

VERSUCH 44

Röntgenreflektometrie an einem Polystyrol-beschichteten Silicium-Wafer

Katharina Brägelmann
katharina.braegelmann@tu-dortmund.de

Lars Kolk
lars.kolk@tu-dortmund.de

Durchführung: 20.01.2020

Abgabe: 04.03.2020

TU Dortmund – Fakultät Physik

Inhaltsverzeichnis

1 Zielsetzung	3
2 Theorie	3
2.1 Grundlagen	3
2.2 Fresnel Formeln	3
2.3 Mehrschichtsysteme	4
2.4 Rauigkeit	5
2.5 Geometriefaktor und Geometriewinkel	6
3 Aufbau und Durchführung des Versuchs	6
3.1 Justage	7
3.2 Messung	8
4 Auswertung	9
4.1 Vorbereitung	9
4.2 Auswertung der Vermessung eines mit Polystyrol beschichteten Silicium-Wafers	12
5 Diskussion	14
Literatur	15
6 Anhang	16

1 Zielsetzung

In diesem Versuch sollen mithilfe der Röntgenreflektometrie die Dichte, Rauigkeit und Schichtdicke eines dünnen Polystyrolfilms untersucht werden.

2 Theorie

2.1 Grundlagen

Brechung findet statt, wenn eine elektromagnetische Welle mit einem elektrischen Feldvektor der Form

$$\vec{E}(\vec{r}) = \vec{E}_0 e^{i\vec{k}\vec{r}}$$

($\vec{k} \hat{=}$ Wellenvektor, $\vec{r} \hat{=}$ Ortsvektor)

von einem Medium mit Brechungsindex n_1 in ein Medium mit Brechungsindex n_2 ($n_1 \neq n_2$) übergeht. Bei der in diesem Versuch verwendeten Röntgenstrahlung handelt es sich dabei um eine elektromagnetische Welle mit einer Wellenlänge zwischen $\lambda = 0,1 \text{ \AA}$ und $\lambda = 10 \text{ \AA}$.

Der Brechungsindex kann als

$$n = 1 - \delta + i\beta$$

($\delta \hat{=}$ Korrekturterm ($O(10^{-6})$), $\beta \hat{=}$ Absorption ($O(10^{-7})$ für $E = 6 \text{ keV}$)

geschrieben werden und ist für Röntgenstrahlung kleiner als eins. Aus dem Snelliusschen Brechungsgesetz

$$\frac{n_1}{n_2} = \frac{\cos \alpha_2}{\cos \alpha_1}$$

und der Annahme, dass es sich bei der Grenzfläche der Medien um eine homogene Ebene handelt, ergibt sich ein kritischer Winkel α_C , bei dem es zur Totalreflexion kommt. Unter Vernachlässigung der Absorption folgt für kleine Winkel näherungsweise

$$\alpha_c \approx \sqrt{2\delta} = \lambda \sqrt{\frac{r_e \rho}{\pi}}. \quad (1)$$

($r_e \hat{=}$ Klassischer Elektronenradius, $\rho \hat{=}$ Elektronendichte des Materials)

2.2 Fresnel Formeln

Im allgemeinen muss bei der Reflektion und Transmission elektromagnetischer Wellen die Polarisation des Lichts berücksichtigt werden. Dies geschieht mithilfe der Fresnel Formeln. Für s-polarisiertes Licht ergeben sich

$$r = \frac{n_1 \cos \alpha_1 - n_2 \cos \alpha_2}{n_1 \cos \alpha_1 + n_2 \cos \alpha_2}$$
$$t = \frac{2n_1}{n_1 \cos \alpha_1 + n_2 \cos \alpha_2}.$$

Für diesen Versuch ist eine Unterscheidung zwischen p und s Polarisation aufgrund der ähnlichen Brechungsindizes $n_1 \approx n_2$ nicht nötig. Die Fresnel Fresnelreflektivität ist für Röntgenstrahlung und für $\alpha_i > 3\alpha_c$ näherungsweise

$$R_f = \left(\frac{\alpha_c}{2\alpha_i} \right).$$

2.3 Mehrschichtsysteme

Da in diesem Versuch mit Polystyrolschicht auf einem Siliziumsubstrat gearbeitet wird, wird im folgendem der Umgang mit Mehrschichtsystemen erläutert. Ein beispielhaftes Verhalten der Reflektivität befindet sich in Abbildung 1.

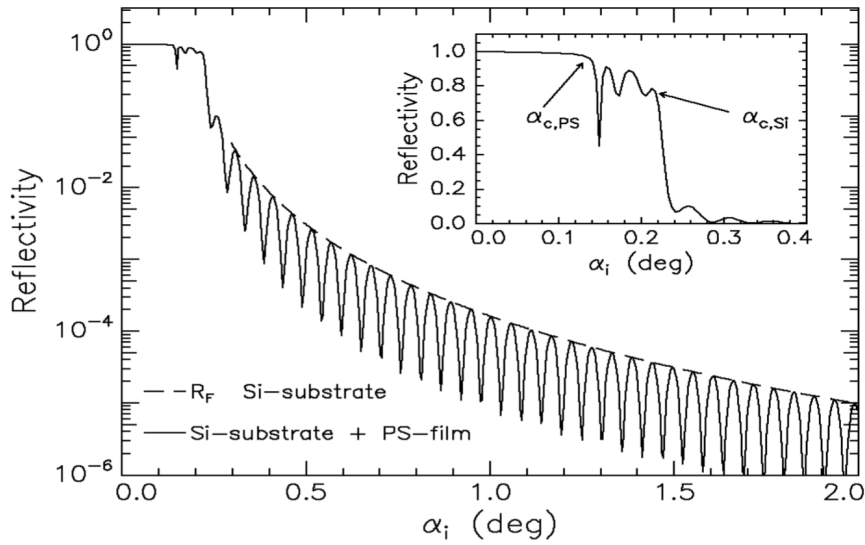


Abb. 1: Beispielhafte Auftragung der Röntgenreflektivität R gegen den Einfallswinkel α_i [1].

In dem dort vergrößerten Bereich sind zwei Totalreflexionen zu erkennen. Bei diesen handelt es sich um die Totalreflexionen von Silizium und des Polystyrolfilm. Für höhere Einfallswinkel folgt der zu erwartende Abfall der Reflektivität. Die dabei beobachtbaren Oszillationen treten aufgrund von Interferenzeffekten an der Oberfläche auf. Mithilfe dieser Oszillationen lassen sich Rückschlüsse auf den Schichtabstand ziehen, da der Gangunterschied für destruktive Interferenzen ein ungerades Vielfaches von $\frac{\lambda}{2}$ sein muss. Damit folgt der Zusammenhang

$$d = \frac{2\pi}{\delta q_z} = \frac{\lambda}{2\delta \alpha_1}.$$

$$(\vec{q} = \vec{k}_2 - \vec{k}_1, q_z = 2k \sin \alpha_1)$$

Handelt es sich nun bei dem zu betrachtendem System um - wie in Abbild 2 dargestellt - ein System mit $N+1$ Schichten, kann die Reflektivität mithilfe des rekursiven Parratt-Algorithmus berechnet werden. Dieser trifft die Annahme, dass es sich bei der untersten Schicht um eine unendlich dicke Schicht handelt, sodass an dieser keine Transmission stattfindet.

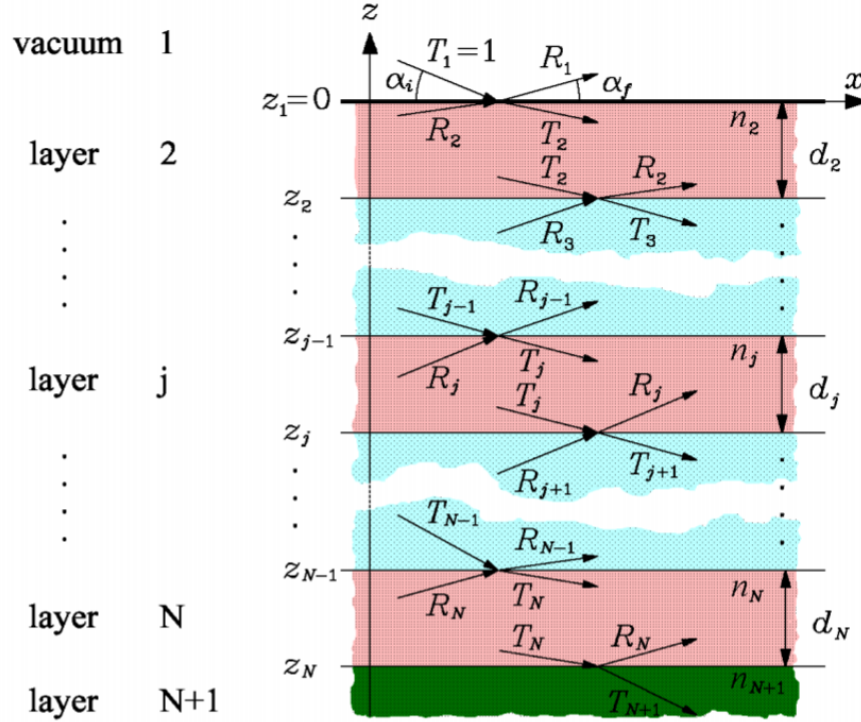


Abb. 2: Beispielhafte Darstellung eines Mehrschichtensystems mit $N+1$ Schichten [1].

2.4 Rauigkeit

Im Kapitel 2.1 wurde angenommen, dass es sich bei den Oberflächen perfekt glatte Oberflächen handelt. Da dies im Experiment jedoch nicht der Fall ist, muss dies bei der Berechnung der Reflektivität berücksichtigt werden. Dafür werden die modifizierten Fresnelkoeffizienten

$$\begin{aligned}\tilde{r}_{j,j+1} &= r_{j,j+1} \exp(-2k_{z,j}k_{z,j+1}\sigma_j^2) \\ \tilde{t}_{j,j+1} &= t_{j,j+1} \exp((k_{z,j}-k_{z,j+1})^2 \cdot \frac{\sigma_j^2}{2})\end{aligned}$$

genutzt.

2.5 Geometriefaktor und Geometriewinkel

Wie in Abbildung 3 zu sehen ist, überstreicht der verwendete Strahl eine größere Fläche, als die Probenoberfläche. Dies führt dazu, dass nur ein Teil der Intensität I reflektiert und somit später detektiert wird. Dies wird durch den Geometriefaktor G berücksichtigt und wird als das Verhältnis der Strahlbreite $D \sin \alpha_i$, die die Probenoberfläche trifft, zur Gesamtstrahlbreite d_0 definiert. Dabei gilt:

$$G = \frac{D \sin \alpha_i}{d_0} \quad \text{mit } \alpha_i < \alpha_g \quad \text{und} \quad (2)$$

$$G = 1 \quad \text{mit } \alpha_i > \alpha_g. \quad (3)$$

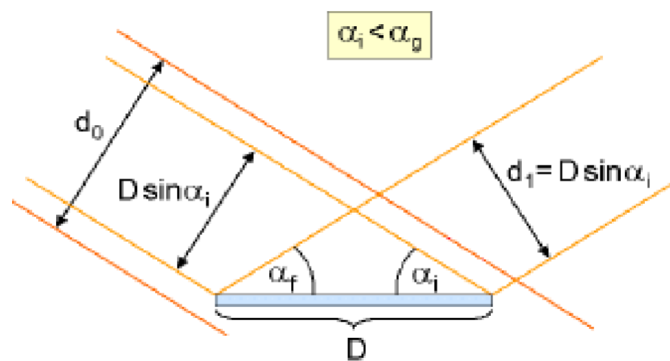


Abb. 3: Veranschaulichung des Geometriewinkels [2].

3 Aufbau und Durchführung des Versuchs

Die Messung wird mit dem in Abbildung 4 zu sehenden D8-Diffraktometer durchgeführt. Bei diesem handelt es sich um eine Röntgenröhre mit einer Kupferanode welche mit einer Spannung von 40 kV und einem Strom von 40 mA betrieben wird.



Abb. 4: Das verwendete D8-Labordiffraktometer [2].

Die aus der Röntgenröhre divergierende Strahlung wird hier mithilfe eines Göbelspiegel gebündelt und monochromatisiert. Der daraus resultierende Strahl besitzt dann eine Wellenlänge von $\lambda = 1,54 \text{ \AA}$. Ein Bild der verwendeten Röntgenröhre befindet sich in Abbildung 5.

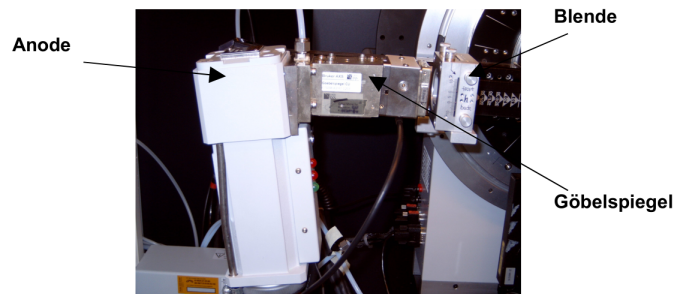


Abb. 5: Röntgenröhre des D8-Labordiffraktometers[2].

3.1 Justage

Detektorscan Um den Detektorscan durchzuführen, wird zunächst die Probe aus dem Strahlengang entfernt. Dabei werden Röntgenröhre und Detektor auf die Position 0° gefahren. Um nun die tatsächliche Nulllage des Detektors zu finden, wird seine Lage um wenige Grad variiert, bis die Intensität des Primärstrahles ein durchläuft Maximum durchläuft, welches für die folgenden Schritte als neue Nullposition des Detektors verwendet wird.

Erster Z-Scan In diesem Schritt wird die Probenjustage angepasst. Dabei wird die z-Position der Probe variiert. Dazu wird diese wieder in den Strahlengang geschoben und die Intensität I gemessen. Ziel dieser Messung ist es, herauszufinden, wann dabei die gemessene Intensität I auf $\frac{1}{2}I_{\text{Max}}$ sinkt. Der z-Wert wird notiert und die Motoren in die entsprechende Position gebracht.

Erster Rockingscan Nachdem in den vorherigen Schritten die zu untersuchende Probe parallel zur Strahlachse ausgerichtet wurde, erfolgt nun der Rockingscan. Bei diesem werden Röntgenröhre und Detektor um die Probe bewegt, wobei der Winkel zwischen Detektor und Probe bei Konstant $2\Theta = 0^\circ$ bleibt. Die Drehung erfolgt dabei im Winkelbereich zwischen -1° und 1° . Aus dem daraus folgendem Intensitätsverlauf wird das Maximum ausgelesen und für die weiteren Schritte verwendet.

Zweiter Z-Scan Die Probe befindet sich aufgrund des Rockingscans nun nicht mehr in der Position, in der die gemessene Intensität I der halben maximalen Intensität $\frac{1}{2}I_{\text{Max}}$ entspricht. Aus diesem Grund muss hier ein erneuter Z-Scan durchgeführt werden.

Zweiter Rockingscan Zur Erhöhung der Präzision wird nun ein zweiter Rockingscan mit $2\Theta = 0,3^\circ$ durchgeführt. Dabei wird für die Drehung ein Winkelbereich von $0,1^\circ$ bis $0,2^\circ$ gewählt.

Dritter Z-Scan Um die Präzision der Justage weiter zu erhöhen, wird anschließend ein dritter z-Scan bei $2\Theta = 0,3^\circ$ durchgeführt. Dabei wird ein Scanbereich zwischen $-0,5\text{ mm}$ und $0,5\text{ mm}$ gewählt. Wie auch bei den vorherigen z-Scans wird hier erneut das Maximum gemessen und die Motoren entsprechend angesteuert.

Dritter Rockingscan Als letzter Schritt wird ein abschließender Rockingscan unter einem Winkel von $2\Theta = 1^\circ$ durchgeführt. Dabei wird ein Scanbereich zwischen den Werten $0,45^\circ$ und $0,55^\circ$ gewählt, da das erwartete Maximum bei $0,5^\circ$ liegt. Nachdem das Maximum gefunden wurde, werden die Motoren entsprechend angesteuert.

3.2 Messung

Nachdem die Justage des D8-Diffraktometer durchgeführt wurde, wird ein sogenannter Reflektivitätsscan durchgeführt. Bei diesem sind der Einfallswinkel α_i und der Winkel zwischen Probe und Detektor α_f gleich. Hierbei wird ein Scanbereich von 0° bis 5° eingestellt und vermessen. Dazu wird eine Schrittweite von $0,05^\circ$ verwendet, wobei als Messzeit pro Datenpunkt 5 s gewählt wird.

Zusätzlich wird ein sogenannter diffuser Scan durchgeführt, der den Anteil der gestreuten Intensität an der Reflektivität bestimmt. Dieser Scan wird mit dem Unterschied durchgeführt, dass die Differenz hier $\Delta a = |\alpha_i - \alpha_f| = 0,1^\circ$ beträgt.

4 Auswertung

4.1 Vorbereitung

Detektor-Scan Zur Bestimmung der maximalen Intensität wird der erste Detektor-Scan in Abbildung 6 dargestellt.

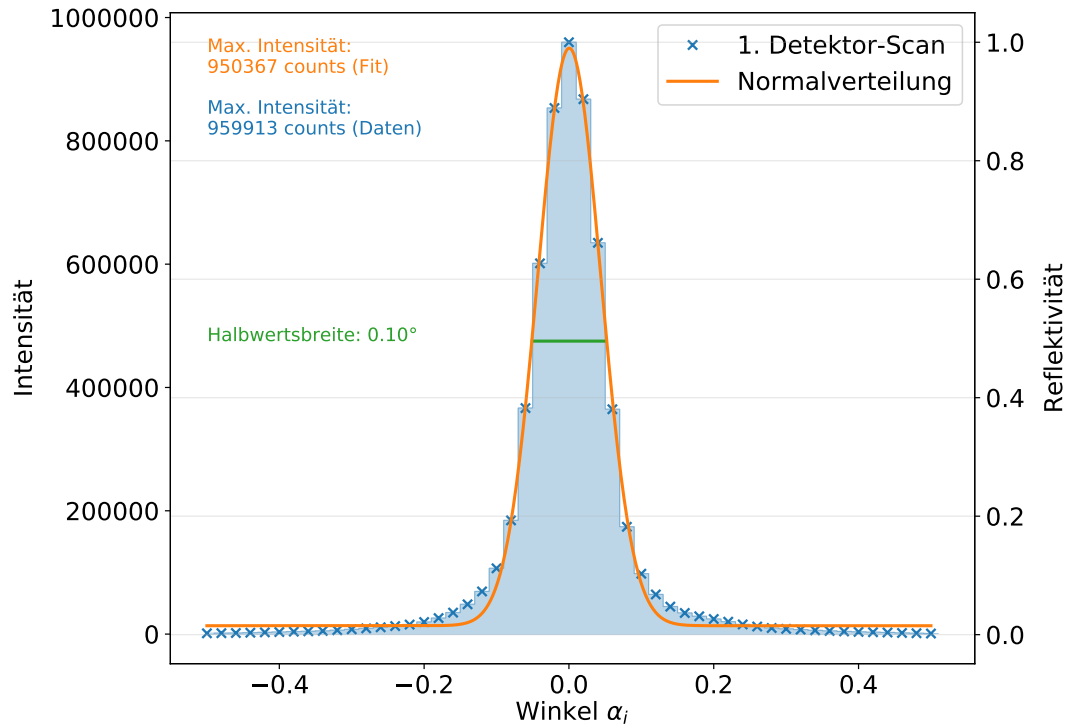


Abb. 6: Detektor-Scan: Die gemessene Intensität aufgetragen gegen den Einfallswinkel α_i .

Das Maximum der aufgenommenen Daten wird zur Normierung der Reflektivitätskala verwendet. An die Daten wird mit *ipython 3.6.8* und *scipy.optimize.curve_fit* eine Gaussverteilung der Form

$$f(\alpha_i) = \frac{a}{\sqrt{2\pi}\sigma^2} \exp\left(-\frac{(\alpha_i - \mu)^2}{2\sigma^2}\right) + b$$

a, b Fit-Parameter, μ Erwartungswert, σ Standardabweichung

gefittet. Die Parameter ergeben sich zu

$$\begin{aligned}
 a &= (102,109\,927\,00 \pm 1,089\,250\,87) \cdot 10^3 \text{ counts} \\
 b &= (13,559\,228\,90 \pm 2,242\,937\,88) \cdot 10^3 \text{ counts} \\
 \mu &= (4,734\,632\,15 \pm 4,710\,145\,25) \cdot 10^{-4} \text{ }^\circ \\
 \sigma &= (4,348\,379\,540\,0 \pm 0,049\,348\,984\,1) \cdot 10^{-2} \text{ }^\circ.
 \end{aligned}$$

Innerhalb aus den Daten wird die Halbwertsbreite (FWHM) zu folgendem Wert berechnet:

$$\text{FWHM} = 10^\circ.$$

z-Scan Der erste z-Scan wird in Abbildung 7 gezeigt.

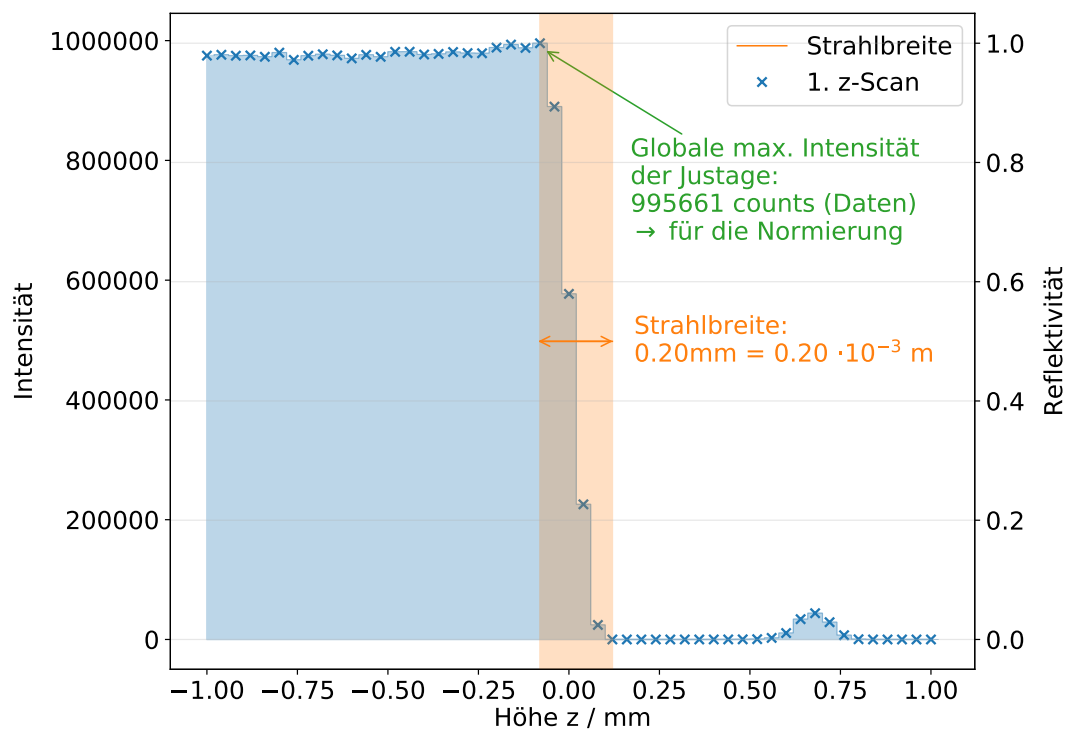


Abb. 7: z-Scan: Hier wird die gemessene Intensität gegen die vertikale Position der Probe aufgetragen.

Aus dem Abstand auf der z-Achse, der zwischen Maximum und Minimum liegt, wird die Strahlbreite bestimmt. Sie wird den Daten als

$$d = 0,2 \cdot 10^{-3} \text{ m}$$

entnommen. Bei der Auswertung dieses Datensatzes fällt auf, dass hier global eine größere Intensität gemessen wird, als im Detektor-Scan, trotz gleicher Messdauer. Somit wird die maximale Intensität

$$I_{max} = 995\,661 \text{ counts}$$

zur Normierung der restlichen Datensätze verwendet.

rocking-Scan Der erste rocking-Scan ist in Abbildung 8 verbildlicht.

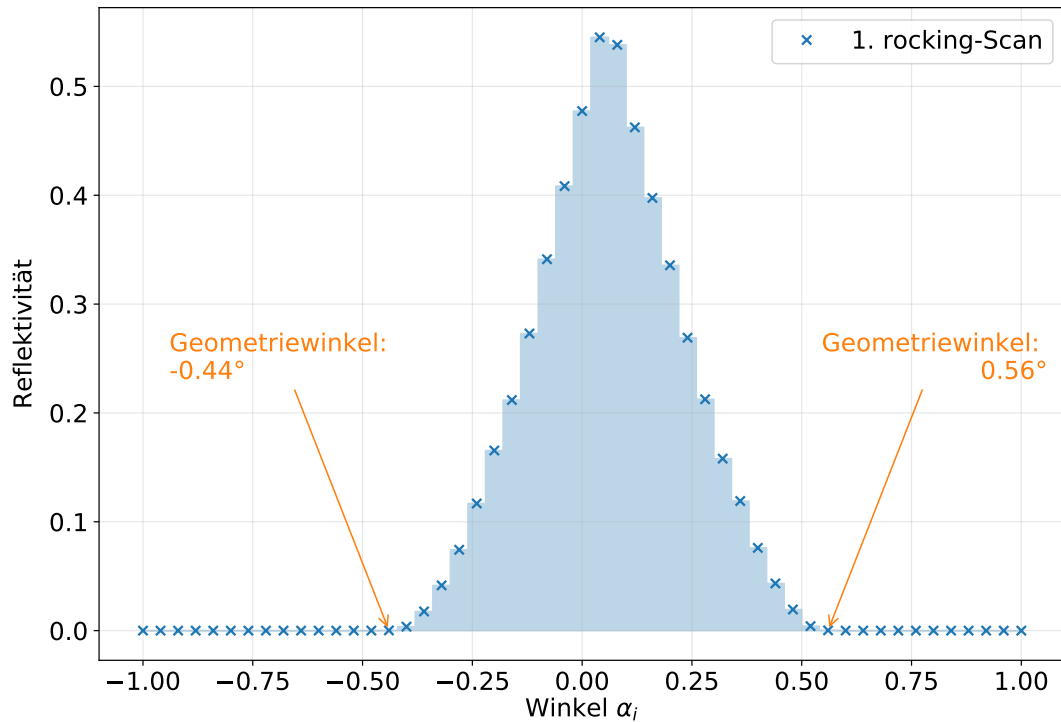


Abb. 8: rocking-Scan: Die Reflektivität der Probe in Abhängigkeit des Einfallswinkels α_i unter konstantem Winkel zwischen Strahlenquelle und Detektor.

Hier werden aus den Daten zwei Werte für den Geometriewinkel abgelesen. Dazu werden die Werte gewählt, bei denen sich die Intensität relevant von 0 unterscheidet. Diese werden anschließend gemittelt:

$$\begin{aligned} a_g &= [0.44^\circ, 0.56^\circ] \\ \overline{a_g} &= 0.50^\circ. \end{aligned}$$

4.2 Auswertung der Vermessung eines mit Polystyrol beschichteten Silicium-Wafers

Zunächst werden die Daten normiert und aufbereitet. Die Messdauer beträgt hier fünf mal so lange, wie die Justage-Messungen. Dieser Faktor wird in die Normierung einbezogen. Die Daten zum Einfallswinkel α_i werden in den Wellenvektorübertrag $q = \frac{4\pi}{\lambda} \sin \frac{\pi}{180} \alpha_i$ überführt. Danach wird der Datensatz der diffusen Messung von dem Datensatz der eigentlichen Messung abgezogen, um Rückstreuereffekte im Schichtsystem aus den Daten zu eliminieren. Anschließend wird der Geometriefaktor berechnet. Hierfür werden die Maße des Wafers ($2\text{ cm} \times 2\text{ cm}$), die in 4.1 berechnete Strahlbreite d und der Geometriewinkel α_g verwendet (Gleichungen (2) und (3)). Der Geometriefaktor ist abhängig vom Einfallswinkel und verhindert die Unterrepräsentation sehr kleiner Winkel, bei denen nicht die gesamte Strahlbreite am Wafer reflektiert wird. Die normierten, korrigierten Daten werden nun durch den Geometriefaktor geteilt und sind hiermit vollständig korrigiert.

Nun wird mithilfe von *scipy.signal.find_peaks* die Region der Kiessing-Oszillationen nach Minima durchsucht. Aus dem Abstand der Minima wird die Schichtdicke des Polystyrol(PS)-Films auf dem Silicium(Si)-Wafer als

$$z_{\text{Minima}} = (882,41 \pm 0,03) \cdot 10^{-10} \text{ m}$$

abgeschätzt.

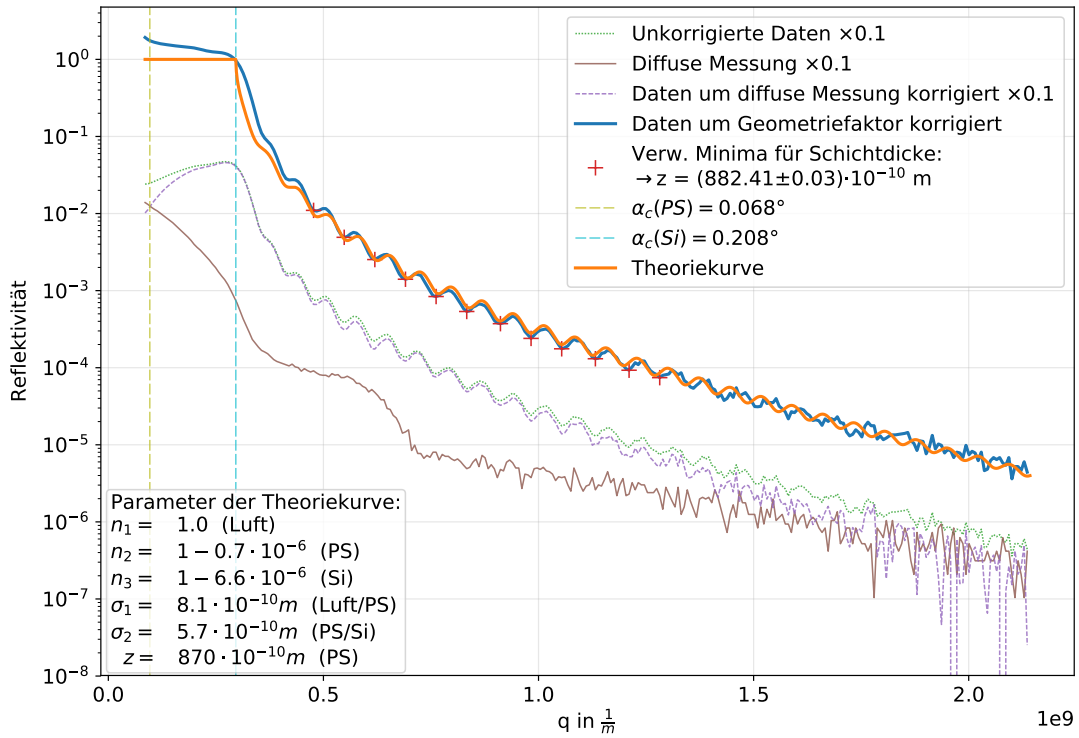


Abb. 9: Vermessung der Reflektivität des PS-Si-Wafers in Abhängigkeit des Wellenvektorübertrags q . Zur Übersichtlichkeit sind die Graphen der Datenaufbereitung nach unten verschoben, sonst verschwinden sie über große Teile der Abbildung unter den anderen Graphen.

Sämtliche Graphen und die verwendeten Minima sind in Abbildung 9 dargestellt. Die Graphen zu den jeweiligen Aufbereitungsschritten sind zur Übersichtlichkeit mit einem dem Faktor 0.1 multipliziert und aufgetragen.

Zur weiteren Bestimmung der interessanten Größen wird eine Theoriekurve nach dem Parratt-Algorithmus konstruiert (Anhang 6). Die Parameter der Theoriekurve sind die Brechungsindizes n_i der drei beteiligten Materialien (Luft, PS, Si), die Rauigkeiten σ_i der beiden Grenzflächen (Luft-PS, PS-Si) und die Schichtdicke z . Manuell werden diese Parameter angepasst, um die Theoriekurve den Daten zu nähern. Die gewählten

Parameter liegen bei:

Luft	$n_1 =$	1,0
PS	$n_2 =$	$1 - 0,7 \cdot 10^{-6}$
Si	$n_3 =$	$1 - 6,6 \cdot 10^{-6}$
Luft-PS	$\sigma_1 =$	$8,1 \cdot 10^{-10} \text{ m}$
PS-Si	$\sigma_2 =$	$5,7 \cdot 10^{-10} \text{ m}$
PS-Schichtdicke	$z =$	$870 \cdot 10^{-10} \text{ m.}$

Aus den Korrekturtermen δ (in $n = 1 - \delta$) lassen sich die kritischen Winkel der Totalreflexion der beiden Materialien berechnen (Gleichung (1)):

$$\text{PS} \quad \alpha_c = \quad 0,068^\circ \quad (4)$$

$$\text{Si} \quad \alpha_c = \quad 0,208^\circ. \quad (5)$$

$$(6)$$

5 Diskussion

Initial lässt sich sagen, dass der Versuch nicht von den Erwartungen abweicht. Die gemessenen Größen liegen in den Größenordnungen der Literaturwerte und der eigenständig reproduzierten Werte. So weicht die abgeschätzte Schichtdicke z_{Minima} nur gering von der manuell angepassten Schichtdicke $z_{\text{Theoriekurve}}$ ab:

Minima	Theoriekurve	Rel. Abw.
$z = (882,41 \pm 0,03) \cdot 10^{-10} \text{ m}$	$z = 870 \cdot 10^{-10} \text{ m}$	1,43 %.

Die Abweichung lässt sich einerseits durch eine nicht optimale Anpassung der Parameter der Theoriekurve erklären, andererseits sind nur begrenzt viele Minima in den anderen Wert der Schichtdicke eingegangen. Eine weitere Messung mit längerer Messdauer pro Winkel könnte die Intensität bei großen Winkeln α_i bzw. Wellenvektorüberträgen q auf repräsentative Zählraten anheben. In der Messung in diesem Versuch werden die Messdaten bei größeren q vom Rauschen überlagert. Zu den Dispersionen δ (in $n = 1 - \delta$) liegen die Literaturwerte [3] bei

	Literatur	aus der Anpassung	Rel. Abw.
PS	$\delta = 3,5 \cdot 10^{-6}$	$\delta = 0,7 \cdot 10^{-6}$	80 %
Si	$\delta = 7,56 \cdot 10^{-6}$	$\delta = 6,6 \cdot 10^{-6}$	12,70 %.

Dabei fällt gerade die starke Abweichung des Korrekturterms des Polystyrols ins Auge. Eine mögliche Erklärung hierfür ist eine möglicherweise ungenaue Wahl des Parameters. Weiterhin sind aber die kritischen Winkel in der passenden Größenordnung [3]:

	Literatur	aus der Anpassung	Rel. Abw.
PS	$\alpha_c = 0,15^\circ$	$\alpha_c = 0,068^\circ$	54,67 %
Si	$\alpha_c = 0,22^\circ$	$\alpha_c = 0,208^\circ$	5,45 %.

Insgesamt lässt sich der Versuch trotz einiger Startschwierigkeiten jedoch als Erfolg betrachten.

Literatur

- [1] Lehrstuhl Experimentelle Physik I. Roentgenreflektometrie Versuch. 2007. URL: http://e1.physik.tu-dortmund.de/cms/Medienpool/Downloads/Roentgenreflektometrie_Versuch.pdf.
- [2] TU Dortmund. Versuchsanleitung V44. Röntgenreflektometrie.
- [3] M. Tolan. X-Ray Scattering from Soft Matter Thin Films. Berlin/Heidelberg/New York: Springer-Verlag, 1999. Kap. Reflectivity of X-Rays from Surfaces - Multiple Interfaces.

6 Anhang

```
0 if name == "main":
1     import numpy as np
2     import sympy as sp
3     import matplotlib.pyplot as plt
4     import matplotlib.patches as mpatches
5     from matplotlib import transforms
6     from matplotlib import rc
7     from scipy.optimize import curve_fit
8     from pylab import figure, axes, pie, title, show
9     from numpy import NaN, Inf, arange, isscalar, asarray, array
10    import sys
11    import scipy.signal as sig
12    import uncertainties
13    import uncertainties.unumpy as unp
14    from uncertainties import ufloat
15    from uncertainties.unumpy import (nominal_values as noms, std_devs as
stds)
16    from math import exp, log, sin, atan
17    import scipy.constants as const
18    from scipy.stats import norm
19    import cmath as cm # was sich halt so ansammelt und was man hier und da
gebrauchen könnte...
20
21    ### Variablen für die ganze Datei
22    l = 1.54e-10 # Wellenlänge
23    ai = np.arange(0.06, 1.505, 0.0005)*np.pi/180 # x-array für Theoriekurve
24    qz = 4*np.pi/l*np.sin(ai) # q-array für die Theoriekurve
25    k = 2*np.pi/l # k-Vektor
26
27    ### Parameter für die manuelle Anpassung des Parratt-Algorithmus
28    ### Angegebene Instruktionen beschäftigen sich mit der Vergrößerung eines
Parameters
29    #####
30    #Brechungsindices
31    delta1 = 0.7e-6 #PS schiebt runter und vergrößert die Amplitude, macht
die Kurve steiler
32    delta2 = 6.6e-6 #Si schiebt Kurve hoch, verkleinert die Amplitude
33    #####
34    n1 = 1 #Luft
35    n2 = 1 - delta1 # PS
36    n3 = 1 - delta2 # Si
37    #####
38    #Rauigkeit
39    sigma1 = 8.1e-10 #PS verkleinert die Amplitude in den hinteren
Oszillationen
40    sigma2 = 5.7e-10 #Si drückt Ende der Kurve runter und verkleinert die
Oszillationen
41    #####
42    #Schichtdicke
43    z = 870e-10 # verkleinert die Wellenlänge der Oszillationen
44    #####
```



```

46 x, y = np.loadtxt('1416_messung.txt', unpack=True, delimiter=', ') #
    Einlesen der Messdaten
48 y = y/(5*995661) # Normierung
    plotallgraphs(x, y) # hier passiert die Magie
50
    print('— done —')
52
def plotallgraphs(x, counts): # Magie
54 ### allg. Plot-Einstellungen
    plt.rcParams['figure.figsize'] = (10, 7)
56 plt.rcParams['font.size'] = 13
    plt.rcParams['lines.linewidth'] = 2
58 ### Verwendete alternative Linestyles
    denslydotted = (0, (1, 1))
60    denslydashed = (0, (3, 1.25))
    longdashes = (0, (10, 3))
62
    fig, ax = plt.subplots()
64 ### ab hier werden die Daten aufbereitet
    q = von_winkel_zu_q(x) # Umrechnung von Winkel zu Wellenvektorübertrag q
66
    a,b = lade_diffuse_messung() # Einlesen der diffusen Messung
68    counts_korrigiert_diffus = korrektur_diffuse_messung(counts, b) #
    Korrigieren der Messung um die diffuse Messung
    counts_korrigiert_geometriefaktor = korrektur_geometriefaktor() #
    Korrigieren der korrigierten Messung um den Geometriefaktor
70    counts_final = counts_korrigiert_geometriefaktor # Umbenennung da mein
    Variablenname furchtbar war...
    minimum_x, minimum_y, peak_array = finde_minima(counts_final, q) # Findet
    Minima im logarithmischen Graph und gibt verschiedene Arrays mit xy-
    Position und zugehörigem Array-Index aus
72    schichtdicke = berechne_schichtdicke(q, peak_array) # Selbsterklärend
74
    u = 40 # Ende des Plateaus
76    beg = 11 # Anfang der sinnvollen Messdaten
    end = 300 # Ende der sinnvollen Messdaten
78
    unkorrr = 1e-01*counts[beg:end] # Verschiebung der Datensätze um 0.1 nach
    unten.
80    diffusedaten = 1e-01*b[beg:end] # Verschiebung der Datensätze um 0.1 nach
    unten. counts_korrigiert_diffus wird in der produzierenden Funktion mit
    0.1 multipliziert, da es sonst Fehler mit den Listen etc gibt... Yolo
82 ### Hier wird geplottet
    ax.plot(q[beg:end], unkorrr,
84           linestyle=denslydotted,
            linewidth = 1,
86           color='C2',
            label='Unkorrigierte Daten ' + r'$\times 0.1$',
88           alpha=0.8)

```

```

90     ax.plot(a[beg:end], diffusedaten,
           linestyle='-',
           linewidth = 1,
92     color='C5',
           label='Diffuse Messung ' + r'$\times 0.1$',
94     alpha=0.8)
96     ax.plot(a[beg:end], counts_korrigiert_diffus[beg:end],
           linestyle=denslydashed,
           linewidth = 1,
98     color='C4',
           label='Daten um diffuse Messung korrigiert ' + r'$\times 0.1$',
100    alpha=0.8)

102    ax.plot(q[beg:end], counts_final[beg:end],
           linestyle='-',
104    color='C0',
           label='Daten um Geometriefaktor korrigiert')
106    ax.plot(minimum_x, minimum_y,
           markerstyle='+',
108    color='C3',
           markersize=10,
           markeredgewidth=1,
110    label='Verw. Minima für Schichtdicke: \n' + r'$\rightarrow$ ' +
112    r'$z = (%.2f$'%(schichtdicke.n * 1e9) + '$\pm$ ' +
           r'$%.2f$'%(schichtdicke.s * 1e9) + r'$\cdot 10^{-10}$ m')
114

116    alphakritPS, qkritPS = kritischerwinkel(delta1) # berechnet den
           kritischen Winkel und plottet ihn anschließend als vertikale Linie
           ax.axvline(qkritPS, ymin=0, ymax=1,
           linewidth=0.7,
118    linestyle=longdashes,
           color='C8',
120    label=r'$\alpha_c$ (PS)=%.3f °$' %alphakritPS)

122    alphakritSi, qkritSi = kritischerwinkel(delta2)
           ax.axvline(qkritSi, ymin=0, ymax=1,
           linewidth=0.7,
124    linestyle=longdashes,
           color='C9',
126    label=r'$\alpha_c$ (Si)=%.3f °$' %alphakritSi)
128 #####
           ### Hier ist der wichtige Aufruf:
130    par = parratt(z)
           plt.plot(qz, par, linestyle='-', color='C1', label='Theoriekurve')
132 #####
           ### hier werden die Ergebnisse auch in die Box im Plot geplottet
134    textstr = '\n'.join((
           'Parameter der Theoriekurve:',
136
           r'$n_1$ = $'+'\t'+
138    r'$ %.1f$' % (n1) + ' (Luft)',

140    r'$n_2$ = $'+'\t'+

```

```

142     r'$ 1 - %.1f \cdot 10^{-6}$' % (delta1*1e06) + ' (PS)',
144     r'$n_3 = $'+'\t'+
144     r'$ 1 - %.1f \cdot 10^{-6}$' % (delta2*1e06) + ' (Si)',
146     r'$\sigma_1 = $'+'\t'+
148     r'$ %.1f \cdot 10^{-10}$ m $' % (sigma1*1e10) + ' (Luft/PS)',
148
150     r'$\sigma_2 = $'+'\t'+
150     r'$ %.1f \cdot 10^{-10}$ m $' % (sigma2*1e10) + ' (PS/Si)',
152
152     ' ' + r'$z = $'+'\t' +
154     r'$ %s \cdot 10^{-10}$ m$' % (int(z*1e10)) + ' (PS)',))
154
156 props = dict(facecolor='white',
156             edgecolor=(0.808, 0.808, 0.808),
158             alpha=0.8,
158             boxstyle='round', pad=0.2')
160 ax.text(0.01, 0.01,
160         textstr,
162         transform=ax.transAxes,
162         va='bottom',
164         ha='left',
164         bbox=props)
166
166 ### Plot-Basics
168 plt.yscale('log')
168 plt.grid(alpha=0.3)
170 ax.legend(fancybox=True, ncol=1)
170 ax.set_xlabel('q in ' + r'$\frac{1}{m}$',
172             labelpad=2)
172 ax.set_ylabel('Reflektivität',
174             labelpad=8)
174 fig.tight_layout()
176 plt.savefig('done_plot_messung.pdf')
176
176 #####
178 def parratt(z):
180     kz1=k*np.sqrt((n1**2-np.cos(ai)**2))
180     kz2=k*np.sqrt((n2**2-np.cos(ai)**2))
182     kz3=k*np.sqrt((n3**2-np.cos(ai)**2))
182
184     r12=(kz1-kz2)/(kz1+kz2)*np.exp(-2*kz1*kz2*sigma1**2)
184     r23=(kz2-kz3)/(kz2+kz3)*np.exp(-2*kz2*kz3*sigma2**2)
186     x2=np.exp(0-(kz2*z)*2j)*r23
186     x1=(r12+x2)/(1+r12*x2)
188     par = np.abs(x1)**2
188     # Es gibt eine ganze Strecke komplexe Daten vor Beginn der Kiessing-
188     # Oszillationen, daher werden die ersten Werte des Arrays nachträglich auf
188     # 1 gesetzt. Sieht interessant aus, wenn man einen Betrag um den Inhalt der
188     # Wurzel in den modifizierten Fresnel-Koeffizienten setzt. Ich hab den
188     # Runtime Error ertragen...
188     for i in range(len(par)):

```

```

190         if (i <= 296):                # 296 ist manuell angepasst
191             par[i] = 1
192         else:
193             pass
194     return par
195     #####
196
197 def kritischerwinkel(delta):
198     alphakrit = np.sqrt(2*delta)*180/np.pi
199     qkrit = von_winkel_zu_q(alphakrit)
200     return(alphakrit, qkrit)
201
202 def berechne_schichtdicke(q, peak_array):
203     abstandderpeaks = []
204     for i in peak_array:
205         if (i<179):
206             tmp = q[i+1]-q[i]
207             abstandderpeaks.append(tmp)
208         else:
209             pass
210     mittelwert = np.mean(abstandderpeaks)
211     fehler = np.std(abstandderpeaks)
212     schichtdicke = 2*np.pi/ufloat(mittelwert, fehler)
213     return schichtdicke
214
215 def finde_minima(m, q):
216     ### Logarithmiere die Daten zur Basis 10
217     logm = []
218     for i in range(len(m)):
219         if (m[i] != 0):
220             tmp = np.log10(m[i])
221             logm.append(tmp)
222         else:
223             logm.append(0)
224     ### Negatives VZ um mit find_peaks arbeiten zu können
225     neglogm = []
226     for i in range(len(logm)):
227         tmp = -logm[i]
228         neglogm.append(tmp)
229     ### Peaks finden
230     peakfinder = sig.find_peaks(neglogm, prominence = 0.07)
231     ### Peaks in den Einträgen:
232     peak_array = [ 66, 76, 86, 96, 106, 116, 127, 137, 147, 158, 169, 179,
233 # 190, 200, 211, 221, 232, 241, # 245, 252, 263, 266, 275, 278, 283,
234 # 285, 292, 294, 299, 304, 306, 312, 314, 316, 320, 326, 328, 331,
235 # 335, 338, 340, 344, 346, 349, 352, 355, 359, 363, 365, 368, 370,
236 # 372, 375, 377, 380, 382, 385, 390, 393, 396, 398, 402, 404, 408,
237 # 413, 415, 421, 423, 426, 429, 433, 435, 438, 442, 445, 448, 453,
238 # 455, 459, 462, 465, 469, 472, 478, 483, 489, 494, 7
239     ] ### Manuell eingekürzt, da Minima bei großen q sehr unscharf
240
241     peak_x = []
242     peak_y = []

```

```

244     for i in peak_array:
245         peak_x.append(q[i])
246         peak_y.append(m[i])
247     return peak_x, peak_y, peak_array

248 def korrektur_geometriefaktor():
249     alpha = 0.5
250     m = []
251     for i in range(len(x)):
252         if (x[i] <= alpha):
253             tmp = y[i] / ((np.sin(x[i]))/(np.sin(alpha)))
254             m.append(tmp)
255         else:
256             tmp = y[i]
257             m.append(tmp)
258     return m

259 def korrektur_diffuse_messung(counts, b):
260     korrigierte_counts = []
261     for i in range(len(counts)):
262         tmp = 1e-01*(counts[i]-b[i])
263         korrigierte_counts.append(tmp)
264     return korrigierte_counts

265 def lade_diffuse_messung():
266     a, b = np.loadtxt('1501_diffus.txt', unpack=True, delimiter=', ')
267     ### Normierung
268     b = b/959913
269     qdiffus = von_winkel_zu_q(a)
270     return qdiffus, b

271 def von_winkel_zu_q(x):
272     q = 4 * np.pi / l * np.sin(x*np.pi/180)
273     return q

274 ### bin kein Informatik-Talent, aber ich bemühe mich um Übersichtlichkeit –
275 ob das geklappt hat, werden wir sehen.

```

content/messung.py