Institut für Physikalische Chemie

Methodenkurs „Anwendungen von Mathematica und Matlab in der Physikalischen Chemie“ im Sommersemester 2016

Prof. Dr. Stefan Weber, Dr. Till Biskup

— Aufgabenblatt 7 zum Teil „Matlab“ vom 26.07.2016 —

Aufgabe 7—1 (Polynomfit)

Die nachfolgende Tabelle zeigt Ihnen diskrete fehlerbehaftete Datenpunkte, die Sie mit einem Polynom n-ter Ordnung beschreiben sollen.

x	y
0	0.298
1	0.770
2	0.384
3	1.508
4	2.588
5	6.014
6	12.062
7	21.648

Geben Sie zunächst die Werte als Vektoren in MATLAB ein und stellen Sie sie als unverbundene Punkte dar. Nutzen Sie anschließend die Möglichkeiten, die MATLAB Ihnen bietet, um Polynome unterschiedlichen Grades an diese Datenpunkte anzupassen.

Stellen Sie die Anpassungen für die Polynome erster bis siebter Ordnung gemeinsam mit den fehlerbehafteten Datenpunkten grafisch dar. Entscheiden Sie selbst, ob es sinnvoll ist, alle diese Polynome in einer Abbildung darzustellen.

Aufgabe 7—2 (Lineare Regression mit festem y-Achsenabschnitt)

Wie Sie wissen, gibt es Situationen, in denen aufgrund des zugrundeliegenden Modells ein Geradenausgleich mit festem y-Achsenabschnitt erforderlich ist. Nachfolgend sind Ihnen wieder fehlerbehaftete diskrete Datenpunkte gegeben, die Sie mit einem Geradenausgleich mit der Randbedingung $x(0) = 1$ beschreiben sollen.

x	y
1	1.7937
2	2.5565
3	3.5307
4	4.4131
5	5.0879

Geben Sie zunächst die Werte als Vektoren in MATLAB ein und stellen Sie sie wieder als unverbundene Punkte dar. Nutzen Sie anschließend die Möglichkeiten, die MATLAB Ihnen bietet, einen Geradenausgleich mit festem y-Achsenabschnitt durchzuführen.

Stellen Sie die Anpassung anschließend gemeinsam mit den fehlerbehafteten diskreten Datenpunkten grafisch dar.

Aufgabe 7—3 (Lineare Regression komplexerer Funktionen)

Letztlich lassen sich alle Funktionen f, die jeweils nur linear von ihren Koeffizienten a_i abhängen,

$$f(a_0, \ldots, a_n) = a_0 + a_1 f_1(x_k) + a_2 f_2(x_k) + \cdots + a_n f_n(x_k) = y(k)$$

als lineares Gleichungssystem darstellen:

$$F\,a = y \qquad \text{mit} \qquad F = \begin{bmatrix} 1 & f_1(x_1) & \cdots & f_n(x_1) \\ 1 & f_1(x_2) & \cdots & f_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & f_1(x_M) & \cdots & f_n(x_M) \end{bmatrix}$$

Wie Sie sehen, handelt es sich bei Polynomen um spezielle Fälle dieser Funktionen, die nur linear von ihren Koeffizienten a_i abhängen: für Polynome gilt $f_n(x_k) = x_k^n$.

Eine besondere Stärke von MATLAB liegt nun gerade im Bereich der linearen Algebra, und entsprechend einfach und elegant können lineare Gleichungssysteme gelöst werden.

Viele Funktionen lassen sich so umschreiben, dass sie nur linear von ihren Parametern abhängen. Als Beispiel sei die Funktion

$$y = A \cdot \sin(x + \phi)$$

genannt. Sie können diese Funktion zweier Parameter sehr einfach in eine Funktion umschreiben, die nur linear von ihren Parametern abhängt:

$$y = a_1 \sin(x) + a_2 \cos(x)$$

Diese Funktion können Sie bequem als lineares Gleichungssystem umformulieren und das wiederum entsprechend elegant mit MATLAB lösen.

Definieren Sie sich in MATLAB die Funktion $f(x) = \frac{\pi}{2} \sin(x + \frac{2\pi}{3})$ und evaluieren Sie sie für $x = [0 \ldots 20]$ mit einer Schrittweite von 0.2. Stellen Sie anschließend das lineare Gleichungssystem auf und lösen Sie es mit MATLAB. Nutzen Sie dabei die Ihnen oben gegebene Möglichkeit, die ursprüngliche Funktion als Summe zweier trigonometrischer Funktionen darzustellen, die jeweils nur linear von ihren Koeffizienten abhängen.

Aus den erhaltenen Koeffizienten können Sie problemlos die Amplitude und die Phasenverschiebung der angepassten Funktion sowie die Funktion selbst errechnen.

Stellen Sie anschließend die ursprünglich definierte Funktion (mit offenen Kreisen als Symbole), die angepasste Funktion sowie in grau die beiden für die Anpassung und die Erzeugung des linearen Gleichungssystems verwendeten Funktionen $\sin(x)$ und $\cos(x)$ gemeinsam im oben angegebenen Intervall dar. Die Abbildung sollte in etwa so aussehen wie in Abb. 1 dargestellt.

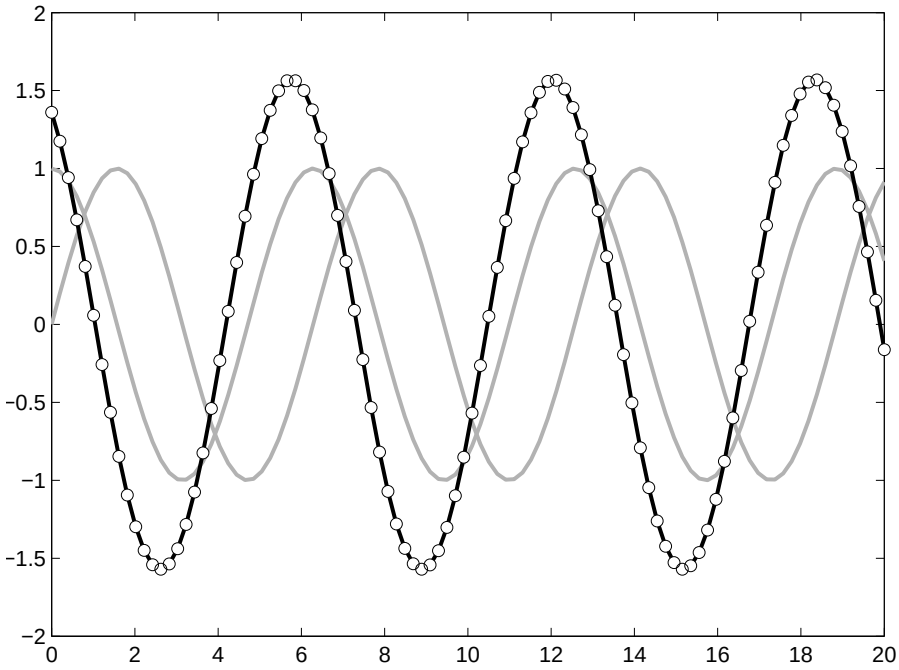

Abbildung 1: Lineare Regression komplexerer Funktionen. Die offenen Kreise repräsentieren die originale Funktion, die grauen Linien die beiden zugrundeliegenden Funktionen, Sinus und Cosinus, und die schwarze Linie das Resultat der Kurvenanpassung mittels linearer Regression.

Zusatzaufgabe: Wiederholen Sie das Procedere, aber addieren Sie diesmal auf die Ausgangsfunktion ein wenig Rauschen. Machen Sie dafür von der MATLAB-Funktion `rand` Gebrauch und skalieren sie das Rauschniveau gegebenenfalls auf ein sinnvolles Niveau: Sie sollten einerseits eindeutig eine Abweichung der Datenpunkte von der Ideallinie erkennen, andererseits aber nach wie vor auch die Form der ursprünglichen Funktion ohne Rauschen.

Aufgabe 7—4 (Nichtlineare Kurvenanpassung beliebiger Funktionen)

Laden Sie sich von der Webseite des Kurses[1] die Daten herunter und stellen Sie sie zunächst (als unverbundene Symbole) dar. Was Sie sehen, erinnert entfernt an eine stark verrauschte Lorentz-Kurve.

Lorentz-Kurven haben in der Spektroskopie eine große Bedeutung. Spektrale Linien, die lediglich lebenszeitverbreitert sind, erscheinen als Lorentz-Linien, inhomogen verbreiterte Linien hingegen weisen ein Gauß-Profil auf. Über die Form der spektralen Linie können Sie also grundlegende Aussagen über die Herkunft der Verbreiterung treffen. Aufgrund ihres charakteristischen, unterschiedlichen Aussehens lassen sich Lorentz- und Gauß-Linienformen in der Regel auch bei verrauschten Daten recht gut unterscheiden.

Nutzen Sie die MATLAB-Funktion `fminsearch`, um eine Lorentz-Kurve der Form

$$f(x) = \frac{a_1}{(x - a_2)^2 + a_3}$$

an die eingelesenen Daten anzupassen. Beachten Sie, dass Sie der Funktion `fminsearch` bereits die zu minimierende Funktion übergeben müssen, in unserem Fall also nicht die oben definierte Lorentz-Funktion, sondern bereits die Summe der Quadrate der Abweichungen.

Hinweis: Es empfiehlt sich, eine (anonyme) Funktion in MATLAB zu definieren, die der obigen Lorentz-Funktion entspricht, und eine zweite Anpassungsfunktion zu definieren, die die Summe der Quadrate

[1] `http://www.till-biskup.de/de/lehre/mathematica-matlab/ss2016/material/09/`

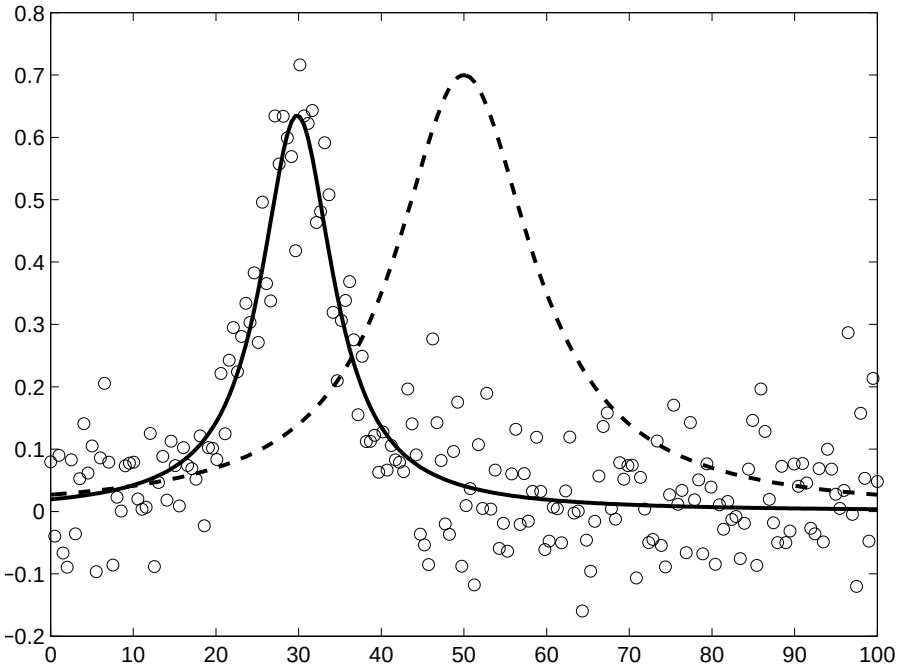

Abbildung 2: Nichtlineare Regression verrauschter Datenpunkte. Die offenen Kreise repräsentieren die verrauschten Daten, die gestrichelte schwarze Linie die Lorentz-Funktion mit den Startwerten, die durchgezogene schwarze Linie das Resultat der Kurvenanpassung mittels nichtlinearer Regression.

der Abweichungen der definierten Lorentz-Funktion von den Daten darstellt und die dann der Routine `fminsearch` übergeben werden kann.

Wählen Sie für die Koeffizienten a_i, $i = 1, 2, 3$ der anzupassenden Lorentz-Funktion die folgenden Werte als Startparameter:

$$a_1 = 70 \qquad a_2 = 50 \qquad a_3 = 100$$

Stellen sie anschließend das Ergebnis der Anpassung gemeinsam mit den eingelesenen Daten und der Lorentz-Funktion mit den oben gegebenen Startwerten für die Koeffizienten a_i in einer gemeinsamen Abbildung dar. Die Abbildung sollte in etwa so aussehen wie in Abb. 2 dargestellt.