Computational Engineering und Robotik

Prof. Jan Peters, J. Carvalho, P. Klink, P. Liu, S. Stark

Sommersemester 2020

Lösungsvorschlag der 8. Übung

1 Kondition und numerische Stabilität (6 Punkte)

a) Berechne für die folgenden Funktionen jeweils die Konditionszahlen. Gib an, in welchen Teilen der Wertebereiche die Funktionen Eingangsfehler stark bzw. schwach verstärken.

- $f_1(x) = \cos(x),\ x \in [0, \pi/2)$
- $f_2(x) = 1 + x,\ x \in \mathbb{R}$
- $f_3(x_1, x_2) = x_1^2 + x_2^2,\ \boldsymbol{x} \in \mathbb{R}^2$
- $f_4(x) = e^{x-1},\ x \in (1, +\infty)$

Lösungsvorschlag: Für f_1 lautet die Konditionszahl für x:

$$\text{cond}_{f_1}(x) = \left|\frac{x}{\cos x}(-\sin x)\right| = |x \tan x|$$

$x \to 0 \implies \text{cond}_f(x) = 0$ ist dann f gut konditioniert. $x \to \pi/2 \implies \text{cond}_f(x) = \infty$ ist dann f schlecht konditioniert. (0.5 Punkt(e))

Für f_2 lautet die Konditionszahl für x:

$$\text{cond}_{f_2}(x) = \left|\frac{x}{1+x}\right|$$

$x \to -1 \implies \text{cond}_f(x) = \infty$ ist dann f schlecht konditioniert um -1, und sonst gut konditioniert.(0.5 Punkt(e))

Für f_3 lautet die Konditionszahl für x_1:

$$\text{cond}^1_{f_3}(x) = \left|2\frac{x_1^2}{x_1^2 + x_2^2}\right| \leq 2,$$

und für x_2:

$$\text{cond}^2_{f_3}(x) = \left|2\frac{x_2^2}{x_1^2 + x_2^2}\right| \leq 2.$$

Also ist dann f gut konditioniert für alle $\boldsymbol{x}$.(0.5 Punkt(e))

Für f_4 lautet die Konditionszahl für x:

$$\text{cond}_{f_4}(x) = \left|\frac{x}{e^{x-1}}e^{x-1}\right| = |x|$$

$x \to +\infty \implies \text{cond}_f(x) = \infty$ ist dann f schlecht konditioniert für große x.(0.5 Punkt(e))

b) Die `softmax` ist eine Funktion, die häufig beim maschinellen Lernen verwendet wird, um die Ausgabe eines neuronalen Netzes in eine diskrete Wahrscheinlichkeitsverteilung umzuwandeln. Wenn die Ausgabe eines neuronalen Netzes $\boldsymbol{x} \in R^d$ ist, dann ist der `softmax`-Operator, `softmax` $: R^d \to R^d$, und der k-te Eintrag des Ausgangsvektors ist

$$\text{softmax}(\boldsymbol{x})_k = \frac{e^{x_k}}{\sum_{j=1}^{d} e^{x_j}}$$

Typischerweise können die Einträge des Vektors $\boldsymbol{x}$ sehr groß sein, z.B. $10^4 < x_j < 10^6$. Die Implementierung des `softmax`, wie in der obigen Funktion angegeben, ist problematisch. Warum ist das der Fall?
Nehmen wir an, dass, obwohl die Einträge in $\boldsymbol{x}$ groß sind, die paarweise Differenz klein ist, d.h. $|x_i - x_j| \leq 1,\ \forall i,j \in \{1,\dots,d\}$. Gib eine modifizierte Version des `softmax` an, so dass seine Berechnung immer noch korrekt, aber stabil ist.

Hinweis: Du brauchst die Begriffe Konditionszahlen oder Rundungsfehler nicht zu verwenden. Denke erst mal über Numerische Überläufe.

Lösungsvorschlag: Die Berechnung von e^{x_j} für große x_j geht bis ins Unendliche und kann daher problematisch sein.(0.5 Punkt(e))

Eine Möglichkeit, dies zu umgehen, besteht darin, den größten Eintrag von $\boldsymbol{x}$ zu subtrahieren. Dann brauchen wir nur noch $e^{x_j - \max(\boldsymbol{x})}$ zu berechnen, was angesichts der Begrenzung auf den paarweisen Abstand garantiert, dass die Berechnung stabil ist.

$$\begin{aligned}
\max(\boldsymbol{x}) &= \max(x_1,\dots,x_d) \\
\text{softmax}(\boldsymbol{x})_k &= \frac{e^{x_k}}{\sum_{j=1}^{d} e^{x_j}} \\
&= \frac{e^{-\max(\boldsymbol{x})}}{e^{-\max(\boldsymbol{x})}} \frac{e^{x_k}}{\sum_{j=1}^{d} e^{x_j}} \\
&= \frac{e^{x_k - \max(\boldsymbol{x})}}{\sum_{j=1}^{d} e^{x_j - \max(\boldsymbol{x})}}
\end{aligned}$$

(1.5 Punkt(e))

c) Wir vergleichen die numerische Stabilität der Funktionen $g_1(x) = x^n - x^{n-1}$ und $g_2(x) = x^{n-1}(x-1)$ für $n \in \mathbb{N}_{\geq 2}$ und prüfen, welche Funktion für die numerische Auswertung an der Stelle $x \approx 1$ besser geeignet ist. Hier ist $x^n = x \cdot x \cdots x$. Betrachte dazu den relativen Fehler, der durch die Auswertung der Funktionen in Gleitpunktarithmetik $\text{rd}(g_i(x))$ für $i \in \{1,2\}$ an der Stelle $x \to 1$ entsteht.

Verwende für die Rechenoperation $\diamondsuit$ den Ausdruck $\text{rd}(a \diamondsuit b) = (a \diamondsuit b)(1+\varepsilon)$ zur Berechnung der Auswertung in Gleitpunktarithmetik.

Lösungsvorschlag: Der absoluter Fehler für g_1, g_2 ist gegeben durch

$$\begin{aligned}
\Delta g_1(x) = |\text{rd}(g_1(x)) - g_1(x)| &= \left|\text{rd}(\text{rd}(x^n) - \text{rd}(x^{n-1})) - g_1(x)\right| \\
&= \left|\left(x^n(1+\varepsilon)^{n-1} - x^{n-1}(1+\varepsilon)^{n-2}\right)(1+\varepsilon) - \left(x^n - x^{n-1}\right)\right| \\
&= \left|\left(x^n - x^{n-1}\right)\left((1+\varepsilon)^{n-1} - 1\right) + x^n \varepsilon (1+\varepsilon)^{n-1}\right|
\end{aligned}$$

(0.75 Punkt(e))

und

$$\begin{aligned}\Delta g_2(x) = |\mathrm{rd}(g_2(x)) - g_2(x)| &= \left|\mathrm{rd}(\mathrm{rd}(x^{n-1})\cdot \mathrm{rd}(x-1)) - g_2(x)\right| \\ &= \left|\mathrm{rd}(x^{n-1}(1+\varepsilon)^{n-2}\cdot(x-1)(1+\varepsilon)) - x^{n-1}(x-1)\right| \\ &= \left|x^{n-1}(1+\varepsilon)^{n-2}(x-1)(1+\varepsilon)^2 - x^{n-1}(x-1)\right| \\ &= \left|x^{n-1}(x-1)\left((1+\varepsilon)^n - 1\right)\right|.\end{aligned}$$

(0.75 Punkt(e))

Damit ergibt sich für den relativen Fehler

$$\lim_{x\to 1}\left|\frac{\Delta g_1(x)}{g_1(x)}\right| = \lim_{x\to 1}\left|\left((1+\varepsilon)^{n-1}-1\right) + \frac{x^n}{x^n - x^{n-1}}\cdot\varepsilon(1+\varepsilon)^{n-1}\right| = \left|\left((1+\varepsilon)^{n-1}-1\right) + \frac{x\varepsilon(1+\varepsilon)^{n-1}}{x-1}\right| = \infty$$

bzw.

$$\lim_{x\to 1}\left|\frac{\Delta g_2(x)}{g_2(x)}\right| = \lim_{x\to 1}\left|(1+\varepsilon)^n - 1\right| = (1+\varepsilon)^n - 1.$$

Folglich ist die Funktion g_2 an der Stelle $x = 1$ stabiler und daher für die Auswertung nahe 1 besser geeignet. (0.5 Punkt(e))

2 Systemidentifikation (4 Punkte)

Betrachte im Folgenden einen mobilen Roboter, den wir modellieren werden, mit einem Punkt-Masse-System mit Masse $m = 1.0\,\mathrm{kg}$, und angenommen, dass **es gibt keine anderen Kräfte, wie z.B. Reibung**. Der Roboter verfügt über ein einziges Antriebssystem, das es ihm erlaubt, sich nur in x-Richtung zu bewegen. Dieses Antriebssystem wird mit einer konstanten Kraft $F > 0$ angetrieben.

Wir kennen den Wert der Kraft F nicht, aber wir werden versuchen, ihn mit Hilfe von Sensoren gesammelten Daten abzuschätzen. Dazu montieren wir auf dem Roboter einen IMU (Inertial Measurement Unit) Sensor, welche die einwirkenden Beschleunigungen misst. Durch Integration der Beschleunigungswerte wird die relative Position zu einem definierten Ursprung bestimmt.
Folgende Roboterpositionen $x(t_i)$ (Tabelle **??**) wurden im Versuch ermittelt, bei dem der Roboter **zu Beginn still stand**:

Messwert i	0	1	2	3	4	5
t_i [s]	0	1	2	3	4	5
$x_{\mathrm{IMU},i} = x_{\mathrm{IMU}}(t_i)$ [m]	0	0.31	0.78	2.17	4.12	5.86

Tabelle 1: IMU Messungen in den ersten 5 Sekunden.

a) Stelle die Dynamikgleichung des Sytems auf. Danach, löst die für $x(t)$ im Allgemein (d.h. die Anfangswerte sind nicht immer Null).

Lösungsvorschlag: Nach dem Newton'schen Gesetz lässt sich die Beschleunigung unter Einwirkung einer Kraft wie folgt modellieren:

$$m\cdot\ddot{x}(t) = F \quad\Rightarrow\quad \ddot{x}(t) = \frac{F}{m}$$

(0.5 Punkt(e))

Die allgemeine Lösung für $\dot{x}(t)$ lässt sich jetzt durch Integrieren bestimmen:

$$\dot{x}(t) = \frac{F}{m} \cdot t + v_0$$

(0.25 Punkt(e))

Die Bewegungsgleichung lässt sich damit durch erneutes Integrieren ermitteln:

$$x(t) = \frac{F}{2m} \cdot t^2 + v_0 \cdot t + x_0$$

(0.25 Punkt(e))

b) Nun wollen wir die konstante Kraft F finden, die die Ergebnisse aus Tabelle **??** hervorgebracht hat. Die Gleichung aus Aufgabe a) setzt die modellierte Position ($\tilde{x}_i$) zu jedem Zeitpunkt in Beziehung zu der aufgebrachten Kraft. Dazu definieren wir eine konvexe Kostenfunktion L. Finde F so, dass L minimiert wird.

$$F^* = \arg\min_F L(F) = \arg\min_F \frac{1}{2} \sum_{i=0}^{n-1} (\tilde{x}(t_i, F) - x_{\text{IMU}}(t_i))^2$$

Lösungsvorschlag: Aus der Tabelle lässt sich $x_0 = 0$ ablesen. Weiterhin ist nach der Aufgabenstellung $v_0 = 0$. Damit ergibt sich die folgende Verfahrensvorschrift:

$$L(F) = \frac{1}{2} \cdot \sum_{i=0}^{5} (\frac{F}{2m} \cdot t_i^2 - x_i)^2$$

(0.5 Punkt(e))

Ein potentielles Minimum tritt auf, wenn die Ableitung $\frac{L(F)}{dF} = 0$ ist.

$$\begin{aligned}
\frac{dL(F)}{dF} &= \sum_{i=0}^{5} (\frac{F}{2m} \cdot t_i^2 - x_i)(\frac{1}{2m} t_i^2) = \sum_{i=0}^{5} (\frac{F}{4m^2} \cdot t_i^4 - \frac{1}{2m} t_i^2 x_i) \\
\Leftrightarrow \frac{dL(F)}{dF} &= F \sum_{i=0}^{5} \frac{1}{4m^2} \cdot t_i^4 - \sum_{i=0}^{5} \frac{1}{2m} t_i^2 x_i = 0 \\
\Leftrightarrow F &= 2m \frac{\sum_{i=0}^{5} t_i^2 x_i}{\sum_{i=0}^{5} t_i^4}
\end{aligned}$$

(0.25 Punkt(e))

Mit den konkreten Werten für m, x_i und t_i erhält:

$$F = 0.48086\,\text{N}$$

(0.25 Punkt(e))

c) Jetzt messen wir auch die Position mit einem Lasersensor, um sie mit den IMU-Messwerten zu vergleichen. Die Lasermesswerte sind in Tabelle **??** aufgeführt. Mit der in b) berechneten Kraft können wir die Position jederzeit mit der Model aus a) zu berechnen. Vergleiche die mit dem Modell erhaltene Position mit der des Lasersensors zu den 5 Zeitpunkten aus Tabelle **??**. Berechne auch den Root-Mean-Square-Error (RMSE), um die durchschnittliche Abweichung der Modellschätzung von der Messung abzuschätzen.

$$\varepsilon_{\text{RMSE}} = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (\tilde{x}(t_i, F) - x_{\text{laser}}(t_i))^2}$$

Messwert i	0	1	2	3	4	5
t_i [s]	0	1	2	3	4	5
$x_{\text{laser},i} = x_{\text{laser}}(t_i)$ [m]	0	0.305	0.71	2.12	4.18	5.84

Tabelle 2: Lasermessungen in den ersten 5 Sekunden

Lösungsvorschlag: Über die in der letzten Teilaufgabe ermittelte Kraft F erhält man die Bewegungsgleichung zur modellbasierten Schätzung der Position $\tilde{x}(t)$:

$$\tilde{x}(t) = \frac{F}{2m} \cdot t^2 = \frac{0.48086}{2.} \cdot t^2 \approx 0.240428 \cdot t^2.$$

Daraus lassen sich folgende Werte berechnen:

Messwert i	1	2	3	4	5	6
t_i [s]	0	1	2	3	4	5
$\tilde{x}(t_i)$ [m]	0	0.2405	0.9618	2.1641	3.8472	6.0113

(0.5 Punkt(e))

Daraus ergibt sich ein RMSE von:

$$\begin{aligned}\varepsilon_{\text{RMSE}} &= \sqrt{\frac{1}{6}\sum_{i=0}^{5}(\tilde{x}(t_i) - x_{laser}(t_i))^2} \\ &\approx 0.186896\end{aligned}$$

(0.5 Punkt(e))

d) Wenn jemand dich bitten würde, eine Vorhersage zu machen, in welcher Position sich den echten Roboter bei 100 Sekunden befinden wird, was würdest du sagen?

Lösungsvorschlag: Wir können die Position bei $t = 100$ mit dem uns vorliegenden Modell berechnen. Obwohl diese Vorhersage wahrscheinlich völlig falsch wäre, da wir ein paar Daten für ungefähr die ersten 5 Sekunden haben, ist die berechnete Kraft vielleicht nicht die wahre. Außerdem ist unser Modell eine sehr vereinfachte Abstraktion des realen Systems. Es gibt mehrere Kräfte und Effekte, die wir nicht berücksichtigen und die möglicherweise die gesamte Dynamik beeinflussen würden. (1.0 Punkt(e))