Ü 5 Fünfte Übungseinheit

Inhalt der fünften Übungseinheit:

- Nichtlineare Datenmodelle
- Kurze Wiederholung: lineare und nichtlineare Datenmodelle
- Kenntnisnachweis-Musteraufgaben
- Bonus-Material: MATLAB-Werkzeuge zum Anpassen von Funktionen an Daten

Ü 5.1 Überbestimmte nichtlineare Systeme, Gauß-Newton-Verfahren

Das Gauß-Newton-Verfahren findet Näherungslösungen (im Sinn der kleinsten Fehlerquadrate) für überbestimmte nichtlineare Systeme. Die Grundidee ist, ähnlich wie bei der Lösung nichtlinearer Gleichungssysteme mit dem Newton-Verfahren, mit der *Jacobimatrix* des nichtlinearen Systems iterativ Korrekturterme zu berechnen. Die Aufgaben 41 und 42 zeigen ausführlich den Rechenweg.

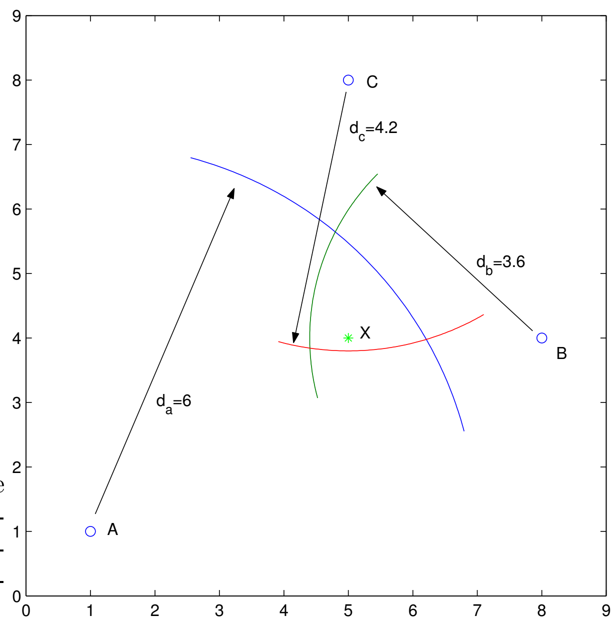
Aufgabe 41: Standortbestimmung durch Trilateration

Die Abstände von drei festen Punkten A, B, C in der xy-Ebene zu einem unbekannten Punkt X sind (etwas ungenau) bekannt. Gesucht ist eine möglichst gute Positionsbestimmung.

Punkt	x	y	Entfernung
	1	1	6
	8	4	3,6
	5	8	4,2

Die entsprechenden Gleichungen lauten:

$$\begin{aligned}\sqrt{(x_1-1)^2+(x_2-1)^2} &= 6\\ \sqrt{(x_1-8)^2+(x_2-4)^2} &= 3.6\\ \sqrt{(x_1-5)^2+(x_2-8)^2} &= 4.2\end{aligned}$$



Den drei Gleichungen entsprechen drei Kreise im $\mathbb{R}^2$. Sie haben keinen gemeinsamen Schnittpunkt. Dementsprechend gibt es keine exakte Lösung des überbestimmten Gleichungssystems.

Schreiben Sie ein MATLAB-Programm, das die kleinste-Quadrate-Anpassung für den Standort findet.

Bemerkung: Diese Aufgabe ist die 2-dimensionale Vereinfachung einer Positionsbestimmung im Raum. Die GPS-Technik beruht auf diesen geometrischen Grundlagen. Ein GPS-Empfänger misst die Abstände (genauer: Signallaufzeiten) zu mehreren Satelliten. Die Signale sind stark verrauscht, erst Ausgleichsrechnung und zusätzliche Datenfilterung gewährleisten eine auf einige Meter genaue Position.

Kurzfassung: Aufgabenstellung und Lösungsweg

Gegeben: überbestimmtes nichtlineares System

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}\ , \quad \mathbf{x} \in \mathbb{R}^n\ , \quad \mathbf{f}(\mathbf{x}) \in \mathbb{R}^m, \quad m > n$$

Iterative Lösung des linearisierten Systems: Ausgehend von Startvektor $\mathbf{x}^{(0)}$ bestimmt man eine Korrektur $\Delta\mathbf{x}$.

Die Rechenvorschrift des Newton-Verfahrens für $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ ergibt ein überbestimmtes lineares System mit der Jacobimatrix D_f

$$D_f \cdot \Delta\mathbf{x} = -\mathbf{f}(\mathbf{x})$$

Verbesserte Lösung $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \Delta\mathbf{x}$.

Konkret für die vorliegende Aufgabe:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \sqrt{(x_1-1)^2+(x_2-1)^2} - 6 \\ \sqrt{(x_1-8)^2+(x_2-4)^2} - 3.6 \\ \sqrt{(x_1-5)^2+(x_2-8)^2} - 4.2 \end{bmatrix}, \qquad D_f = \begin{bmatrix} \frac{x_1-1}{\sqrt{(x_1-1)^2+(x_2-1)^2}} & \frac{x_2-1}{\sqrt{(x_1-1)^2+(x_2-1)^2}} \\ \frac{x_1-8}{\sqrt{(x_1-8)^2+(x_2-4)^2}} & \frac{x_2-4}{\sqrt{(x_1-8)^2+(x_2-4)^2}} \\ \frac{x_1-5}{\sqrt{(x_1-5)^2+(x_2-8)^2}} & \frac{x_2-8}{\sqrt{(x_1-5)^2+(x_2-8)^2}} \end{bmatrix}$$

Mit Startvektor $\mathbf{x} = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$ erhält man

$$\mathbf{f}\left(\begin{bmatrix} 5 \\ 4 \end{bmatrix}\right) = \begin{bmatrix} -1 \\ -3/5 \\ -1/5 \end{bmatrix},\ D_f = \begin{bmatrix} \frac{4}{5} & \frac{3}{5} \\ -1 & 0 \\ 0 & -1 \end{bmatrix}\ , \quad \text{lin. Syst.} \begin{bmatrix} \frac{4}{5} & \frac{3}{5} \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3/5 \\ 1/5 \end{bmatrix}$$

Ergibt $\Delta x_1 = 1/25, \Delta x_2 = 7/25 \quad \longrightarrow$ verbesserte Position $[5.04; 4.28]$.

Sie wiederholen diese Rechenschritte und iterieren, bis sich die Position im Rahmen einer vernünftig gewählten Fehlerschranke nicht mehr ändert. Vergleichen Sie dazu den Übungs-Abschnitt Ü 3.5: der Rechengang und vor allem auch die Implementierung in MATLAB sind nahezu gleich. Unterschied: die Matrix des Gleichungssystems ist nun nicht mehr quadratisch und auch nicht konstant; sie hängt von den jeweils aktuellen Näherungswerten für $\mathbf{x}$ ab.

Natürlich sagt dieses Kochrezept nichts zur Theorie überbestimmter linearer Systeme oder zu den Eigenschaften der so berechneten Ausgleichslösung. Die Idee, ein nichtlineares System mittels Jacobi-Matrix linearisiert anzunähern, wird Ihnen aber in der Praxis ständig begegnen.

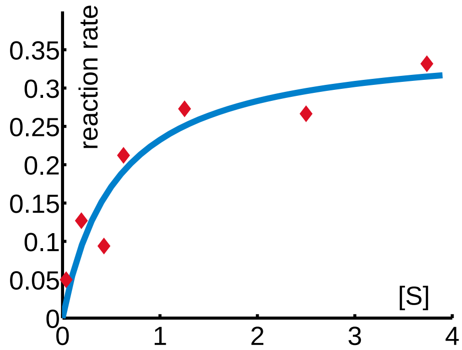
Aufgabe 42: Gauß-Newton-Verfahren in Wikipedia

Die Angabe zum folgenden Beispiel stammt aus der englischen bzw. französische Wikipedia (Stichworte *Gauss-Newton algorithm* bzw. *Algorithme de Gauss-Newton*). Lösen Sie die Aufgabe mit folgender Anleitung in MATLAB.[26]

[26] Wer dieses Beispiel in der englischen oder französischen Wikipedia-Seite durchliest und bei der Abbildung der Modellkurve auf "more details" klickt, findet fertigen MATLAB-Code zur Lösung des Beispiels und Zeichnen der Kurve. Aber das Verstehen von fremdem Code ist auch nicht einfach.

Biologie-Experimente zur Beziehung zwischen Substanz-Konzentration x und Reaktionsrate y in einer durch Enzyme vermittelten Reaktion haben die Daten in folgender Tabelle ergeben:

x	0.038	0.194	0.425	0.626	1.253	2.500	3.740
y	0.05	0.127	0.094	0.2122	0.2729	0.2665	0.3317



Gesucht ist eine Kurve (Modellfunktion) der Form

$$y = a\frac{x}{b+x}$$

die im Sinn der kleinsten Quadrate die Daten am besten approximiert. Die Parameter a und b sind zu bestimmen.
Einsetzen der Daten ergibt sieben nichtlineare Gleichungen in den beiden Unbekannten a, b.

$$\begin{aligned} a\frac{0.038}{b+0.038} - 0.05 &= 0 \\ a\frac{0.194}{b+0.194} - 0.127 &= 0 \\ &\vdots \\ a\frac{3.740}{b+3.740} - 0.3317 &= 0 \end{aligned}$$

Das Gleichungssystem in Vektor-Schreibweise:

$$\mathbf{f}(\mathbf{x}) = 0 \text{ mit } \mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^7$$

Die Jacobi-Matrix D_f dieses Systems ist eine 7×2-Matrix. Zeile i enthält die partiellen Ableitungen der i-ten Gleichung nach den Unbekannten a und b.

$$(D_f)_{i1} = \frac{x_i}{b+x_i}, \quad (D_f)_{i2} = -\frac{ax_i}{(b+x_i)^2}$$

Der weitere Rechenweg verläuft völlig analog zur Anleitung im Übungs-Abschnitt Ü 3.5 und der Musterlösung auf Seite 27 im Skriptum: Wählen Sie als Startwert `[a;b]=[0.9;0.2]`, werten Sie $\mathbf{f}$ und D_f aus. Die kleinste-Quadrate-Näherung aus dem überbestimmten Gleichungssystem $D_f \mathbf{\Delta x} = -\mathbf{f}$ liefert den Korrekturvektor $\mathbf{\Delta x}$.

Beachten Sie: bei einer quadratischen Jacobi-Matrix liefert der MATLAB-Befehl `Df(x0)\f(x0)` die Lösung des Gleichungssystems; hat die Jacobi-Matrix mehr Zeilen als Spalten (überbestimmtes System), liefert derselbe Befehl *nicht* die Lösung (es gibt keine exakte Lösung!), sondern eine Anpassung mit geringstmöglichem Restfehler (die „am wenigsten falsche Antwort“)!

Bonus-Frage: Unser Newton-Musterprogramm findet die verbesserte Näherung mit dem MATLAB-Befehl

```
x = x0 -  Df(x0)\f(x0);    % Newton-Schritt
```

Das Wikipedia-Beispiel nennt die Jacobi-Matrix J_f und verwendet für denselben Schritt (in MATLAB-Code geschrieben) den Befehl

```
x = x0 - (J'*J)\(J'*f(x0));
```

Warum einerseits `Df(x0)\f(x0)` und andererseits `(J'*J)\(J'*f(x0))`? Was wird da jeweils gerechnet? Vor- und Nachteile der beiden Varianten im Vergleich?

Aufgabe 43: Kalorimeterversuch

Der Datensatz `Kalorimeter.dat` (klick!) (auch auf der Übungsseite zum Runterladen) enthält Wertepaare (Zeit t in Minuten, Temperatur T in Celsius) zu einem Kalorimeterversuch, gemessen ab der Einbringung eines Versuchsobjektes. Die Temperatur sinkt zuerst rasch, steigt dann aber wieder und gleicht sich langsam der Umgebungstemperatur an.

Der Verlauf der Temperatur T als Funktion der Zeit t soll durch ein Modell der Form

$$T(t) = T_0 + C_1 \exp(-\lambda_1 t) + C_2 \exp(-\lambda_2 t)$$

beschrieben werden. Bestimmen Sie die Parameter dieses nichtlinearen Modells ausgehend von den Startwerten

$$T_0 = 6 \qquad C_1 = -1 \qquad \lambda_1 = 1/10 \qquad C_2 = 20 \qquad \lambda_2 = 2$$

(drei Iterationen des Newton-Verfahrens reichen aus.) Zeichnen Sie Messpunkte und Modellfunktion.

Ü 5.2 Kurze Wiederholung: lineare und nichtlineare Datenmodelle

Die Aufgaben zur Anpassung von Funktionen an Daten beschränkten sich in der vorigen Einheit auf *lineare Modelle*. Dabei bedeutet „linear“: die gesuchte Anpassung f ist eine *Linearkombination* von n Basisfunktionen $\phi_1, \phi_2, \dots, \phi_n$ in der Form

$$f = a_1\phi_1 + a_2\phi_2 + \cdots + a_n\phi_n \quad .$$

Wichtig: Die Basisfunktionen $\phi_1, \dots, \phi_n$ können beliebige, auch nichtlineare Funktionen sein – es sind die gesuchten Koeffizienten $a_1, \dots, a_n$, die nur linear im Ansatz auftreten!

> Datenanpassung mit linearen Modellen führt auf ein überbestimmtes Gleichungssystem für die gesuchten Koeffizienten $a_1, \dots, a_n$. In den Spalten der Matrix stehen die Werte der Basisfunktionen $\phi_1, \dots, \phi_n$ an den gegebenen Datenpunkten.

Verwirrender Weise wird für die Anpassung einer Ausgleichs-Geraden auch der Begriff „lineare Regression“ verwendet. Unter „polynomiale Regression“ versteht man die Anpassung eines Polynoms an gegebene Daten, wobei es sich aber um ein lineares Datenmodell handelt.

> Unterscheiden Sie:
>
> - Typ der Anpassungsfunktionen: Gerade ("lineare Regression"), Polynom ("polynomiale Regression"), Winkel- oder Exponentialfunktionen...
>
> - Verknüpfung der Ansatzfunktionen: linear, nichtlinear. Dazu gehören die Begriffe lineares/nichtlineares Datenmodell.

Vergleichen Sie: Lineare Modelle

Aufgabe 38: Ansatz $z(x,y) = a_1 + a_2 x + a_3 x^2 + a_4 y$.
Hier sind die Basisfunktionen $\phi_1(x,y) = 1, \phi_2(x,y) = x, \phi_3(x,y) = x^2, \phi_4(x,y) = y$.

Aufgabe 39: Ansatz $t(d,T_0) = a_1 + a_2 d + a_3 T_0 + a_4 d T_0$.
Basisfunktionen sind $\phi_1(d,T_0) = 1, \phi_2(d,T_0) = d, \phi_3(d,T_0) = T_0, \phi_4(d,T_0) = d \cdot T_0$.

Aufgabe 40: Ansatz $y = a_1 + a_2 \cos\left(x\dfrac{\pi}{6}\right) + a_3 \sin\left(x\dfrac{\pi}{6}\right)$
Basisfunktionen sind $\phi_1(x) = 1, \phi_2(x) = \cos\left(x\frac{\pi}{6}\right), \phi_3(x) = \sin\left(x\frac{\pi}{6}\right)$.

Nichtlineare Modelle

Aufgabe 42: Ansatz $y = a\frac{x}{b+x}$. Hier sind die gesuchten Koeffizienten a und b nichtlinear verknüpft!

Aufgabe 43: Ansatz $T(t) = T_0 + C_1 \exp(-\lambda_1 t) + C_2 \exp(-\lambda_2 t)$.
Wären λ_1 und λ_2 bekannt, dann wäre es ein linearen Modell mit Parametern T_0, C_1 und C_2. Weil auch die beiden λ_i gesucht sind, wird die Aufgabe nichtlinear.

Aufgabe 41: Ansatz für Distanz d zwischen Punkt P und X: $d = \sqrt{(x_1 - p_1)^2 + (x_2 - p_2)^2}$; mehrere Datensätze für Punkt-Koordinaten $P = (p_1, p_2)$ und zugehörige Distanzen d sind gegeben. Gesucht: Koordinaten $X = (x_1, x_2)$.

Datenanpassung mit nichtlinearen Modellen führt auf ein überbestimmtes nichtlineares Gleichungssystem. Das Gauss-Newton-Verfahren löst dieses System iterativ; anders als beim Standard-Newton-Verfahren ist die Jacobi-Matrix hier nicht quadratisch; jeder Iterationsschritt löst ein überbestimmtes lineares Gleichungssystem.

Kapitel Ü 5.1 in diesen Unterlagen zeigt einen Standard-Lösungsweg zur nichtlinearen Anpassung. Der folgende Abschnitt Ü 5.4 zeigt weitere Möglichkeiten und stellt MATLAB-Werkzeuge vor. Vielleicht finden Sie diese Werkzeuge einfacher und intuitiver zu verwenden als den Lösungsweg aus Kapitel Ü 5.1.

Ü 5.3 Kenntnisnachweis

Für den ersten Kenntnisnachweis werden zwei Aufgaben zu folgenden Themenkreisen gestellt:

- Lineare Gleichungssysteme: Lösbarkeit, Lösungsmenge
- Nichtlineare Gleichungssysteme: Newton-Raphson-Verfahren, `fsolve` oder einfache Fixpunkt-Iteration.
- Zeichnen von Kurven, graphische Lösung von Gleichungen.
- Darstellen von Funktionen $z = z(x,y)$ als Isolinien oder Flächen, graphisches Finden von Minima/Maxima.
- Lineare Datenmodelle: Anpassen von Funktionen an Daten

Hier folgen drei Muster-Angaben.

Muster-Angabe A

Die Funktion

$$f(x,y) = \left[x^2 + \sin y\right]^2 + [e^x + y]^2$$

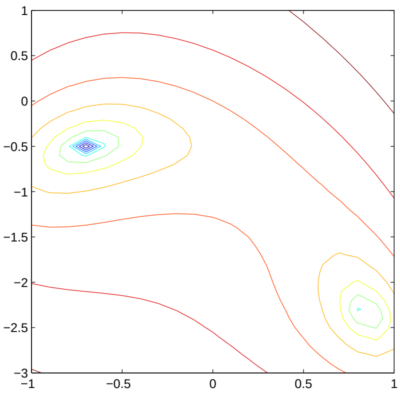
hat Minimalwert 0 für genau jene Wertepaare $(x;y)$, die Lösung des folgenden nichtlinearen Gleichungssystems sind:

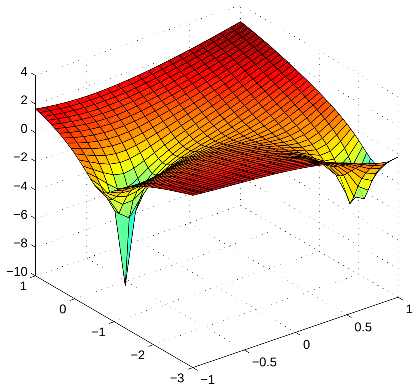
$$\begin{aligned} x^2 + \sin y &= 0 \\ e^x + y &= 0 \end{aligned}$$

Aufgabe 1

8 Punkte

Zeichnen Sie für $\log f$, den Logarithmus der gegebenen Funktion, im Bereich $-1 < x < 1$ und $-3 < y < 1$ eine `contour`- und eine `surf`-Grafik in der Art der Abbildung. (In der logarithmischen Darstellung erkennt man die Lage des Minimums deutlicher als in direkter Darstellung von f.) Verwenden Sie ≈ 20 Gitterpunkte in x- und ≈ 40 Gitterpunkte in y-Richtung.





Aufgabe 2

12 Punkte

Lösen Sie das nichtlineare Gleichungssystem mit dem Newton-Verfahren. Finden Sie beide Lösungen im Bereich $-1 < x < 1$ und $-3 < y < 1$. Passende Startwerte können Sie aus der grafischen Darstellung entnehmen. Abbruchkriterium: Aufeinanderfolgende Iterationen unterscheiden sich, gemessen in der 1-Norm, weniger als 10^{-8}.

Muster-Angabe B

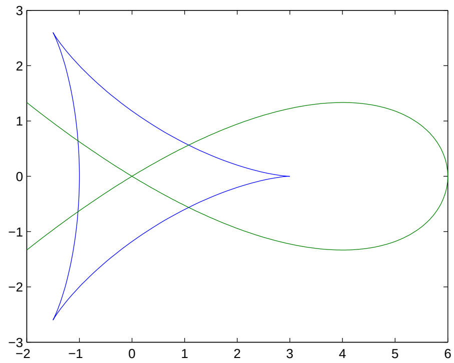
Aufgabe 1

8Punkte

Zwei Kurven in Parameterdarstellung sind gegeben, eine *dreispitzige Hypozykloide* und und *Tschirnhausens Kubische*.

Kurve 1	Kurve 2
$x = 2\cos(s) + \cos(2s)$	$x = 2(3 - t^2)$
$y = 2\sin(s) - \sin(2s)$	$y = 2t(1 - \frac{1}{3}t^2)$

Stellen Sie die beiden Kurven für $s \in [0, \quad 2\pi]$ und $t \in [-2, \quad 2]$ in der xy-Ebene dar und bestimmen Sie graphisch die Lage der beiden Schnittpunkte in der oberen Halbebene.



Aufgabe 2

12 Punkte

Die Schnittpunkte der beiden Kurven von Aufgabe 1 sind auch als Lösung des folgenden Gleichungssystems bestimmt.

$$(x^2 + y^2)^2 + 18(x^2 + y^2) - 27 = 8(x^3 - 3xy^2)$$
$$x^3 = 6\left(x^2 - 3y^2\right)$$

Finden Sie die Lösungen des nichtlinearen Gleichungssystems in der oberen Halbebene mit der MATLAB-Funktion `fsolve`. Passende Startwerte können Sie aus der grafischen Darstellung entnehmen.

Machen Sie die Probe: Setzen Sie die von `fsolve` berechneten Werte in die Funktion ein. Idealer Weise sollte das Ergebnis der Nullvektor sein. Geben Sie die 2-Norm des Restvektors an.

Muster-Angabe C

Aufgabe 1

12 Punkte

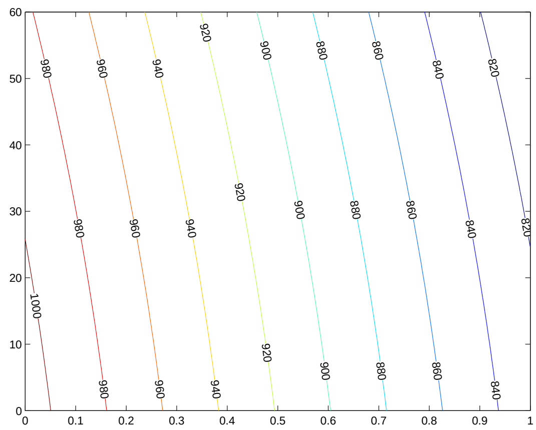
Für die Dichte ρ eines Alkohol-Wasser-Gemisches in Abhängigkeit von Temperatur T (in Celsius) und Alkoholgehalt c (Massenanteil) liegen folgende Tabellenwerte vor:

T	20	40	60	80	20	40	60	80
c	0.1	0.1	0.1	0.1	0.2	0.2	0.2	0.2
rho	981.8	974.8	965.2	954.6	968.6	958.5	945.8	931.3

Finden Sie für diese Daten eine Anpassung der Form

$$\rho(T, c) = a_1 + a_2 T + a_3 T^2 + a_4 c$$

und zeichnen Sie ein Diagramm in der Art der nebenstehenden Abbildung, (x-Achse: Konzentration $0 \le c \le 1$, y-Achse: Temperatur $0 \le T \le 60$,)



Antwort Koeffizienten a_1, a_2, a_3, a_4 :

Aufgabe 2

8 Punkte

Lineare Gleichungssysteme: Lösbarkeit, Lösungsmenge

Gegeben ist ein lineares Gleichungssystem $Ax = b$ mit Matrix A und rechter Seite b.

$$A = \begin{bmatrix} 1 & -3 & 5 & 1 & -1 \\ 3 & -8 & 17 & 5 & -1 \\ 2 & -4 & 15 & 9 & 5 \\ 7 & -21 & 37 & 14 & 3 \\ 9 & -29 & 44 & 18 & 13 \end{bmatrix}, \quad b = \begin{bmatrix} 15 \\ 59 \\ 70 \\ 129 \\ 142 \end{bmatrix}.$$

Verwenden Sie entsprechende MATLAB-Befehle und geben Sie im Skript als Kommentarzeilen die Anworten auf folgende Fragen

- Ist das System eindeutig lösbar? Falls ja, geben Sie den Lösungsvektor an.
- Ist das System nicht lösbar? Falls ja, geben Sie die kleinste-Quadrate-Näherungslösung (die „am wenigsten falsche Lösung“) an.
- Ist das System mehrdeutig lösbar? Geben Sie einen Lösungsvektor mit möglichst kleiner 2-Norm an.

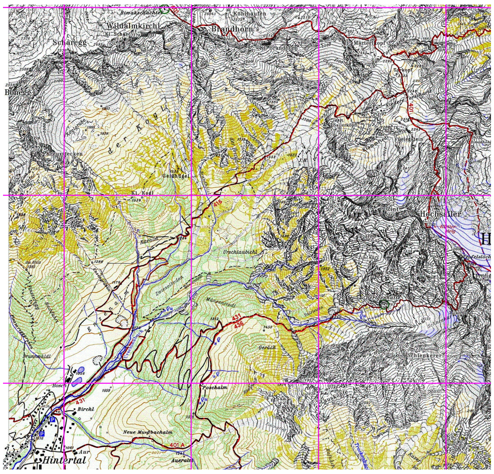
Aufgabe 44: Lineares Datenmodelle anpassen: Georeferenzierung

Das ist noch eine Aufgabe zum Thema „Lineare Datenmodelle“ zur Vorbereitung auf den ersten Kenntnisnachweis. Eine Musterlösung dazu steht zur Verfügung.

Laden Sie dazu hier (klick!) oder von der Übungs-Homepage die Daten und ein Musterprogramm herunter. Die MATLAB-Befehle

```
imdata = imread('HohTs.jpg');
image(imdata)
```

lesen eine Bilddatei (Ausschnitt aus der Alpenvereinskarte Hochkönig, Hagengebirge 1:25000) ein und stellen sie im *figure*-Fenster dar.



In diese Bilddatei sollen GPS-Daten (geogafische Länge und Breite) eingezeichnet werden. Dazu brauchen Sie eine Zuordnung von Pixel-Koordinaten im Bild zu geographischen Koordinaten.

Die Zuordnung

$$\text{Bild-Koordinaten} \longleftrightarrow \text{Welt-Koordinaten}$$

heißt *Bildregistrierung*. Sie verwendet *Kontrollpunkte*: Pixel-Punkte im Bild mit bekannten Welt-Koordinaten, hier geogr. Länge λ und Breite ϕ. Daten für 9 Kontrollpunkte sind im Musterprogramm enthalten.

Finden Sie eine Anpassung der Form

$$x = a_0 + a_1\lambda + a_2\phi$$
$$y = b_0 + b_2\lambda + b_2\phi$$

Pixel-Koord.		Welt-Koord.	
x	y	λ	ϕ
369	2299	12.983	47.417
1143	2299	13.000	47.417
1917	2299	13.017	47.417
⋮	⋮	⋮	⋮

Wie groß ist die maximale Abweichung der Bild-Koordinaten x, y zwischen Modell und Originaldaten? (Idealer Weise sollte die Abweichung nicht mehr als 1–2 Pixel betragen. Die Welt-Koordinaten wurden aber hier um eine Dezimalstelle zu viel gerundet.)

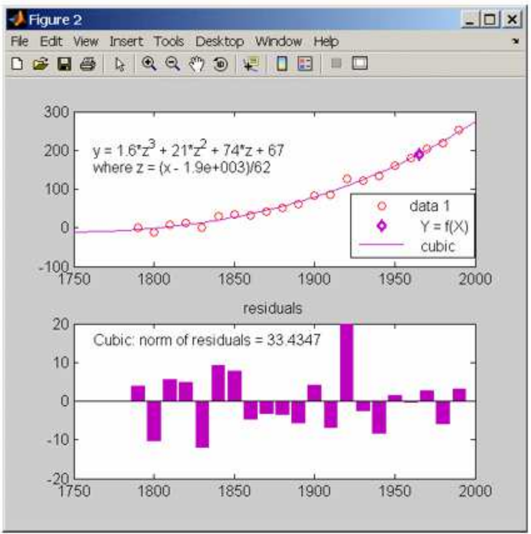
Der MATLAB-Befehl `load trackpnts.mat;` lädt eine lange Liste mit $\lambda\phi$-Koordinaten eines GPS-Tracks. Rechnen Sie diese Koordinaten in Pixel-xy-Koordinaten um und zeichnen Sie diese.

Ü 5.4 MATLAB-Werkzeuge zum Anpassen von Funktionen an Daten

(Bonus-Material für Interessierte: Im Vergleich zu älteren Versionen bietet MATLAB inzwischen tolle neue Werkzeuge. Schaut euch 's an, es zahlt sich aus!)

Zur MATLAB-Grundausstattung gehört das *Basic Fitting Tool*. Damit sollten Sie unbedingt umgehen können. Machen Sie sich gleich einmal damit vertraut: Finden Sie die entsprechende Anleitung in der MATLAB-Hilfe (Vers. 2022b) unter MATLAB > Data Import and Analysis > Descriptive Statistics > Interactive Fitting oder (geht schneller) suchen Sie direkt nach dem Stichwort *Interactive Fitting*.

Sie finden dort ein durchgearbeitetes Beispiel, das die Verwendung des *Basic Fitting Tools* für interaktives Anpassen von Kurven an Datenpunkte erklärt.



Arbeiten das Beispiel in der MATLAB-Hilfe durch und stellen Sie die Approximation graphisch dar, etwa so wie in der nebenstehenden Abbildung.

Darüber hinaus bietet MATLAB in der *Statistics and Machine Learning Toolbox* und in der *Curve Fitting Toolbox* viele Werkzeuge zur Datenanpassung, weit mehr als wir in diesen Übungen behandeln können. Die Aufgaben 45 und 47 zeigen beispielhaft, welche Möglichkeiten die Befehle `cftool` und `polytool` bieten.

Aufgabe 45: Basic Fitting Tool im Vergleich zu Curve Fitting Toolbox

Hier lösen Sie das nichtlineare Ausgleichsproblem aus Aufgabe 42 mit MATLAB-Toolboxen.

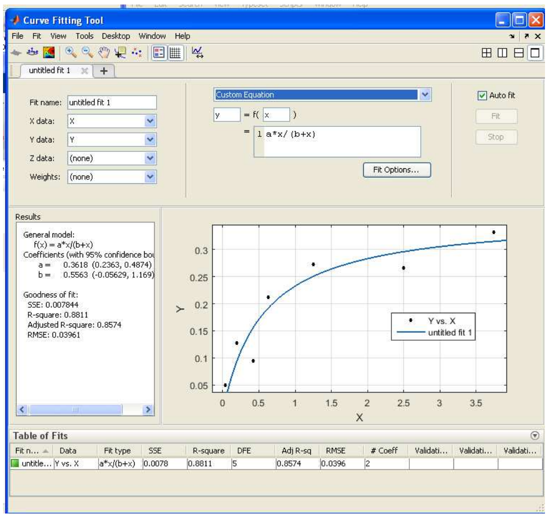
Legen Sie mit MATLABs *basic fitting tool* eine Ausgleichsgerade, ein quadratisches und dann ein kubisches Ausgleichspolynom durch folgende Datenpunkte und plotten Sie die Ergebnisse.

```
X = [0.038 0.194 0.425  0.626  1.253  2.500  3.740];
Y = [0.05  0.127 0.094 0.2122 0.2729 0.2665 0.3317];
```

Es handelt sich hier um die Daten aus Aufgabe 42; dort finden Sie auch eine Abbildung mit gut angepasster Kurve. Welche anderen Optionen des *basic fitting tool* liefern plausible Kurven? Was bedeuten die Fehlermeldungen, die Sie ab Grad 7 bekommen?

Es stellt sich heraus: Kein Ausgleichspolynom kann die Datenpunkte befriedigend approximieren. Das liegt teils daran, dass die Datenpunkte aufgrund der Messunsicherheit weit gestreut sind, aber auch daran, dass sich im gemessenen Experiment die y-Werte mit zunehmendem x asymptotisch einem konstanten Wert nähern. Kein Polynom kann so ein asymptotisches Verhalten beschreiben.

Der Befehl `cftool` öffnet die *Curve Fitting App*. Hier können Sie neben Polynomen auch weitere Modellfunktionen anpassen. Versuchen Sie es: Wählen Sie links oben unter *X Data* und *Y Data* die entsprechenden Datenvektoren aus den im Workspace vorhandenen Variablen. Oben Mitte können Sie den Funktionstyp wählen. Probieren Sie zuerst *Polynomial* mit verschiedenen Polynom-Graden. Wählen Sie anschließend *Custom Equation* und geben Sie den Funktionstern `a*x/(b+x)` ein. (Das ist die Modellfunktion aus Aufgabe 42!) Ihre Anpassung sollte so aussehen wie hier gezeigt.

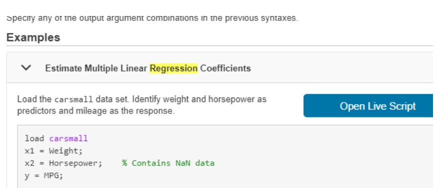


Zusätzlich zu den berechneten Werten der Koeffizienten a und b liefert MATLAB auch ein Konfidenzintervall. Interpretation: Falls den Daten tatsächlich ein Modell der Form $y = ax/(b + x)$ zugrunde liegt, aber die y-Daten zufallsbedingt verrauscht sind, dann berechnet MATLAB *Schätzungen* $\hat{a}$ und $\hat{b}$ für die Parameter a beziehungsweise b. Zum Beispiel:

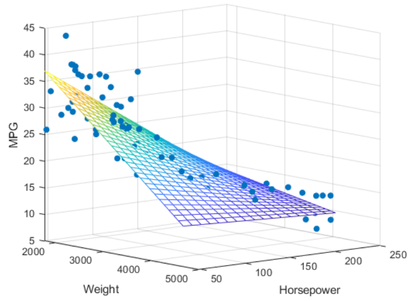
```
Coefficients (with 95% confidence bounds):
       a =       0.3618  (0.2363, 0.4874)
       b =       0.5563  (-0.05629, 1.169)
```

Die Interpretation „Die tatsächlichen Werte von a und b liegen mit 95%-iger Sicherheit im berechneten Intervall“ ist so nicht korrekt; die tatsächlichen Werte von a und b liegen entweder drinnen oder nicht – da ist kein Spielraum für Wahrscheinlichkeiten. Es ist umgekehrt: die berechneten Grenzen des Konfidenzintervalls sind unsicher. Jede unabhängige Wiederholung der Messung liefert (für die immer gleichen Modellparameter a und b) einen neuen Satz von Datenpunkten mit anderen zufallsbedingten Fehlern. MATLABs Schätz-Methode berechnet dazu Konfidenzintervalle mit jeweils etwas anderen Grenzen. Meistens (in 95% der Fälle) schätzt Matlab die Fehlergrenzen richtig, aber in 5% der Fälle überdeckt das berechnete Konfidenzintervall die wahren Werte nicht.

Aufgabe 46: Der Befehl *regress* aus der Statistics Toolbox



Die Matlab-Hilfe Zum Stichwort „regress“ zeigt ein Beispiel mit Daten zu Gewicht, Motorleistung und Benzinverbrauch von Autos und findet dazu eine Anpassungs-Funktion. Vergleichen Sie mit Aufgabe 39: dort wird ein Datenmodell derselben Art berechnet, die Form der Matrix ist gleich.



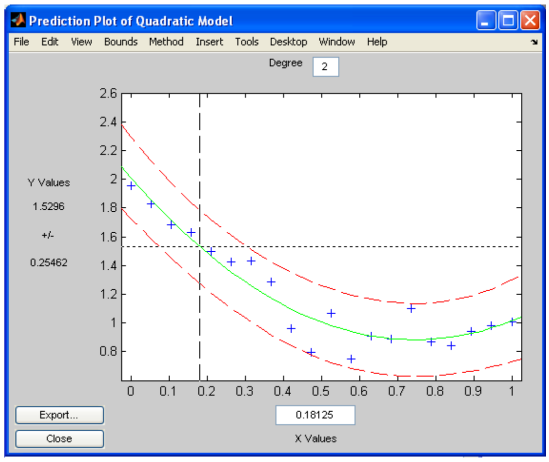
Sie können in der MATLAB-Hilfe auf *Open Live Script* klicken und gelangen so in den *Live Editor*, der mehr Möglichkeiten bietet als der Standard-Editor. Die Dateiendung `*.mls` kennzeichnet Live-Skript-Dateien.
Wenn Sie neugierig sind, probieren Sie den Live-Editor aus. Wenn Sie in der gewohnten Arbeitsumgebung bleiben wollen, speichern Sie dieses Musterbeispiel mittels *save as* als Dateityp *MATLAB Code Files*, Dateiendung `*.m`

Orientieren Sie sich an diesem Muster und erstellen Sie für die Daten aus Aufgabe 39 (Durchmesser, Ausgangstemperatur, Kochzeit) mit dem `regress`-Befehl eine Anpassung und eine ähnliche Graphik wie oben (mit Kochzeit als z-Achse).

Aufgabe 47: Der Befehl *polytool* aus der Statistics toolbox

Die folgenden MATLAB-Befehlszeilen erzeugen entlang der Funktion $y = 2 - 3x + 2x^2$ Datenpunkte, die durch zufällige, normalverteilte Störungen verrauscht sind.

```
n=20;
x=linspace(0,1,n);
y=2 - 3*x + 2*x.^2 + 0.1*randn(1,n);
```



Der Befehl `polytool(x,y)` öffnet ein Fenster ähnlich dem basic fitting tool. Erzeugen Sie damit ein Bild wie die nebenstehende Abbildung und erklären Sie:

- Was bedeuten die verschiedenen Kurven?
- Wozu dient das verschiebbare Achsenkreuz, was sind die angezeigten X-Values und Y-Values?
- Wenn Sie auf Export... klicken, was bedeuten „Parameters“ und Parameters CI“?