SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta chemickej a potravinárskej technológie

Evidenčné číslo: FCHPT-5415-44240

Prístupy k minimalizácii po častiach lineárnych funkcii

Bakalárska práca

2013 **Ján Minárik**

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta chemickej a potravinárskej technológie

Evidenčné číslo: FCHPT-5415-44240

Prístupy k minimalizácii po častiach lineárnych funkcii

Bakalárska práca

Študijný program: automatizácia, informatizácia a manažment v chémii a potravinárstve
Číslo študijného odboru: 2621
Názov študijného odboru: 5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo
Školiace pracovisko: Ústav informatizácie, automatizácie a matematiky
Vedúci záverečnej práce: Ing. Juraj Števek, PhD.

Bratislava 2013 **Ján Minárik**

Slovenská technická univerzita v Bratislave
Ústav informatizácie, automatizácie a matematiky

Fakulta chemickej a potravinárskej technológie
Akademický rok: 2012/2013
Evidenčné číslo: FCHPT-5415-44240

STU
FCHPT

ZADANIE BAKALÁRSKEJ PRÁCE

Študent:	**Ján Minárik**
ID študenta:	44240
Študijný program:	automatizácia, informatizácia a manažment v chémii a potravinárstve
Kombinácia študijných odborov:	5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo
Vedúci práce:	Ing. Juraj Števek, PhD.
Miesto vypracovania:	ÚIAM FCHPT

Názov práce: **Prístupy k minimalizácii po častiach lineárnych funkcii**

Špecifikácia zadania:

Prediktívne riadenie (Model Predictive Control – MPC) patrí medzi intenzívne študované oblasti automatizácie v dnešnej dobe. Vysoká kvalita riadenia, robustnosť a možnosť zahrnutia ohraničení riadeného systému do riadiaceho algoritmu sú základné výhody algoritmov prediktívneho riadenia.
V poslednej dekáde je vývoj riadiacich algoritmov v tejto oblasti sústredený na splnenie prísnych výpočtových podmienok akými sú: nízke pamäťové nároky (stovky kB) a krátke výpočtové cykly (10ns 10ms). Medzi najperspektívnejšie algoritmy z rodiny metód prediktívneho riadenia patria algoritmy založené na explicitnom prediktívnom riadení (eMPC).
Práca je zameraná na optimalizáciu riadiaceho pravidla explicitného prediktívneho regulátora, ktorý je navrhnutý vo forme po častiach lineárnej funkcie (piecewise linear – PWL).

Jednotlivé ciele a výzvy projektu:

1. Cieľom práce je naštudovať teóriu explicitného prediktívneho riadenia (eMPC) a spôsoby reprezentácie po častiach lineárnych funkcii (piecewise linear – PWL).
2. Cieľom práce je naprogramovať balík funkcií v prostredí Matlab, ktoré budú vykonávať operácie s PWL funkciami:

konverzia medzi vybranými formami PWL funkcie
algebraicka minimalizácia PWL funkcie
geometrická minimalizácia jednorozmernej PWL funkcie
3. Výzvou je navrhnúť minimalizáciu dvojrozmernej PWL funkcie

Zoznam odbornej literatúry:

1. KVASNICA, M. *Real-Time Model Predictive Control via Multi-Parametric Programming:Theory and Tools : Theory and Tools.* Saarbrücken: VDM Verlag Dr. Müller, 2009. 231 s. ISBN 978-3-639-20644-9.
2. KVASNICA, M. – FIKAR, M. Clipping-based complexity reduction in explicit MPC. *IEEE Transactions on Automatic Control Vol. 57, Iss. 7.* s. 1878–1883. ISSN 0018-9286.
3. Geyer, T., Torrisi, F., and Morari, M. (2008). Optimal complexity reduction of polyhedral piecewise affine systems. Automatica, 44(7), 1728–1740.
4. Wen, C., Ma, X., and Ydstie, B.E. (2009). Analytical expression of explicit MPC solution via lattice piecewise-affine function. Automatica, 45(4), 910 – 917.

Riešenie zadania práce od: 18. 02. 2013

Dátum odovzdania práce: 25. 05. 2013

L. S.

Ján Minárik
študent

prof. Ing. Miroslav Fikar, DrSc.
vedúci pracoviska

prof. Ing. Miroslav Fikar, DrSc.
garant študijného programu

Abstrakt

Táto práca sa venuje riešeniu problému explicitného prediktívneho riadenia (eMPC), ako zjednodušit evaluáciu riadiaceho pravidla. Ak je riadenie vyriešené parametricky, dostaneme po častiach afínnu funkciu (PWA), ktorá mapuje vektor stavov do vektora optimálnych akčných zásahov. Venujem sa mriežkovej reprezentácií PWA funkcie. V práci som vytvoril matlabovské algoritmi pomocou, ktorých sa dá táto reprezentácia zjednodušiť. V dôsledku čoho sa urýchli celá evaluácia riadiaceho pravidla.

Kľúčové slová: explicitné prediktívne riadenie, po častiach afínna funkcia, mriežková reprezentácia.

Abstract

This thesis addresses solution of explicit model predictive control (eMPC) problem, specifically to simplification of the explicit control law. If the solution is parametric we deal with piecewise affine function (PWA), which maps vector of states into vector of optimal control inputs. I focus to a lattice representation of PWA function. I created matlab algorithms which simplify this representation. This simplification leads to faster evaluation of the control law.

Keywords: explicit model predictive control, piecewise affine function, lattice PWA.

Obsah

1 Úvod **7**
1.1 Základné definície 8
1.2 Online MPC 10
1.3 Offline MPC 11
1.4 Spôsob evaluácie explicitného prediktívneho riadenia 12
1.4.1 Sekvenčné prehľadávanie 12
1.4.2 Binárne prehľadávacie stromy 12
1.4.3 Intervalové prehľadávacie stromy 12
1.5 PWA funkcia 13

2 Definícia problému **15**
2.1 Existujúce prístupy optimalizácie 15
2.1.1 Zlučovanie regiónov 15
2.1.2 Mriežková reprezentácia 15

3 Návrh riešenia **18**
3.1 Implementácia mriežkovej reprezentácie v Matlabe 18
3.2 Implementácia metód zjedušenia mriežkovej reprezentácie 19
3.2.1 Zjednodušenie po riadkoch (RVS lema) 19
3.2.2 Zjednodušenie po stĺpcoch (CVS lema) 19
3.2.3 Maximálna afínna funkcia (MAF) 21
3.3 Experiment - príklad 22
3.4 Zhodnotenie 23

4 Prílohy **24**

Kapitola 1

Úvod

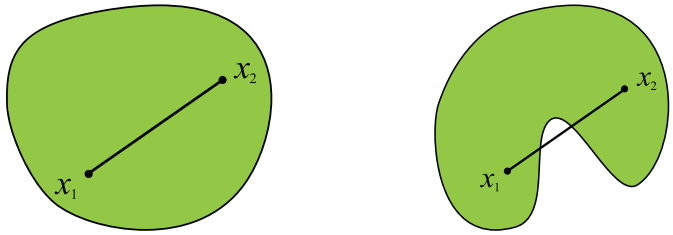
Prediktívne riadenie s modelom (MPC) sa používa všade tam, kde treba riadiť zložitý systém a dodržiavať ohraničenia, ako je napríklad bezpečná teplota, presná koncentrácia alebo limity materiálu. Riadiace pravidlo sa získava riešením ohraničeného optimalizačného problému v každom riadiacom cykle. Ak narastá počet premenných v optimalizačnej úlohe, narastá tiež čas výpočtu, ktorý môže ľahko prekročiť dĺžku riadiaceho cyklu. Je preto nevyhnutné nájsť taký optimalizačný prístup, ktorý bude spĺňať takéto ohraničenie. Jedným z riešení je explicité MPC[1]. Výstupom explicitného MPC je riešenie vo forme počastiach afínnej funkcie(PWA), ktorá je definovaná nad stavovým priestorom optimalizačnej úlohy. Úloha výpočtu optimálneho akčného zásahu sa zredukuje na nájdenie aktívnej afínnej funkcie v tabuľke afínnych funkcií. Bežne používanou metódou na nájdeníe aktívnej afínnej funkcie pre dané x je sekvenčné prehľadávanie, alebo binárne prehľadávacie stromy. V svojej práci sa venujem mriežkovej reprezentácii PWA funkcie[1] z riešenia, získaného z multiparametrického toolboxu[2]. Wenn a kolektív vo svojom článku [1] navrhli dve lemy pomocou ktorých sa dá jednoducho, ale efektívne znížiť zložitosť mriežkovej reprezentácie.

V prvej kapitole uvediem základné definície, ktoré ďalej vo svojej práci používam, vysvetlím prediktívne riadenie s modelom v reálnom čase ako aj explicitné prediktívne riadenie. Druhá kapitola sa venuje pravidlám na zjednodušenie PWA funkcie v mriežkovej reprezentácií. V tretej kapitole demonštrujem efektívnosť tohto zjednodušenia na niekoľkých PWA funkciách.

1.1 Základné definície

V tejto kapitole zadefinujem pojmy, ktoré ďalej budem používať.

Definícia 1.1 Konvexná množina:[3] Množina C je konvexná ak každá úsečka spájajúca 2 body z množiny C leží tiež celá vo vnútry C. $\{x|x = \theta x_1 + (1 - \theta x_2), \forall x_1, x_2 \in C, \forall\theta, 0 \leq \theta \leq 1\} \subseteq C$.



(a) Konvexná množina (b) Nekonvexná množina

Obr. 1.1: Ilustrácia konvexného a nekonvexného útvaru.

Definícia 1.2 Okolie bodu:[4] Okolie bodu $x \in \mathbb{R}^n$ je množina bodov, ktoré ležia vo vnútri gule so stredom v x a polomerom ϵ.

Definícia 1.3 Uzavretá množina:[4] Množina S je uzavretá vtedy, ak každý bod, ktorý nie je z S má okolie ktoré nie je z S.

Definícia 1.4 Ohraničená množina:[4] Množina S je ohraničená v $\mathbb{R}^n$ ak vieme nájsť guľu $B_\epsilon = \{x \in R^n | \|x\|_2 \leq \epsilon\}$ s polomerom $\epsilon \leq \inf$, ktorá obsahuje celú množinu S.

Definícia 1.5 Kompaktná množina:[4] Množina je v $\mathbb{R}^n$ je kompaktná ak je uzavretá aj ohraničená.

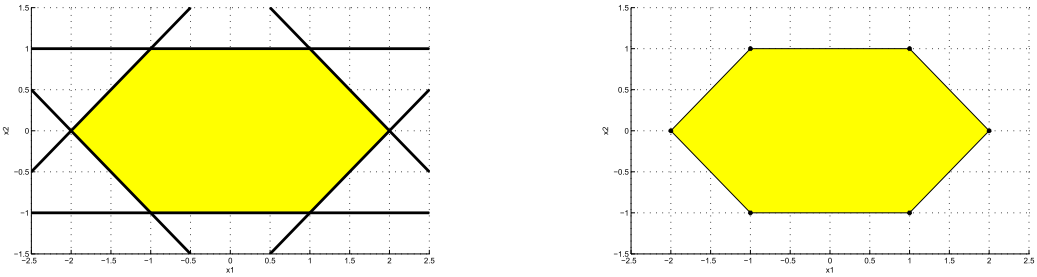
Definícia 1.6 $\mathcal{H}$-Polyhredón:[5] Konvexná množina $S \subseteq \mathbb{R}^n$ rerezentovaná ako prienik konečného počtu uzavretých polpriestorov sa nazýva H-Polyhredón. $\mathcal{S} = \{x \in \mathbb{R}^n | \mathcal{S}^x x \leq \mathcal{S}^0\}$, kde $\mathcal{S}^x \in \mathbb{R}^{q \times n}, \mathcal{S}^0 \in \mathbb{R}^q$, q je pocet pol-priestorov ktoré vytvárajú polyhedrón a operátor $\leq$ vyjadruje porovnanie vektorov po prvkoch.

Definícia 1.7 $\mathcal{H}$-Polytóp:[5] Ohraničený polyhedrón $\mathcal{P} \subseteq \mathbb{R}^n$, $\mathcal{P} = \{x \in \mathbb{R}^n \mid \mathcal{P}^x x \leq \mathcal{P}^\circ\}$ sa nazýva H-Polytóp.

Definícia 1.8 ($\mathcal{V}$-polytop, [5]): Polytop môže byť tiež reprezentovaný konvexnou kombináciou jeho vrcholov V_p:

$$\mathcal{P} = \{x \in \mathbb{R}^n \mid x = \sum_{i=1}^{v_p} a_i V_p^{(i)}, 0 \leq a_i \leq 1, \sum_{i=1}^{v_p} a_i = 1\} \tag{1.1}$$

kde $V_p^{(i)}$ oznaeuje i-tý vrchol $\mathcal{P}$ a v_p je celkový počet vrcholov z $\mathcal{P}$.



(a) $\mathcal{H}$-reprezentácia polytópu (b) $\mathcal{V}$-reprezentácia polytópu

Obr. 1.2: Ilustrácia $\mathcal{V}$-reprezentácie a $\mathcal{H}$-reprezentácie polytópu

Definícia 1.9 (Dimenzia polytopu, [6]): Polytop $\mathcal{P} \subset \mathbb{R}^n$ je dimenzie $d \leq n$, ak existuje d-rozmerná guľa s rádiusom $\epsilon > 0$ obsiahnutá v $\mathcal{P}$ a neexistuje žiadna (d+1)-rozmerná guľa s rádiusom $\epsilon > 0$ obsiahnutá v $\mathcal{P}$.

Definícia 1.10 (Stena, Vrchol, hrana, hrebeň, aspekt, [6]): Lineárna nerovnosť $a^T x \leq b$ platí pre všetky $x \in \mathcal{P}$. Podmnožina $\mathcal{F}$ polyhedronu sa nazýva stena v $\mathcal{P}$, ak môže byť reprezentovaná ako

$$\mathcal{F} = P \cap \{x \in \mathbb{R}^n | a^T x = b\} \tag{1.2}$$

pre niektoré platné nerovnosti $a^T x \leq b$. Steny polyhedronu $\mathcal{P}$ dimenzií 0, 1, (n-2) a (n-1) sa nazývajú vrcholy, hrany, hrebene, respektívne aspekty.

Definícia 1.11 (Minimálna reprezentácia polytópu, [6]): Hovoríme, že polytóp $\mathcal{P} \subseteq \mathbb{R}^n, \mathcal{P} = \{x \in \mathbb{R}^n \mid \mathcal{P}^x x \leq \mathcal{P}^\circ\}$ je minimálnou reprezentáciou ak odstránenie hocijakého z riadkov z $\mathcal{P} \subseteq \mathbb{R}^n, \mathcal{P} = \{x \in \mathbb{R}^n \mid \mathcal{P}^x x \leq \mathcal{P}^\circ\}$ ho zmení. Inými slovami neobsahuje žiadne redudantné riadky.

1.2 Online MPC

V tejto kapitole vysvetlím úlohu prediktívneho riadenia s modelom(MPC) a jeho výhody. Hlavnou myšlienkou MPC, je vytvorenie matematického modelu riadeného procesu, ktorý charakterizuje jeho vývoj počas niekolkých krokov do budúcnosti [6]. Do tohto modelu sme schopný zahrňúť ekonomické, technologické a bezpečnostné kritéria vo forme ohraničení. Potom riešenie optimalizačného problému vedie k vyššiemu ekonomickému výťažku a nižším rizikám pri operácii riadeného zariadenia. Možnosť zahrnúť ohraničenia do úlohy riadenia je jednou z najväčších výhod MPC oproti iným typom riadenia. Cieľom prediktívneho riadenia je vyriešit ohraničený optimalizačný problém, ktorého riešením je sekvencia optimálnych akčných zásahov $\mathbf{u}_n^* = [u_1^*, u_2^*, ..., u_n^*]$, ktorý minimalizuje požadovanú veličinu (napríklad spotrebu paliva) s tým, že rešpektuje všetky stanovené ohraničenia. V praxi sa v i-tom riadiacom kroku vypočíta vektor $\mathbf{u}_n^*$ a aplikuje sa prvý člen vektora akčných zásahov a ostatné sa zahodia.V dalšom kroku sa na základe nových meraní vyrieši optimalizačný problém a znova sa aplikuje iba prvý člen vektora optimálnych akčných zásahov, tento cyklus sa opakuje v každom kroku riadenia. Z tohto mechanizmu vyplýva jeden doležitý predpoklad. A to je vyriešenie celého optimalizačného problému a aplikovanie u_1^* na riadený systém v časovom intervale jedného riadiaceho kroku. V prípade, žeby sme túto operáciu neurobili v danom časovom intervale, môže dôjsť k nestabilite celého systému. Čím máme menší čas medzi jednotlivými akčnými zásahmi, tým potrebujeme výkonejší hardware s príslušným operačným softwarom na riešenie daného optimalizačného problému. Toto je hlavný dôvod prečo je online MPC využívane prevažne na systémy s pomalou dynamikou. Preto keď sa rozhodujeme pre prediktívne riadenie v reálnom čase, musíme sa brať do úvahy dôležité faktory. Zložitosť riadeného systému a jeho modelu, naše očakávania a cenu hardwaru, softwaru ktorý chceme na riadenie použit. Väčšinou chceme vyriešit optimalizačný problém čo najrýchlejšie a pomocou čo najlacnejšieho hardwaru. Napríklad Digital Signal Processors (DSP), Field Programmable Gateway Arrays (FPGA), alebo Programmable Logic Controllers (PLC)[6]. V porovnaní s osobným počítačom, ktorý je vybavený CPU, majú tieto zariadenia menšiu výpočtovú silu a oveľa menej pamäte, kde môžeme ukladať údaje. Práve tu spočíva výzva v návrhu prediktívneho riadenia.

1.3 Offline MPC

Ako som spomenul v predošlej sekcií implemetácia prediktívneho riadenia v reálnom čase, je náročná pre procesy s krátkou periódou riadenia (rádovo milisekundy a menej). Avšak pokiaľ, je riešenie optimalizačného problému urobené parametricky [6] dostaneme explicitné riešenie vo forme regulátora. Takýto regulátor je vo forme PWA funkcie, ktorá mapuje stav x na vektor optimálneho riadiaceho zásahu u^*, pre akúkoľvek hodnotu x z oblasti riešenia. PWA funkcia je uložená v pamäti vo forme tabuľky. Počas riadenia, už nieje potrebné riešit zložitý optimalizačný problém, ale stačí len nájst vhodné riešenie a vyčíslit príslušnú afínnu funkciu. Táto operácia sa dá vykonať v priebehu mili až mikro sekúnd. V prípade použitia multiparametrického toolboxu[2] dostaneme PWA funkciu vo forme $u^* = f_{PWA}(x) = F_r x + G_r$ ak $x \in \mathcal{P}_r$ pre $\mathcal{P}_r = \{x | H_r \leq K_r\}$, kde $r = 1, ..., R$ a R je rovný počtu regiónov nad ktorými je definované riešenie. Táto časť sa vykoná off-line. Pri on-line výpočte sa v každom kroku vyhodnotí $f_{PWA}(x)$ pre $x = x(t)$ v každom riadiacom cykle. Aby explicitné prediktívne riadenie fungovalo v praxi, musia byt splnené 3 kritéria:[6]

1. musí byt možné vypočítat PWA reprezentáciu $f_{PWA}(x)$.
2. musíme byt schopný uložit všetky parametre PWA funkcie v pamäti našej riadiacej jednotky, t.j. matice F_r, G_r, H_r, K_r pre všetky $r = 1, ..., R$.
3. musí byt možné vyčísliť $f_{PWA}(x)$ pre hocijaké $x = x(t)$ v priebehu jedného riadiaceho cyklu.

1.4 Spôsob evaluácie explicitného prediktívneho riadenia

1.4.1 Sekvenčné prehľadávanie

Ide o časovo najdrahší prístup, ktorý spočíva v postupnom prehľadávaní regiónov. Či je $x \in \mathcal{P}_i$ sa postupne overuje pre $i = 1, ..., R$. Ak platí nerovnost $H_r x \leq K_r$ pre všetky riadky matice H_r tak sme našli príslušný región. Potom z pamäte načítame matice F_r a G_r a vyčíslime $F_r x + G_r$. V najhoršom prípade náročnost tohto algoritmu je $\mathcal{O}(R)$, kedže prehľadáme všetkých R regiónov. Z toho vyplýva, že budeme musieť spraviť N_f násobení, N_f sčítaní a N_f porovnaní počas prehľadávania. Následne $m * n$ násobení a m sčítaní na vyčíslenie $F_r x + G_r$. N_f je počet hrán $\{P_r\}_{r=1}^{R}$, alebo $N_f = \sum_{r=1}^{Rc_r}$.

1.4.2 Binárne prehľadávacie stromy

Táto metóda spočíva v rozdelení regiónov $\mathcal{P}_r$ do binárneho prehľadávacieho stromu a tým sa zredukuje zložitosť na $\mathcal{O}(log_2 R)$. Algoritmus je nasledovný. V každom kroku, je nájdená rovina, ktorá rozdelí priestor riešenia pomocou lineárneho programovania. Množina $\{\mathcal{P}_r\}_{r=1}^{R}$ je rozdelená na dve podmnožiny $\mathcal{P}_{I-}$ a $\mathcal{P}_{I+}$ s približne rovnakou kardinalitou. Algoritmus pokračuje novo vzniknutými podmnožinami, až kým kardinalita podmnožiny nieje rovná jednej[6].

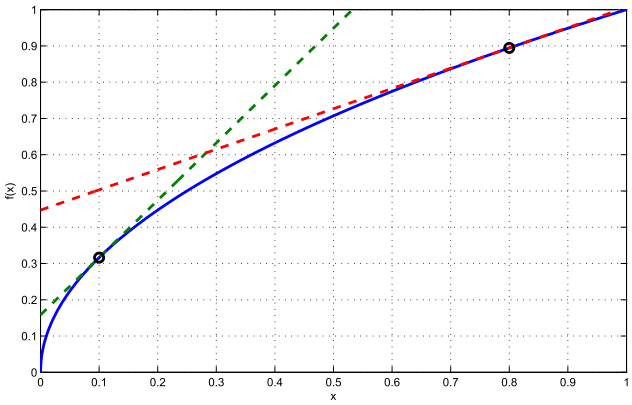
1.4.3 Intervalové prehľadávacie stromy

Idea tejto metódy, je vo výbere niekoľkých oblastí z množiny $\{P_r\}_{r=1}^{R}$, ktoré by mohli obsahovat x_0. Následne sú kandidáti prehľadaný. Samostný strom je prehľadaný so zložitostou $\mathcal{O}(log_2 R)$, avšak teoreticky v najhoršiom prípade môžeme získat až R kandidátov, čo znamená, že časová náročnost može byť až $\mathcal{O}(R)$. Napriektomu prípady z praxe dávaju veľmi dobré výsledky[6]. Tieto tromy sa dajú vytvoriť s použitím $\mathcal{O}(R)$ zložitým lineárnym programovaním, pre porovnanie pri binárnych prehľadávacích stromoch je náročnosť $\mathcal{O}(R^2)$.

1.5 PWA funkcia

Kvalita linearizácie funkcie rastie s počtom bodov v okolí ktorých je funkcia linearizovaná. Označme si $f_{lin,i}(x,u)$ ako linearizáciu $f(x,u)$ okolo i-teho bodu. Ak spravíme n_L týchto aproximácií dostaneme:

$$\mathcal{F} = \begin{bmatrix} f_{LIN,1} \\ f_{LIN,2} \\ \vdots \\ f_{LIN,N_L} \end{bmatrix} \tag{1.3}$$



Obr. 1.3: Ukážka klesajúcej kvality lineárizacie funkcie $f(x) = \sqrt{x}$

Na vybratie ktorá z linearizácií bude použitá na konkrétne u a x použijeme agregátor funkcie. Matematicky povedané $f(x,u) \approx g(F,x,u)$.Ak využijeme agregátor na základe selekcie

$$g(F,x,u) = f_{LIN,i}(x,u) \; ak \; \begin{bmatrix} x \\ u \end{bmatrix} \in \mathcal{P}_i \tag{1.4}$$

kde $\mathcal{P}_i$ značí región x-u priestoru, kde i-ta linearizácia dominuje nad všetkými ostatnými. $\mathcal{P}_i$ možeme chápať tiež ako región platnosti $f_{LIN,i}(x,u)$. Toto sa nazýva po častiach affiná (lineárna) funkcia, skrátene PWA. Z definície je zrejmé, že rozmery matice, v ktorej chceme uchovať koeficienty pri jednotlivých premenných x a u závisia od ich počtu ako aj od počtu regiónov, nad ktorými je PWA funkcia definovaná. Čiže matica parametrov bude mať rozmery $n_L \times (n_x + n_u)$. Z toho vyplýva, že len na uloženie všetkých čísel potrebujeme $m \times (n_x + n_u) \times$ počet bitov na jedno číslo s desatinnou čiarkou pre jeden región PWA funkcie. Toto číslo ešte treba vynásobiť počtom regiónov, aby sme dostali celkový počet bitov ktoré daná PWA funkcie bude v pamäti hardwaru zaberať. Počet regiónov vplýva na najhorší prípad algoritmu, ktorý

hľadá, kde sa x nachádza. Na evaluáciu vplýva veľkost vektora x a u. Druhý spôsob ako uložiť PWA funkciu je v min-max forme. Vtedy je celá PWA funkcia prepísaná do formy minima z maxím afínnych funkcí. V tomto zápise ušetríme pamäť, lebo nemusíme zapisovať matice, ktoré definujú regióny, konkrétne H a K.

Kapitola 2

Definícia problému

Ak chceme zrýchliť evaluáciu PWA funkcia máme na výber buď znížit počet premenných x a u, ale znížit počet regiónov, na ktorých je PWA funkcia definovaná. Prvý prípad je z praktického hľadiska náročný, niekedy až neprijateľný, preto sa venujem druhému spôsobu. V súčastnosti je navrhnutých niekoľko spôsobov ako tento problém riešit, napríklad zlučovanie regiónov[7], clipping filter[8] alebo prepis celej funkcie do mriežkovej reprezentácie[1]. Z týchto metód som si vybral tretiu, a venujem sa matlabovskej implementácií algoritmov na zjednodušie mriežkovej reprezentácie.

2.1 Existujúce prístupy optimalizácie

2.1.1 Zlučovanie regiónov

V explicitnom riešení sa stáva, že niektoré regióny zdielajú rovnaké riadiace pravidlo. Čiže $F_r = F_q$ a $G_r = G_q$. Tieto opakujúce sa matice, len zbytočne zaberajú miesto a preto môžu byť nahradené smerníkom na prvý výskyt matíc F_r a G_r. Regióny s rovnakou afínnou funkciou môžu byť zlúčené do väčších celkov, tak aby sa zachovala konvexnosť regiónov[6]. Výsledok bude nová PWA funkcia $\tilde{f}_{PWA}(x)$, ktorá je ekvivalentná originálu. Inak povedané $\tilde{f}_{PWA}(x) = f_{PWA}(x), \forall x \in \cup_i \mathcal{P}_i$.

2.1.2 Mriežková reprezentácia

Ak je PWA funkcia spojitá na celom definičnom obore, môžeme ju prepísat do tzv.mriežkovej reprezentácie. Mriežková reprezentácia vyjadrí celú PWA funkciu v tvare min-max funkcie. Zadefinujem teraz maticu štruktúry[1]:

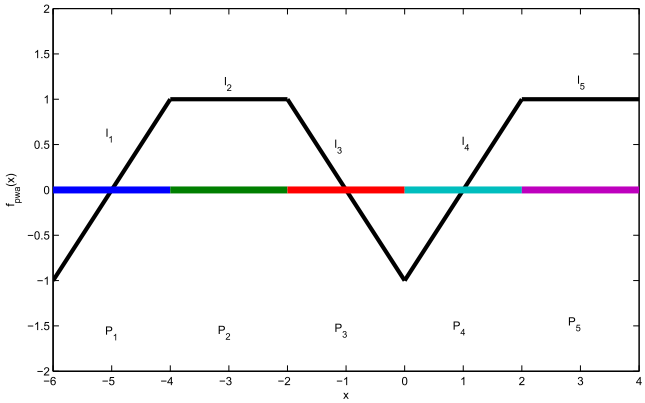
Definícia 2.1 Nech $\ell_i(x) = \alpha_i x + \beta_i, prex \in \mathcal{P}_i$ a $\ell_j(x) = \alpha_j x + \beta_j$, pre $x \in \mathcal{P}_j$, kde $\mathcal{P}_i$ aj $\mathcal{P}_j$ sú regióny toho istého polyhedrónu a $\ell_i(x)$ aj $\ell_j(x)$ sú jednotlivé lineárne funkcie zo spojitej PWA funkcie. Potom:

$$\psi_{ij} = \begin{cases} 1 & ak\ \ell_i(x) \geq \ell_j(x) \\ 0 & inak \end{cases} \tag{2.1}$$

$\mathbf{\Psi} = [\psi_{ij}]^{M \times M}$ je matica štruktúry.

Príklad: majme nasledujúcu PWA funkciu:

$$f(x) = \begin{cases} l_1 = x + 5 & \text{ak } x \in \mathcal{P}_1 = [-6, -4] \\ l_2 = 1 & \text{ak } x \in \mathcal{P}_2 = [-4, -2] \\ l_3 = -x - 1 & \text{ak } x \in \mathcal{P}_3 = [-2, 0] \\ l_4 = x - 1 & \text{ak } x \in \mathcal{P}_4 = [0, 2] \\ l_5 = 1 & \text{ak } x \in \mathcal{P}_5 = [2, 4] \end{cases} \tag{2.2}$$



Obr. 2.4: Graf PWA funkcie z príkladu

Matica štruktúry bude vyzerať:

$$\Psi = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{2.3}$$

Túto maticu vieme potom jednoducho prepísať do formy minima z maxím jednotlivých lineárnych funkcií. V nasledujúcej kapitole ukážem, že táto matica štruktúry je

ekvivalentná nasledujúcej min-max reprezentácií:

$$f(x) = min\left\{\ell_1, max\left\{\ell_4, \ell_5\right\}, max(\ell_3, \ell_4), max\left\{\ell_3, \ell_5\right\}\right\} \tag{2.4}$$

Wenn vo svojej práci[1] uvádza lemu2, ktorá zjednoduší porovnávanie či funkcia $\ell_i(x)$ je na celom intervale vačšia ako $\ell_j(x)$.

Definícia 2.2 Nech R_i a R_j sú dva n-dimenzionálne konvexné polytópy, kde $\ell_i(x), \ell_j(x)$ sú ich lokálne lineárne funkcie a $i, j \in \{1, ..., M\}$. Potom matica štruktúry $\mathbf{\Psi} = [\psi_{ij}]^{M\times M}$ može byt vypočítaná pomocou:

$$\Psi_{ij} = \begin{cases} 1 & ak \quad \ell_i(v_k) \geq \ell_j(v_k) \quad v_k \in \{1, \cdots, K\} \\ 0 & ak \quad \ell_i(v_k) < \ell_j(v_k) \quad v_k \in \{1, \cdots, K\} \end{cases} \tag{2.5}$$

kde v_k značí vrcholy polytópu R_i, R_j respektíve a K je celkový počet týchto vrcholov.

Definícia 2.3 (Min-max reprezentácia PWA funkcie,[1]):Majme maticu štruktúry $\Psi = \mathbb{R}^{N\times N}$, kde $\Psi_{ij} \in \{0, 1\}$ a nech $\ell_i(x) = \alpha_i x + \beta_i, i = 1, \cdots, N$. Potom funkcia $f : \mathbb{R}^n \to \mathbb{R}$:

$$f(x \mid \Psi) = \min_{1\leq i\leq N} \left\{ \max_{1\leq j\leq N, \Psi_{ij}=1} \{\ell_j(x)\} \right\} \tag{2.6}$$

sa nazýva mriežková reprezentácia PWA funkcie.

Matica štruktúry bude obsahovať i riadkov a j stĺpcov jednotiek a núl. Ľahko sa prepíše do min-max formy. Prejdeme jednotlivo všetkých i riadkov a v každom i-tom maxime budeme hľadať maximum zo všetkých lokálnych funkcií, kde v j-tom stĺpci sa nachádza 1.

Kapitola 3

Návrh riešenia

Vo svojej práci [1] navrhol Wenn dve lemy ako zjednodušiť mriežkovú reprezentáciu PWA funkcie. Jedná sa o zjednodušenie matice štruktúry po riadkoch, tiež nazývanú RVS lema(row vector simplification lema) a zjednodušenie po stĺpcoch, alebo CVS lema (column vector simplification lema). V nasledujúcich odsekoch ich uvediem. Vždy budem začínat definíciou, matlabovskou implementáciou a príkladom.

3.1 Implementácia mriežkovej reprezentácie v Matlabe

V prílohe prikladám algoritmus na vytvorenie mriežkovej reprezentácie z ľubovolnej spojitej PWA funkcie. Vstup do algoritmu sú matice F_i, G_i a P_n ktoré vytvoríme použitím multiparametrického toolboxu[2]. Matica F_i je matica koeficientov pri našich stavových premenných x a G_i je stlpcový vektor konštánt. Pn musíme pomocou funkcie extreme z MP toolboxu upravit na $\mathcal{V}$-reprezentáciu (reprezentáciu polytópu pomocou súradnic jeho vrcholov). Celý algoritmus sa skladá z jedného for cyklu v ktorom postupne prejde jednotlivé lokálne lineárne funkcie a druhého vnoreného cyklu v ktorom sú jednotlive funkcie navzájom porovnané a vyhodnotené na základe lemy2. Výstup z algoritmu je binárna matica štruktúry Ψ.

3.2 Implementácia metód zjedušenia mriežkovej reprezentácie

3.2.1 Zjednodušenie po riadkoch (RVS lema)

Definícia 3.1 (RVS lema, [1]): Nech $f(x \mid \psi) : D \subseteq \mathbb{R}^n \to \mathbb{R}$ je LPWA funkcia odvodená zo spojitej PWA funkcie s N lineárnymi úsekmi použitím definície matice štruktúry. Ak platí prvková nerovnost $\varphi_i - \varphi_j \leq 0$ pre všetky $i, j \in \{1, \cdots, N\}$ potom existuje zjednodušená matica štruktúry $\tilde{\Psi}^{(N-1)\times N}$, že platí:

$$f(x \mid \Psi) = f(x \mid \tilde{\Psi}) \tag{3.1}$$

kde $\Psi \in \mathbb{R}^{N\times N} = [\varphi_1, \cdots, \varphi_N]$ a $\tilde{\psi} \in \mathbb{R}^{(N-1)\times N} = [\varphi_1, \cdots, \varphi_{j-1}, \varphi_{j+1}, \cdots, \varphi_N]$.

Kedže jednoducho povedané RVS lema, hovorí, že ak sú rovnaké riadky, alebo v opakujúcom riadku je 1 na mieste kde v aktuálnom aktívnom riadku je 0, môžeme opakujúci riadok vylúčit bez toho aby sme zmenili význam lPWA funkcie. Preto tento algoritmus začne prechádzať maticu štruktúry a ak nájde riadok nad ktorým je aktívny riadok dominantný zmaže tento redundantný riadok a zvyšné posunie o 1 hore. Pokračuje nasledujúcim riadkom až kým nedôjde na koniec. Cyklus sa zastaví keď algoritmus prejde posledný riadok matice. Vstup aj výstupom z algoritmu je matica štrutúry.

Ak aplikujeme túto lemu na PWA funkciu z predošlého príkladu, dostaneme:

$$\Psi = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ - & - & - & - & - \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{3.2}$$

3.2.2 Zjednodušenie po stĺpcoch (CVS lema)

Najskôr zadefinujem komparatívne matice, ktoré v leme budem potrebovať.

Definícia 3.2 (komparatívna matica, [9]): Binárna matica $C(l_i) \in \mathbb{B}^{N\times N}$ pre jednu

i-tu afinnu funkciu $l_i(x)$spojitej PWA funkcie je definovaná ako:

$$C(l_i) = \begin{bmatrix} (\ell_i > \ell_1)_{\mathcal{P}_1} & (\ell_i > \ell_1)_{\mathcal{P}_2} & \cdots & (\ell_i > \ell_1)_{\mathcal{P}_N} \\ (\ell_i > \ell_2)_{\mathcal{P}_1} & (\ell_i > \ell_2)_{\mathcal{P}_2} & \cdots & (\ell_i > \ell_2)_{\mathcal{P}_N} \\ \vdots & \vdots & \ddots & \vdots \\ (\ell_i > \ell_N)_{\mathcal{P}_1} & (\ell_i > \ell_N)_{\mathcal{P}_2} & \cdots & (\ell_i > \ell_N)_{\mathcal{P}_N} \end{bmatrix} \tag{3.3}$$

kde

$$(\ell_i > \ell_j)_{\mathcal{P}_k} = \begin{cases} 1 & ak \quad min_{x \in int\mathcal{P}_k}(\ell_i(x) - \ell_j(x)) > 0 \\ 0 & inak \end{cases} \tag{3.4}$$

a množina komparatívnych matíc je:

$$C = \{C(\ell_1), C(\ell_2), \cdots, C(\ell_N)\} \tag{3.5}$$

Komparatívne matice pre PWA funkciu z príkladu:

$$\begin{aligned} C(\ell_1) &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}, C(\ell_2) = C(\ell_5) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \\ C(\ell_3) &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}, C(\ell_4) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{3.6}$$

Definícia 3.3 (CVS lema, [9]):Nech $f(x \mid \psi) : \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R}$je lPWA funkcia definovaná na N lineárnych regiónoch. Nech $l_{max} = l_m(x)$ je lokálna maximálna afinná funkcia $f(x \mid \psi)$podľa definície z nasledujúcej sekcie a C je množina komparatívnych matíc funkcie$f(x \mid \psi)$. Potom môžeme aplikovat nasledujúce pravidlá:

1. Pre hocijaké $i \in \{1, \cdots, N\}$,ak $\Psi_{im} = 1$ nastaviť $\Psi_{ij} = 0, \forall j \in \{1, \cdots, N\} \setminus m$

2. Pre hocijaké $i, j, k \in \{1, \cdots, N\}$ že $\Psi_{ij} = 1$ a $\Psi_{ik} = 1$ a $j \neq k$,

   - ak k-ty riadok $C(\ell_i)$ obsahuje iba samé nuly nastaviť $\Psi_{ij} = 0$
   - ak k-ty riadok $C(\ell_i)$ obsahuje iba samé jednotky nastaviť $\Psi_{ik} = 0$

3. Ak $\Psi_{ij} = 0, \forall j \in \{1, \cdots, N\}$ potom existuje zjednodušená matica štruktúry $\tilde{\Psi} \in \mathbb{B}^{(N-1)\times N}$, že platí

$$f(x \mid \Psi) = f(x \mid \tilde{\Psi}) \tag{3.7}$$

Vstupom do algoritmu je matica štruktúry, matice F_i, G_i a matica súradnic vrcholov polytópu Pn. Algoritmus na začiatku zavolá pomocnú funkciu na vytvorenie všetkých komparatívnych matíc. Funkcia vráti množinu komparatívnych matíc, ktorú následne algoritmus prehľadá aby našiel riadky v ktorých sú buď samé jednotky alebo nuly. Následne aplikuje column vector simplification lemu. Výstupom z algoritmu je zredukovaná matica štruktúry. Algoritmus má ešte jeden voliteľný vstup *options*, ktorý môže nadobúdať hodnoty 1 alebo defaultne 0. V prípade jeho zvolenia, algoritmus ešte pred prehľadávaním množiny komparatívnych matíc zavolá pomocnú funkciu na nájdenie maximálnej affinej (lineárnej) funkcie, ktorá síce predĺži čas ktorý algoritmus potrebuje na prebehnutie, ale dokáže ešte viac zjednodušit PWA funkciu. V experimente sa venujem aplikácií aj s hľadaním supervektora, aj bez neho.

3.2.3 Maximálna afínna funkcia (MAF)

Definícia 3.4 (Maximálna afínna funkcia,[9]): MAF zo spojitej PWA funkcie $\ell_{max} = \alpha_m^T x + \beta_m$je taká funkcia, že platí:

- $\ell_{max} \geq \alpha_j^T x + \beta_j, \forall j \in \{1, \cdots, N\}, x \in \mathcal{P}_j$
- maximalizuje k; pre $k = 0$; ak $\ell_{max} \geq \alpha_j^T x + \beta_j, \forall j \in \mathbb{N}_1^N$ potom $k = k + 1$

Inými slovami MAF l_{max} ohraničuje zhora všetky ostatné funkcie na všetkých regiónoch. Pri viacerých vhodných kandidátoch hľadáme takú funkciu, ktorá je rovná najväčšiemu počtu funkcií z PWA systému. V prípade, že nájdeme takúto funkciu, môžeme aplikovat prvý predpoklad z CVS lemi a tým zjednodušit min-max reprezentáciu.

Ak využijeme MAF pri CVS leme na PWA funkciu z príkladu dostaneme oveľa jednoduchšiu min-max reprezentáciu:

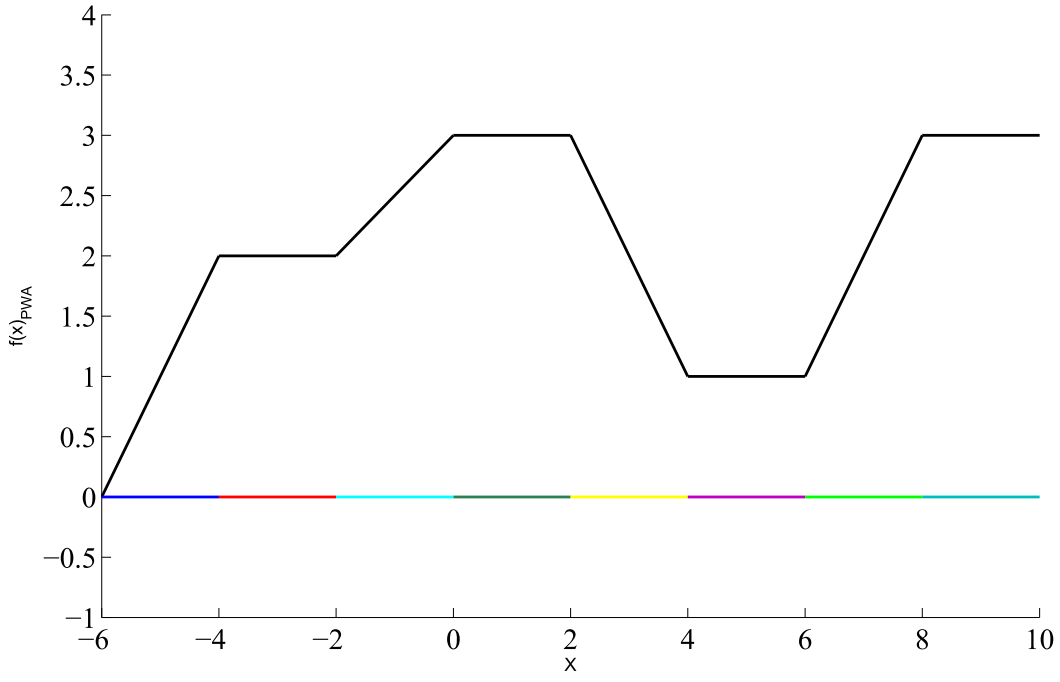
$$f(x) = min\{\ell_1, \ell_2, max(\ell_3, \ell_4)\} \tag{3.8}$$

Tabuľka 3.1: Tabuľka výsledkov experimentov:

PWA funkcia	bez aplikácie lém	RVS lema	CVS lema	celkové zjednodušenie (%)
PWA(prikl)	12	10	7(4)	42(67)
PWA(exp.1)	33	22	11	67

3.3 Experiment - príklad

Experiment som urobil na 1 rozmernej PWA funkcii definovanej na 8 polytopických regiónoch. Na PWA funkciu som aplikoval algoritmus na vytvorenie matice štruktúry. Potom som ju zjednodušil pomocou RVS lemy. Výsledky pred a po tomto zjednodušení sú uvedné v prvom a druhom stĺpci tabuľky. Na túto zjednodušenú maticu som potom aplikoval CVS lemu s MAF aj bez neho. V prípade, že MAF priniesla želaný výsledok je uvedený v zátvorke.



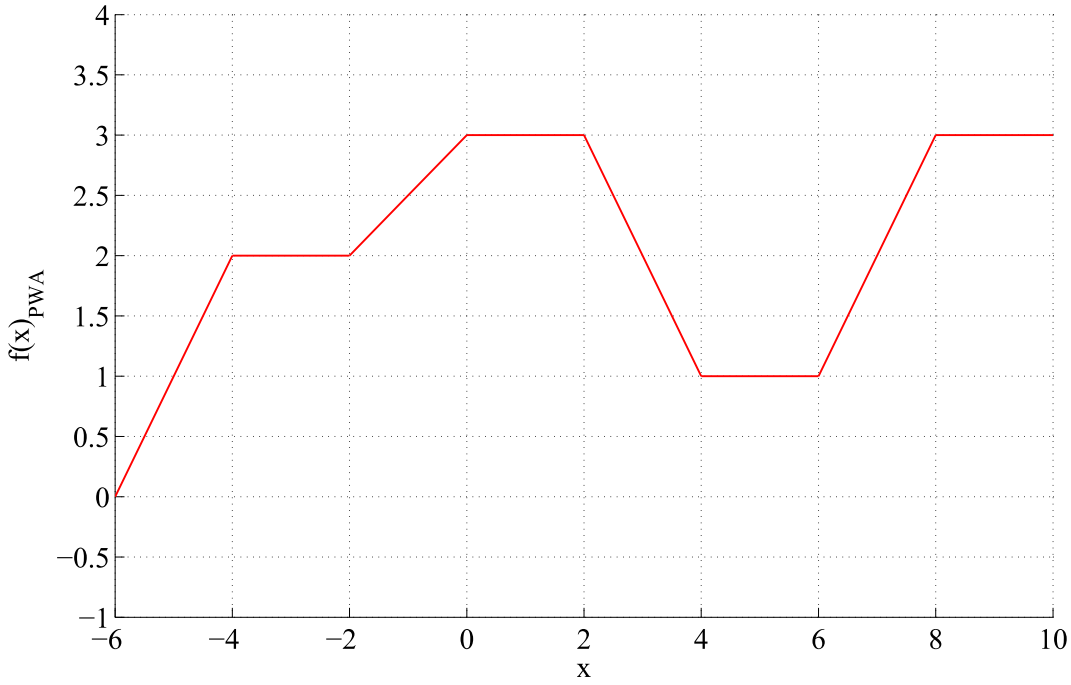
Obr. 3.5: PWA funkcia z prvého experimentu

Výsledná min-max reprezentácia PWA funkcie z prvého experimentu:

$$f(x)_{PWA} = \min\left\{\max\left(\ell_1, \ell_2\right), \max\left(\ell_2, \ell_3\right), \max\left(\ell_7, \ell_8\right), \max\left(\ell_5, \ell_6, \ell_7\right), \max\left(\ell_5, \ell_8\right)\right\} \tag{3.9}$$

3.4 Zhodnotenie

Obrázok 3.6 ukazuje, že algoritmi fungujú správne. Červenou farbou som nechal vykresliť novú zjednodušenú PWA funkciu a modrou pôvodnú PWA. Červená zjednodušená, ktorá bola aplikovaná ako druhá úplne prekryla povôdnú modrú. Vo svojej práci som ukázal, že mriežková reprezentácia PWA funkcie má veľký význam v explicitnom prediktívnom riadení. Nielen kvôli tomu, že v pamäti výpočtovej jednotky zaberá menej miesta, ale aj vďaka jednoduchosti a efektívnosti zjednodušovania jej reprezentácie. V matlabe boli implementované algoritmy pre prácu s mriežkovou reprezentáciou PWA funkcie a jej optimalizáciu. Budúca práca bude spočívať v rozšírení matlabovských skriptov o funkcie prevodu medzi jednotlivými formami PWA funkcie (LPWA, PPWA, min-max).



Obr. 3.6: Modrá originálna PWA funkcia je prekrytá červenom zjednodušenou min-max PWA.

Kapitola 4

Prílohy

Na priloženom CD sa nachádzajú všetky algoritmy, ktoré som v práci použil. Tu uvediem, len časť ktorá sa zobrazí v Matlabe po zadaní príkazu help.
function struct_mat = structure_matrix(Fi, Gi, Pn)
Algorithm creates structure matrix from matrices of continuous PWA function
Inputs
Fi = matrix of parameters
Gi = matrix of constants
Pn = matrix of polytope's vertexes
Output
structure matrix

function struct_mat = row_vector_simplification(struct_mat)
Algorithm simplifies LPWA by removing redundant rows
Input
structure matrix
Output
structure matrix

function [struct_mat, C] = column_vector_simplification(struct_mat, Fi, Gi, Pn, varargin)
Algorithm simplifies LPWA function by removing redundant columns. If you would like to search for maximal affine function, enter 1 as optional parameter after Pn matrix.
===INPUTS===

struct_mat = structure matrix

Fi, Gi = matrices of parameters from MPT toolbox

Pn = matrix of politopic vertexes

===OUTPUTS===

structure matrix and vector of comparative matrices

TestBc

Jednorozmerna spojita PWA funkcia na testovanie algoritmov k Bc.praci

Count_fnc(sm)

Spocita pocet afinnych funkcii

Literatúra

[1] C. Wen, X. Ma, and B. E. Ydstie, "Analytical expression of explicit mpc solution via lattice piecewise-affine function," *Automatica*, vol. 45, pp. 910–917, 2009.

[2] K. M., G. P., B. M., and C. F.J., "Multi-parametric toolbox (mpt)," *http://control.ee.ethz.ch/ mpt/*, 2006.

[3] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[4] E. W. Weisstein, "Mathworld wolfram web resource," *http://mathworld.wolfram.com/*.

[5] B. Grunbaum, *Convex Polytopes*. Graduate Texts in Mathematics, Interscience Publishers, 1967.

[6] M. Kvasnica, *Real-Time Model Predictive Control Via Multi-Parametric Programming: Theory and Tools*. VDM Publishing, 2009.

[7] T. Geyer, F. D. Torrisi, and M. Morari, "Optimal complexity reduction of polyhedral piecewise affine systems," *Automatica*, vol. 44, pp. 1728–1740, 2008.

[8] M. Kvasnica and M. Fikar, "Clipping-based complexity reduction in explicit mpc," *IEEE Trans. Automat. Contr.*, vol. 57, pp. 1878–1883, 2012.

[9] M. Kvasnica, "Research project: Construction of bounded piecewise affine functions," *Research project*, 2012.