion_1, np.vdot(row, L_norm))
77
78 plt.hist(Projektion_0, label='P0 = Signal', bins=30, color='b', alpha=0.8)
79 plt.hist(Projektion_1, label='P1 = Untergrund', bins=30, color='r', alpha=0.8)
80 plt.xlabel('Projektionsebene')
81 plt.ylabel('Anzahl Ereignisse')
82 plt.legend(loc="best")
83 plt.tight_layout()
84 plt.savefig('Projektionen.pdf')
85 plt.clf()
86
87 # Aufgabenteil e), f) und g) #####
88 lcut = np.linspace(-5,5, 100)
89 Back = Projektion_1
90 Sig = Projektion_0
91
92 #leere Arrays erzeugen
93 eff = np.zeros(len(lcut))
94 rein = np.zeros(len(lcut))
95 ver = np.zeros(len(lcut))
96 sign = np.zeros(len(lcut))
97 #Effizienz und Reinheit für jedes l_cut berechnen
98 for i in range(len(lcut)):
99     tp = len(Sig[Sig > lcut[i]])
100    fp = len(Back[Back > lcut[i]])
101    fn = len(Sig[Sig <= lcut[i]])
102
103    eff[i] = tp/(tp + fn)
104    rein[i] = tp/(tp + fp)
105    if fp != 0:
106        ver[i] = tp/fp
107    sign[i] = tp/(np.sqrt(tp + fp))
108
109 # Maximum des Verhältnisses berechnen
110 lcut_maxv = lcut[np.argmax(ver)]
111 print('lcut_maxv = ', lcut_maxv)
112
113 # Maximum der Signifikanz berechnen
114 lcut_maxs = lcut[np.argmax(sign)]
115 print('lcut_maxs = ', lcut_maxs)
116
117
118 # Plots
119 plt.plot(lcut, eff, 'r-', label='Effizienz')
120 plt.plot(lcut, rein, 'b-', label='Reinheit')
121 plt.xlabel(r"\lambda_{cut}")
122 plt.ylabel('Effizienz und Reinheit')
123 plt.legend(loc="best")
124 plt.tight_layout()
125 plt.savefig('EffizienzReinheit.pdf')
126 plt.clf()
127
128 plt.plot(lcut, ver, 'y-', label='Verhältnis')
129 plt.xlabel(r"\lambda_{cut}")
130 plt.ylabel('Signal/Background')
131 plt.axvline(x=lcut_maxv, linestyle='--', label='Maximum')
132 plt.legend(loc="best")
133 plt.tight_layout()
134 plt.savefig('Verhältnis.pdf')
135 plt.clf()
136
137 plt.plot(lcut, sign, 'r-', label='Signifikanz')
138 plt.axvline(x=lcut_maxs, linestyle='--', label='Maximum')
139 plt.xlabel(r"\lambda_{cut}")
140 plt.ylabel('Signifikanz')
141 plt.legend(loc="best")
142 plt.tight_layout()

```

```

141 plt.savefig('Signifikanz.pdf')
142 plt.clf()
143
144 # Aufgabenteil h) #####
145 P2_p = pd.read_hdf('zwei_populationen.h5', key='P_0_1000')
146
147 mue2=np.array([P2_p.x.mean(), P2_p.y.mean()])
148 print('Mittelwerte:')
149 print('mue2 = ', mue2)
150
151 V_P2 = P2_p.cov()
152 V_P21 = P2_p.append(P1_p).cov()
153
154 print('Kovarianzmatrizen:')
155 print('V_P2 = ')
156 print(V_P2)
157 print('V_P21 = ')
158 print(V_P21)
159
160 print()
161
162 S2 = np.zeros(2)
163 for index, row in (P2_p - mue2).iterrows():
164     S2 = S2 + vprod(row)
165
166 # Addition der Matritzen
167 SW2 = S2 + S1
168 print('S2 = ', S2)
169 print('SW2 = ', SW2)
170
171 print()
172
173 L2 = np.dot(np.linalg.inv(SW2), (mue2-mue1))
174 print('L2 = ', L2)
175 # Geradengleichung noch ausrechnen!!!
176 norm2 = np.linalg.norm(L2)
177 print("Normierung2: ", norm2)
178 L2_norm = L2/norm2
179 print('L2_norm = ', L2_norm)
180
181
182 Projektion2_2 = np.array([])
183 for index, row in (P2_p).iterrows():
184     Projektion2_2 = np.append(Projektion2_2, np.vdot(row, L2_norm))
185
186 Projektion2_1 = np.array([])
187 for index, row in (P1_p).iterrows():
188     Projektion2_1 = np.append(Projektion2_1, np.vdot(row, L2_norm))
189
190 plt.hist(Projektion2_2, label='P2 = Signal', bins=30, color='b', alpha=0.8)
191 plt.hist(Projektion2_1, label='P1 = Untergrund', bins=30, color='r', alpha=0.8)
192 plt.xlabel('Projektionsebene')
193 plt.ylabel('Anzahl Ereignisse')
194 plt.legend(loc="best")
195 plt.tight_layout()
196 plt.savefig('Projektionen2.pdf')
197 plt.clf()
198
199
200 Back2 = Projektion2_1
201 Sig2 = Projektion2_2
202
203 #leere Arrays erzeugen
204 eff2 = np.zeros(len(lcut))
205 rein2 = np.zeros(len(lcut))
206 ver2 = np.zeros(len(lcut))
207 sign2 = np.zeros(len(lcut))
208 #Effizienz und Reinheit für jedes l_cut berechnen
209 for i in range(len(lcut)):
210     tp2 = len(Sig2[Sig2 > lcut[i]])
211     fp2 = len(Back2[Back2 > lcut[i]])
212     fn2 = len(Sig2[Sig2 <= lcut[i]])
213

```

```

214     eff2[i] = tp2/(tp2 + fn2)
215     rein2[i] = tp2/(tp2 + fp2)
216     if fp2 != 0:
217         ver2[i] = tp2/fp2
218     sign2[i] = tp2/(np.sqrt(tp2 + fp2))
219
220 # Maximum des Verhältnisses berechnen
221 lcut2_maxv = lcut[np.argmax(ver2)]
222 print('lcut2_maxv = ', lcut2_maxv )
223
224 # Maximum der Signifikanz berechnen
225 lcut2_maxs = lcut[np.argmax(sign2)]
226 print('lcut2_maxs = ', lcut2_maxs)
227
228
229 # Plots
230 plt.plot(lcut, eff2, 'r-', label='Effizienz')
231 plt.plot(lcut, rein2, 'b-', label='Reinheit')
232 plt.xlabel(r"\lambda_{cut}")
233 plt.ylabel('Effizienz und Reinheit')
234 plt.legend(loc="best")
235 plt.tight_layout()
236 plt.savefig('EffizienzReinheit2.pdf')
237 plt.clf()
238
239 plt.plot(lcut, ver2, 'y-', label='Verhältnis')
240 plt.xlabel(r"\lambda_{cut}")
241 plt.ylabel('Signal/Background')
242 plt.axvline(x=lcut2_maxv, linestyle='--', label='Maximum')
243 plt.legend(loc="best")
244 plt.tight_layout()
245 plt.savefig('Verhältnis2.pdf')
246 plt.clf()
247
248 plt.plot(lcut, sign2, 'r-', label='Signifikanz')
249 plt.axvline(x=lcut2_maxs, linestyle='--', label='Maximum')
250 plt.xlabel(r"\lambda_{cut}")
251 plt.ylabel('Signifikanz')
252 plt.legend(loc="best")
253 plt.tight_layout()
254 plt.savefig('Signifikanz2.pdf')
255 plt.clf()

```