Universität Konstanz
Fachbereich Mathematik und Statistik
Sommersemester 2019

Alexander Taveira Blomenhofer
Markus Schweighofer

Übungsblatt 1 zur Polynomialen Optimierung

Aufgabe 1 (24 Punkte). Betrachte das polynomiale Optimierungsproblem

$$(P) \qquad \text{minimiere } x^3 + 9x^2 + 24x + 15 \text{ über } x \in \mathbb{R} \text{ mit } x + 3 \geq 0.$$

(a) Füge zu (P) die beiden Scharen von redundanten Ungleichungen

$$(ax + b)^2 \geq 0 \qquad (a, b \in \mathbb{R}) \qquad \text{und} \tag{1}$$
$$(ax + b)^2(x + 3) \geq 0 \qquad (a, b \in \mathbb{R}) \tag{2}$$

hinzu.

(b) Finde symmetrische Matrixpolynome $P_1 \in S\mathbb{R}[X]^{2\times 2}$ und $P_2 \in S\mathbb{R}[X]^{2\times 2}$ so, dass (1) und (2) durch die positive Semidefinitheit von $P_1(x)$ und $P_2(x)$ ausgedrückt werden.

(c) „Linearisiere“ (P) zu einem semidefiniten Programm

$$\begin{aligned}(P_1) \qquad \text{minimiere } & y_2 + 9y_1 + 24x + 15 \\ \text{über } & x, y_1, y_2 \in \mathbb{R} \\ \text{mit } & M_1(x, y_1) \succeq 0 \\ & M_2(x, y_1, y_2) \succeq 0,\end{aligned}$$

wobei $M_1 \in S\mathbb{R}[X, Y_1]^{2\times 2}$ und $M_2 \in S\mathbb{R}[X, Y_1, Y_2]^{2\times 2}$ lineare Matrixpolynome sind.

(d) Verschaffe Dir Zugriff auf einen Rechner mit einer Version von MATLAB[1], die eine Symbolic Math Toolbox[2] beinhaltet, welche durch MuPAD[3] realisiert ist. Falls Du einen eigenen Rechner hast, kannst Du hierfür die kostenlose MATLAB-Landeslizenz[4] des Rechenzentrums nutzen. Falls Du keinen eigenen Rechner hast kannst Du Dir Zugang zum PhyMa-Rechnerpool[5] verschaffen. Mit dem Befehl `mupad` im Kommandofenster von MATLAB kannst Du testen, ob MuPAD läuft. Installiere eine aktuelle Version des kostenlosen MATLAB-Pakets YALMIP[6] (oder bespreche die Installation gegebenenfalls mit den Administratoren des PhyMa-Pools). Vergesse nicht, die Verzeichnisse von YALMIP in den Suchpfad von MATLAB aufzunehmen. Teste mit dem Befehl `yalmip('version')`, ob YALMIP lauffähig ist.

[1] `http://de.wikipedia.org/wiki/Matlab`
[2] `http://de.mathworks.com/products/symbolic/`
[3] `http://de.wikipedia.org/wiki/MuPAD`
[4] `https://www.kim.uni-konstanz.de/services/software-und-hardware/matlab/`
[5] `http://phyma.uni-konstanz.de/`
[6] `https://yalmip.github.io`

Beantrage eine freie akademische Lizenz für MOSEK und installiere MOSEK (achte dabei darauf, dass die Lizenzdatei an der richtigen Stelle liegt[7] und gefunden wird). Teste mit dem Befehl `mosekopt`, ob MOSEK lauffähig ist. Teste nun mit dem Kommando `yalmiptest`, ob YALMIP den SDP-Solver von MOSEK gefunden hat:

```
>> yalmiptest
+++++++++++++++++++++++++++++++++++++++++++++
|       Searching for installed solvers       |
+++++++++++++++++++++++++++++++++++++++++++++
|        Solver|   Version/module|      Status|
+++++++++++++++++++++++++++++++++++++++++++++
...
|         MOSEK|             SOCP|       found|
|         MOSEK|              SDP|       found|
|         MOSEK|        GEOMETRIC|       found|
...

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|                   Test|   Solution|                 Solver message|
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|   Core functionalities|        N/A|   Successfully solved (YALMIP)|
|                     LP|    Correct|    Successfully solved (MOSEK)|
|                     LP|    Correct|    Successfully solved (MOSEK)|
...
|                    SDP|    Correct|    Successfully solved (MOSEK)|
|                    SDP|    Correct|    Successfully solved (MOSEK)|
|                    SDP|    Correct|    Successfully solved (MOSEK)|
|                    SDP|    Correct|    Successfully solved (MOSEK)|
...
|      Moment relaxation|    Correct|    Successfully solved (MOSEK)|
|         Sum-of-squares|    Correct|    Successfully solved (MOSEK)|
...
```

Installiere mindestens einen weiteren von YALMIP unterstützten SDP-Solver: CSDP, DSDP, LOGDETPPA, PENLAB, SDPA, SDPLR, SDPT3, SDPNAL, SEDUMI oder PENSDP[8]. Arbeite die YALMIP-Tutorials die Abschnitte "Getting started", "Linear programming" und "Semidefinite programming" durch.

(e) Lege eine Datei `hellosdp.m` an, die mit den Zeilen `sdpvar x` und `y=sdpvar(2,1)` beginnt. Definiere die Zielfunktion von (P_1) und die Matrizen M_1 und M_2 in den folgenden Zeilen dieser Datei. Löse (P_1) mit einem Aufruf von `optimize`.

[7] `http://docs.mosek.com/7.1/toolsinstall/Obtaining_and_installing_a_license.html`
[8] `https://yalmip.github.io/allsolvers/`

(f) Benutze `value`, um den berechneten Optimalwert von (P_1) auszugeben. Stimmt er mit dem Optimalwert von (P) (bis auf numerische Fehler) überein?

(g) Linearisiere die Scharen (1) und (2) zu Scharen von linearen Ungleichungen. Auf diese Weise kann man (P_1) als ein „unendliches lineares Programm“ mit unendlich vielen Nebenbedingungen interpretieren. Bezeichne

$$(x^*, y_1^*, y_2^*) = (\texttt{value(x)}, \texttt{value(y(1))}, \texttt{value(y(2))})$$

die errechnete optimale Lösung. Wie kann man aus den Matrizen

$$M_1(x^*, y_1^*) = \texttt{value(M1)} \text{ und } M_2(x^*, y_1^*, y_2^*) = \texttt{value(M2)}$$

erfahren, welche dieser unendlich vielen Nebenbedingungen in (x^*, y_1^*, y_2^*) *aktiv* sind (das heißt dort mit Gleichheit gelten)?

(h) Finde mit Hilfe der Erkenntnisse aus (g) endlich viele Ungleichungen der Form (1) und (2) derart, dass die zugehörige Linearisierung (P_2) von (P) ein lineares Programm mit $P_2^* = P^*$ ist. Löse dieses lineare Programm mit YALMIP.

Bemerkungen: Es ist geplant, reguläre Übungsblätter am 15. April, 26. April, 10. Mai, 24. Mai, 7. Juni und 21. Juni auszugeben. Die Bearbeitungszeit beträgt jeweils zwei Wochen (dieses Mal etwas weniger). Diese Blätter müssen schriftlich bearbeitet werden und werden bepunktet. Erstellter MATLAB-Code wird bei diesen Blättern nicht elektronisch abgegeben, sollte aber schriftlich vorliegen (handschriftlich oder beigehefteter Ausdruck). Auch zu Aufgaben wie 1(d) und 1(e) auf diesem Blatt müssen die gemachten Beobachtungen und überwundenen Probleme schriftlich dokumentiert werden. Jeder Teilnehmer muss die Lösungen eigenständig und individuell aufschreiben. Zusammenarbeit ist erlaubt, aber nicht beim Aufschreiben. Es müssen etwa die Hälfte der Punkte erreicht werden und der Teilnehmer muss in der Lage sein, seine Lösung in der Übungsgruppe zu präsentieren und zu erklären.

Zusätzlich ist geplant, am 26. April, 17. Mai und 14. Juni Programmieraufgaben auszugeben, die nur elektronisch in Form von gut dokumentiertem und lauffähigem MATLAB-Code (bzw. MuPAD-Code) abgegeben werden können. Dort beträgt die Bearbeitungszeit etwa zwischen zwei und vier Wochen.

Inwiefern bei den Programmieraufgaben Zusammenarbeit möglich ist, wird noch bekanntgegeben. Auch hier muss etwa die Hälfte der Punkte erreicht werden. Am Ende des Semesters gibt es eine Prüfung (wahrscheinlich schriftliche Klausur, eventuell am 2. August). Die erreichten Punkte in den regulären Übungen und den Programmierübungen werden in die erreichte Endnote eingerechnet.

Abgabe bis Freitag, den 26. April 2019, um 9:59 Uhr in die Zettelkästen neben F411.