Praktikum 5

Markus Roos, Simon Stingelin

09.04.2024

Aufgaben:

1 Lineare Ausgleichsrechnung mit Hilfe der QR-Zerlegung

1.1 Lernziele

- Sie lernen die Matrix-Faktorisierung mit Hilfe der QR-Zerlegung.
- Sie verstehen die Anwendung der QR-Zerlegung zur Lösung eines linearen Ausgleichsproblems
- Sie verstehen die Berechnung der QR-Zerlegung mit Hilfe der Householder-Transformation
- Sie können die QR-Zerlegung anwenden.

1.2 Theorie

In einem ersten Schritt betrachten wir die Berechnung der QR-Zerlegung mit Hilfe der Householder-Transformation. Die Theorie ist im Skript Kapitel 2.2.3 zu finden und wurde in der Vorlesung präsentiert.

1.3 Auftrag 1

- Erarbeiten Sie die Matrix-Faktorisierung Schritt für Schritt. Sie können sich dazu am Jupyter-Notebook (Link unten) orientieren. Beantworten Sie folgende Fragen:
  1. Wie lautet Q, R so, dass $A = Q \cdot R$ gilt?
  2. Welche Dimension hat Q, R?
  3. Welche Dimension hat das reduzierte Problem?
  4. Warum reichen die reduzierten Matrizen Q, R?
- Implementieren Sie abschliessend eine Methode, welche die QR-Zerlegung einer beliebigen Matrix $A \in \mathbb{R}^{m \times n}$ berechnet.

  Kontrollieren Sie Ihr Resultat mit der QR-Zerlegung von numpy:

```
Q,R = np.linalg.qr(A)
```

Jupyter-Notebook:

Householder-Transformation schrittweise

[ ]:
```
import numpy as np
from numpy.linalg import norm
```

Die allgemeine Schreibweise der Householder-Transformation für einen beliebigen Vektor w ist gegeben durch

$$H(w) = \mathrm{id} - 2\,\frac{w \cdot w^T}{\langle w, w \rangle} \tag{1}$$

wobei $w \cdot w^T$ das Kroneckerprodukt

$$w \cdot w^T = (w_i\, w_j)_{i,j=1\ldots m} \in \mathbb{R}^{m \times m}$$

sei.

[ ]:
```
# Selber implementieren
def HouseholderTransformation(w):
    return <<snipp>>
```

Gesucht ist der geeignete Normalenvektor so, dass der gespiegelte Spaltenvektor auf die e_1 Achse zu liegen kommt. Sei mit y der Spaltenvektor bezeichnet, so kann man zeigen (siehe Skript), dass der Vektor

$$w = y \pm \|y\|_2 e_1 \tag{2}$$

die gewünschte Eigenschaft hat. Um **Auslöschung** in der Berechnung von w zu vermeiden, wählt man

$$w = y + \mathrm{sign}(y_1)\|y\|_2 e_1 \tag{3}$$

mit

$$\mathrm{sign}(s) = \begin{cases} 1 & \text{für} s \geq 0 \\ -1 & \text{sonst.} \end{cases} \tag{4}$$

[ ]:
```
def mysign(x): # numpy sign liefert 0 für 0
    if x >= 0:
        return 1
    else:
        return -1
```

Funktion für den n-dimensionalen Einheitsvektor

[ ]:
```
def e(n):
    return np.array([1]+[0 for k in range(n-1)])
```

Mit Hilfe der Householder-Transformation soll nun die Matrix A in eine orthogonale Matrix Q und reguläre obere Dreiecksmatrix R zerlegt werden. Im Beispiel wählen wir eine zufällige Matrix $A \in \mathbb{R}^{10\times 5}$.

[ ]:
```
A = np.array([[-1,  7, -8, -9,  6],
       [-6, -8,  0,  3,  8],
       [-4, -2,  8,  0, -2],
       [-1, -9,  4, -8,  2],
       [-3, -5, -5,  7, -4],
       [-7, -4,  7, -1,  5],
       [-9, -7,  6, -5, -8],
       [-4, -3, -5,  3, -6],
       [ 5,  7,  5, -4, -5],
       [ 4, -6, -8, -2, -5]],dtype=float)
m,n = A.shape
```

1. Spalte

[ ]:
```
k = 0
```

Die Hyperebene ist definiert durch

[ ]:
```
w = <<snipp>>
```

Für die Householder-Transformationsmatrix angewand auf A erhalten wir

[ ]:
```
Q1 = HouseholderTransformation(w)
A1 = Q1@A
```

In der ersten Spalte der Zwischenmatrix A_1 stehen nun abgesehen vom ersten Eintrag Nullen:

[ ]:
```
print(np.round(A1,4))
```

2. Spalte

```
[ ]: k = 1
```

Die Hyperebene ist definiert durch

```
[ ]: w = <<snipp>>
```

wobei nun das letzte Resultat A_1 benutzt wird. Die Householder-Transformationsmatrix wird nun nur auf die Submatrix von A_1 angewand und in der Submatrix von A_1 wiederum gespeichert:

```
[ ]: Q2 = HouseholderTransformation(w)
A1[k:,k:] = Q2@A1[k:,k:]
```

Die Dimension der zweiten Householder-Transformationsmatrix Q_2 ist

```
[ ]: Q2.shape
```

In dem ersten beiden Spalte der Zwischenmatrix A_1 steht:

```
[ ]: print(np.round(A1,4))
```

3. - 5. Spalte

Wir automatisieren nun den Prozess und überschreiben die Submatrizen der Matrix A_1 sukzessive:

```
[ ]: for k in range(2,n):
    print('Spalte '+str(k+1))
    w = <<snipp>>
    Qk = HouseholderTransformation(w)
    A1[k:,k:] = <<snipp>>
    print(np.round(A1,4))
```

Q, R berechnen

- Berechnen sie abschliessend Q, R so, dass $Q \cdot R = A$ gilt.
- Vergleichen Sie Ihr Resultat mit der Funktion von NumPy.

```
[ ]: Q,R = np.linalg.qr(A)
```

```
[ ]: Q.shape
```

```
[ ]: R.shape
```

Frage: Warum reicht diese reduzierte Q und R Matrix?

2 Industrielle Anwendung aus der Gas Analytik

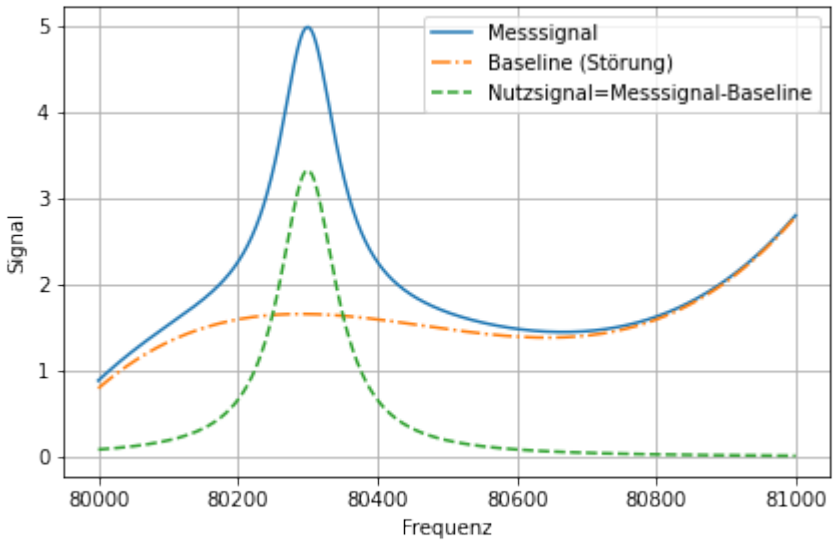
2.1 Lernziele

- Sie können die QR-Zerlegung auf ein industrielles Beispiel anwenden.
- Sie verstehen die Ursache der schlechten Konditionierung.
- Sie kennen Methoden zur Verbesserung der Kondition.

2.2 Theorie und Auftrag 2

Anwendung aus der Messtechnik: Baseline Fit

In spektrometrischen Anwendungen der Messtechnik hat man oft das Problem, dass die sogenannte Baseline (Untergrund) der Messung unbekannt ist. In der Darstellung unten sehen Sie eine typische Situation:



Für die Interpretation des physikalischen Vorgangs ist man nur am Peak (grün) interessiert. Die Rohsignale entsprechen jedoch der blauen Kurve. Das Baselinesignal (orange) ist die Folge von Hintergrundprozessen oder Eigenschaften des Messgerätes und sollte diskriminiert werden. Die folgende Anleitung zeigt Ihnen, wie das mit Hilfe der Ausgleichsrechnung gelingt.

Beschreibung des Untergrundes (Baseline)

Wir gehen davon aus, dass sich der Untergrund durch ein unbekanntes **Polynom 3. Grades** beschreiben lässt, welches **additiv** zum eigentlichen **Peak der Resonanz** dazu kommt. Damit können wir die Messdaten durch eine *lineare* Kombination von Funktionsansätzen fitten.

Lorentz-Shape einer Resonaz

Die Systemreaktion wird mit Hilfe der Lorentz-Shape $l(x')$ beschrieben:

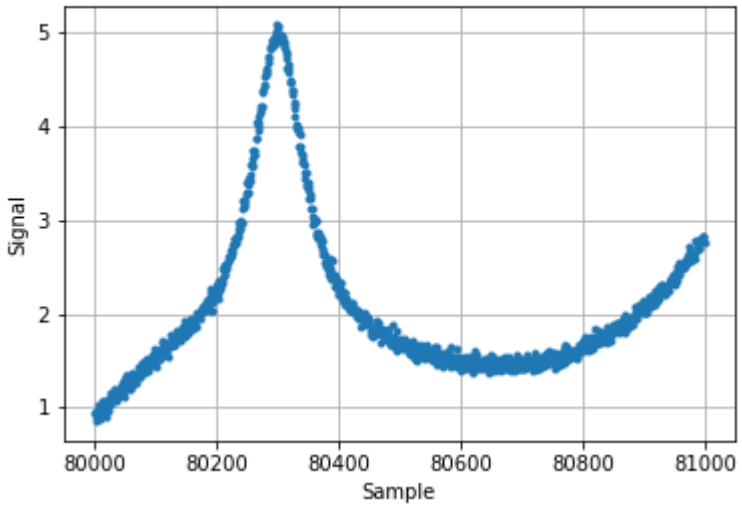
$$l(x') = \frac{1}{1+(x')^2}, \quad x' = \frac{x - x_0}{s_0/2}.$$

wobei x_0 die Resonanzfrequenz und s_0 die Breite des Peaks bezeichnen. In vielen Anwendungen ist die Resonanzfreqeunz, also x_0 zum vornherein bekannt, aber nicht der Wert von s_0. Wenn die Breite s_0 aus den Messdaten bestimmt werden muss, wird die Problemstellung *nichtlinear*.

Deshalb gehen wir hier von einer **bekannten** Breite $s_0 = 100$ aus, die Resonanz befinde sich bei $x_0 = 80.3 \cdot 10^3$. Unbekannt ist jedoch die Amplitude des Peaks, d.h. die multiplikative Konstante vor $l(x')$.

Rauschenanteil

Im allgemeinen wird die Aufgabe erschwert durch Rauschen im Messsignal, d.h. realistische Daten sehen folgendermassen aus:



Sie finden diese Daten im Downloadbereich als „data.txt“. Eingelesen werden diese Werte via:

```
import numpy as np
np.loadtxt('data.txt')
```

resp. unter Matlab:

```
load('data.txt')
```

Auftrag 2

Aufgabe 1: Separation von Lorentz-Peak und Untergrund

- Bestimmen Sie die Systemmatrix A für das lineare Ausgleichsproblem zur Separation des Untergrundes.
- Berechnen Sie die Lösung des Ausgleichsproblems mit folgenden Varianten:
  - Mit der Normalengleichung via $A^T A$

    Bemerkung: Benutzen Sie hier **alle** gelernten Ansätze (LR, Cholesky, QR)! Sie werden sehen, dass die Matrix so schlecht konditioniert ist, dass nur mit QR eine halbwegs brauchbare Lösung berechnet werden kann!
  - Mit der QR-Zerlegung der Matrix A direkt

  Bestimmen Sie die Kondition der jeweiligen Matrix und geben Sie einen Kommentar ab, was diese Werte über die erwarte numerische Genauigkeit aussagen.
- Stellen Sie den Lorentz-Peak als Funktion mit der gefitteten Amplitude graphisch dar, zusammen mit den verrauschten Daten von welchen Sie den Untergrund *abziehen*.

Aufgabe 2: Verbesserung der Numerik (optional)

- Verbessern Sie die Kondition der beteiligten Matrizen, indem Sie die x-Achse auf das Intervall $[0, 1]$ skalieren.
- Wiederholen Sie die Berechnungen mit den skalierten, verschobenen x-Werten und dokumentieren Sie die Unterschiede zum unskalierten Fall.

3 Abgabe

Bitte geben Sie Ihre Lösungen bis spätestens vor dem nächsten Praktikum 6 ab. Kurzer Bericht mit den Ergebnissen und python Code, Jupyter-Notebook.

Downloads:

- PDF-Dokumentation:
  - `Anleitung Praktikum 5`
  - `data.txt`
- Jupyter-Notebooks:
  - `Householder Transformation`
  - `Industrielle Anwendung aus der Gas Analytik`