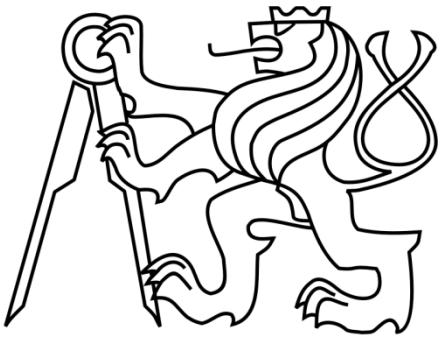
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA STROJNÍ

ÚSTAV MECHANIKY, BIOMECHANIKY A MECHATRONIKY

Odbor mechaniky a mechatroniky



Diplomová práce

Zlepšení manipulovatelnosti tensegritických struktur využitím sítě lokálních lineárních modelů

Praha, 2021 **Bc. Martin Jílek**

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Jílek** Jméno: **Martin** Osobní číslo: **466681**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav mechaniky, biomechaniky a mechatroniky**

Studijní program: **Aplikované vědy ve strojním inženýrství**

Specializace: **Mechatronika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Zlepšení manipulovatelnosti tensegritických struktur využitím sítě lokálních lineárních modelů

Název diplomové práce anglicky:

Pokyny pro vypracování:

1) Seznamte se s problematikou tensegritických struktur.
2) Seznamte se s identifikačními postupy pomocí sítě lokálních lineárních modelů.
3) Vytvořte simulační model rekonfigurovatelného rovinného manipulátoru a proveďte optimalizaci jeho manipulovatelnosti.
4) Použijte postup optimalizace manipulovatelnosti na tensegritickou strukturu.

Seznam doporučené literatury:

[1] Skopec, T.: Kalibrace paralelních mechanismů s adaptivní složitostí modelu, Ph.D. disertační práce, FS ČVUT v Praze, 2012.
[2] Nelles, O.: Nonlinear System Identification with Local Linear Neuro-Fuzzy Models : Ph.D. Thesis, Technische Universitat Darmstadt, 1998.
[3] Skelton, R.E., Adhikari, R., Pinaud, J.-p., Chan, W.: An Introduction to the Mechanics of Tensegrity Structures, in Proceedings of the 40th IEEE Conference on Decision and Control, Orlando, 2001.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Petr Beneš, Ph.D., odbor mechaniky a mechatroniky FS

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **28.04.2021** Termín odevzdání diplomové práce: **13.08.2021**

Platnost zadání diplomové práce: _____

Ing. Petr Beneš, Ph.D.	doc. Ing. Miroslav Španiel, CSc.	prof. Ing. Michael Valášek, DrSc.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

Datum převzetí zadání Podpis studenta

Anotační list

Jméno autora:	Martin Jílek
Název diplomové práce:	Zlepšení manipulovatelnosti tensegritických struktur využitím sítě lokálních lineárních modelů
Anglický název:	Dexterity improvement of tensegrity structures using local linear models
Akademický rok:	2020/2021
Obor studia:	Mechatronika
Ústav/odbor:	Ústav mechaniky, biomechaniky a mechatroniky Odbor Mechaniky a mechatroniky
Vedoucí diplomové práce:	Ing. Petr Beneš, Ph. D.
Bibliografické údaje:	Počet stran: 63 Počet obrázků: 59 Počet příloh: 1
Klíčová slova:	manipulovatelnost, tensegrita, neuro-fuzzy, lolimot
Keywords:	dexterity, tensegrity, neuro-fuzzy, lolimot
Anotace:	V této diplomové práci se zabývám problematikou zlepšení manipulovatelnosti tensegritických struktur. Součástí práce je rešerše ohledně historie a aktuálního stavu vývoje tensegritických struktur. Dále práce obsahuje základní poznatky ohledně neuronových sítí a fuzzy logiky. Podrobně je zde popsán identifikační algoritmus LOLIMOT (Local linear model tree) a zároveň je implementován v prostředí MATLAB. Na jednoduchém pětikloubovém mechanismu je aplikován postup zlepšení manipulovatelnosti v definovaném pracovním prostoru a je řešena jeho kinematika. Na tento mechanismus je následně natrénován algoritmus LOLIMOT. Obdobně je proces zlepšení manipulovatelnosti řešen na 3-vrstvé rovinné tensegritické struktuře.

Abstract:

This master thesis discovers topic of improving dexterity of tensegrity structures. It includes comprehensive research of history and current advancement in field of tensegrity structures research. Furthermore, it consists of basic ideas summary of neural networks and fuzzy logic. Identification algorithm LOLIMOT (Local linar model tree) is covered in detail and implemented in MATLAB. This approach to improvement of dexterity is applied on 5 - joint mechanism in pre-defined workspace. Also kinematics of this mechanism is explored. LOLIMOT neuro-fuzzy network is trained on this mechanism. In addition, process of improving dexterity is applied to 3 - stage planar tensegrity structure.

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.

V Praze, dne …….. ……

Podpis

Poděkování

Rád bych poděkoval svému vedoucímu diplomové práce, Ing. Petru Benešovi, Ph. D, za pomoc při zpracování. Dále bych chtěl poděkovat své rodině, která mě po celou dobu studia plně podporovala.

Obsah

Seznam obrázků

1. Úvod

Jedním z důležitých úkolů při navrhování robotických manipulátorů je zajištění co možná nejoptimálnějšího využití silového působení pohonů na koncový efektor daného manipulátoru. Tímto ukazatelem kvality návrhu a poměrně jednoduše kvantifikovatelným parametrem je manipulovatelnost, jinak označovaná také jako dexterita. Jedná se v podstatě o zobecnění pojmu „převod“ mezi rychlostí pohonů a koncového efektoru.

Kromě dnes již naprosto běžných sériových struktur manipulátorů a o něco novějších paralelních struktur se v poslední době klade důraz na vývoj i tzv. „tensegritických“ struktur. Jedná se o hybridní struktury skládající se z lan s proměnnou délkou a tuhých tyčí. Změnou délky lan je uskutečňován pohyb. Z logiky věci vyplývá, že tyto struktury budou mít výrazně více pohonů, než stupňů volnosti. Zřejmě největším současným problémem pro vědce zabývající se touto problematikou je návrh řízení. Navíc je z hlediska kinematického i dynamického popisu velice náročné tyto objekty modelovat. Z historického hlediska je nejprve vytvořen fyzikální model požadované struktury a na něj je použit zákon řízení, který se snaží přivést systém do požadovaného stavu. Propojit návrh struktury a řízení je obtížné i pro špičkové výzkumné týmy. Jako příklad mohu uvést projekt „smart structures“, financovaný agenturou DARPA, snažící se řídit letecká křídla změnou tvaru. Existující křídlo bojového letounu F-16 bylo vybaveno nikl-titanovým aktuátorem pro aplikaci velkého kroutícího momentu na křídlo s cílem jeho prohnutí až o 20°. Řídící systém a aktuátor zkroutily křídlo pouze o 7°, jelikož bylo vyžadováno příliš mnoho energie pro dosažení požadovaného zkrutu od své rovnovážné polohy. Na rozdíl od tohoto přístupu, idea tensegritních struktur je změna samotné rovnovážné polohy k dosažení požadovaného tvaru, takže není vyžadována žádná další energie pro zachování takovéto konfigurace. Spojení návrhu fyzikální struktury i zákona řízení by vyžadovalo menší energetické úsilí od aktuátorů k dosažení stejných výsledků. [1]

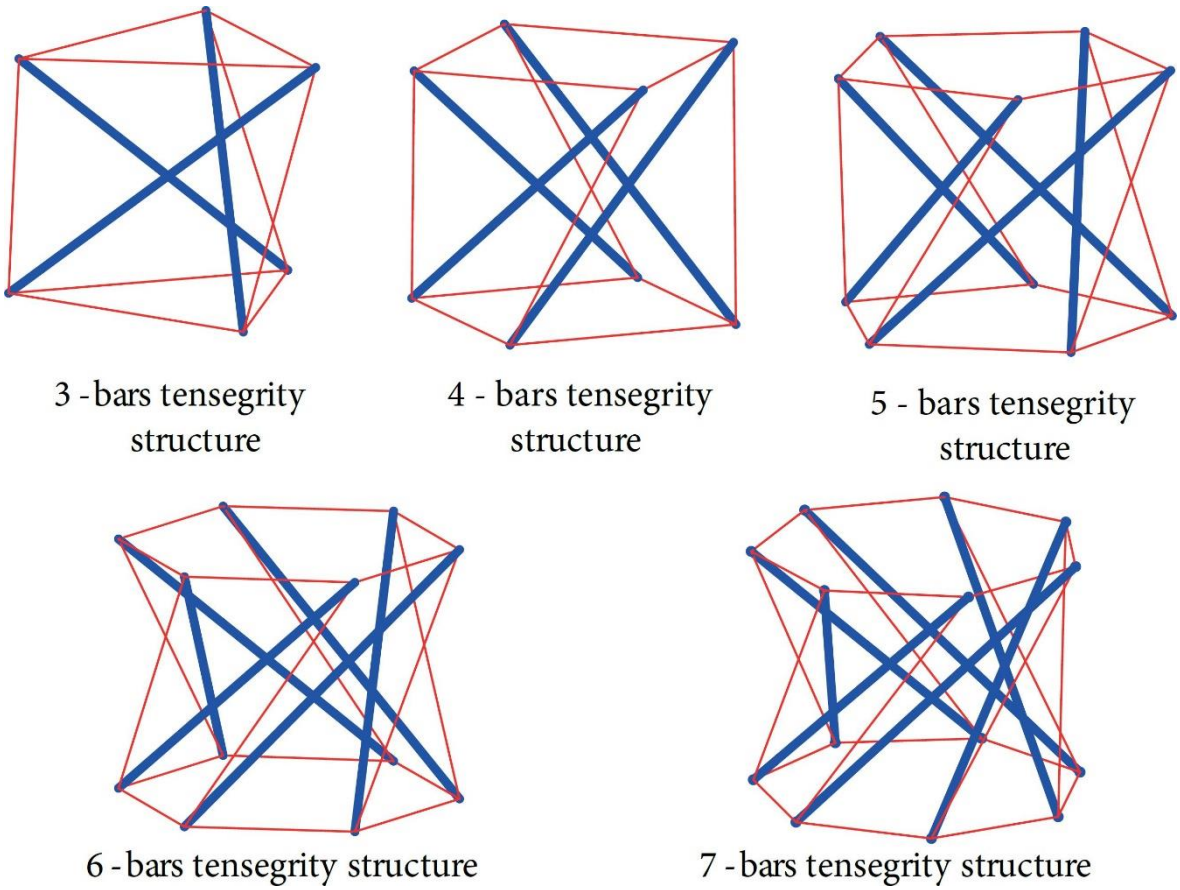
Sítí lokálních lineárních modelů rozumíme neuro-fuzzy síť, která vytváří model určitého problému na základě měření či simulačních dat. Spočívá v postupném dělení složitého nelineárního modelu na mnoho menších, lineárních modelů, jejichž platnost je popsána fuzzy logikou. Existuje tedy mezi nimi plynulý přechod, nejedná se o binární úlohu platí/neplatí, nýbrž prolínání více modelů s určitou procentuální platností. Tato práce se zaměřuje zejména na algoritmus LOLIMOT (Local Linear Model Tree), který je implementován pro identifikaci rovinného manipulátoru i tensegritické struktury.

2. Cíle

Mezi cíle patří seznámení se s problematikou tensegritických struktur a to jak v rovině, tak i v prostoru. Dále si tato práce klade za cíl uvedení do problematiky identifikace systémů, s důrazem na algoritmus LOLIMOT. Tento algoritmus bude použit pro vytvoření modelu jednoduchého rovinného mechanismu a optimalizaci jeho manipulovatelnosti při zavedení základního principu tensegritických struktur, čímž je nadbytečnost pohonů. Následně bude síť lokálních lineárních modelů použita při optimalizaci manipulovatelnosti rovinného tensegritického manipulátoru.

3. Seznámení s problematikou tensegritických struktur

Tensegritou rozumíme strukturu, která má určitý počet nesouvislých komponent namáhaných v tlaku, s další souvislou skupinou komponent vně namáhanou v tahu. Důležité je třeba zmínit, že takto popsaný systém je ve stabilní rovnovážné poloze. Tyto dvě skupiny se nazývají tyče a lana. Jak je již z názvu zřejmé, tyče jsou namáhané v tlaku a lana v tahu. Na Obr. 1 Základní typy prostorové tensegritické jednotky jsou popsány základní typy prostorové tensegrity s 3 až 7 tyčemi. [2]



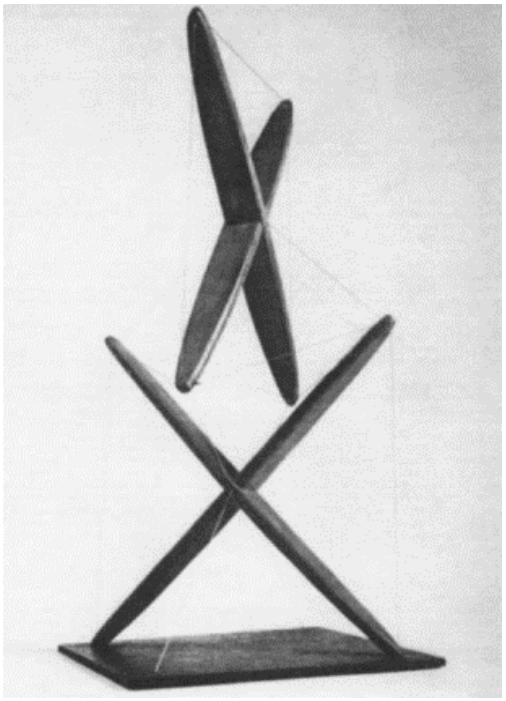
Obr. 1 Základní typy prostorové tensegritické jednotky [3]

Existuje vícero druhů tensegritických struktur (např. rozvinutelné tensegrity [4]), ovšem vzhledem k tomu, že ve výsledku se zabýváme manipulátory, omezíme se ve výkladu pouze na struktury složené z válcových jednotek.

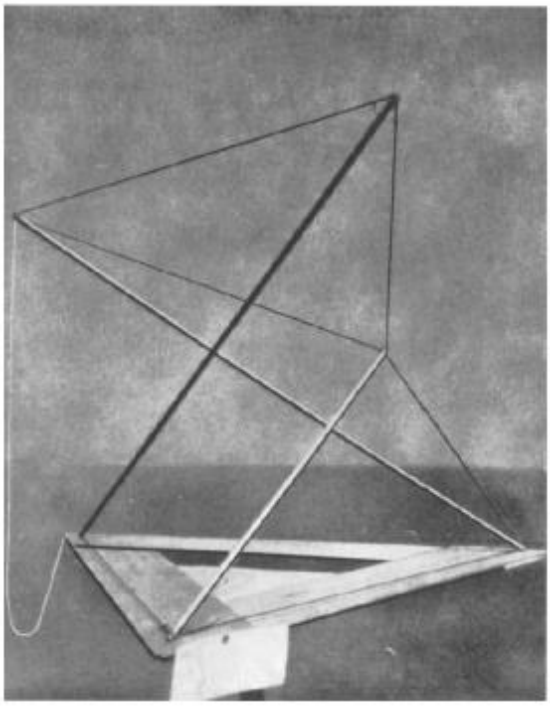
V současné době je výzkum tensegritických struktur omezen na analýzu mechanických vlastností a kinematického chování. Ve většině studií je matematický a mechanický model struktury odvozen na základě zobecněných souřadnic uzlů základní jednotky. [3]

3.1. Historie pojmu a předchozí výzkum

Historie tensegritních struktur sahá do roku 1920, kdy byla představena první takováto struktura svého druhu. Jejím tvůrcem byl ruský umělec Ioganson. Jak je vidět z Obr. 2, tato konstrukce zatím nemá tvar obálky válce, chybí jí totiž horní



Obr. 3 X-shape tensegrita [1]



Obr. 2 proto-tensegritní struktura [12]

podstava. Je tvořena 3 tyčemi a 8 lany, z nichž 7 lan je předepjatých a poslední slouží ke změně konfigurace struktury tensegritické jednotky.

Problémem bylo, že při posuvu z rovnovážné polohy struktura nedržela tvar, takže se neřadí mezi tensegritické struktury, nýbrž tzv. proto-tensegritní struktury. [2]

Pokud bychom se bavili o pravých tensegritních strukturách, museli bychom se ohlédnout do roku 1948. V tomto roce byl sestaven první fyzikální model, ten se sestával ze 4 tyčí, které byly uzpůsobeny do tvaru písmena „X“ a 14 lan. Tento demonstrátor je na Obr. 3

3.2. Matematický model jednovrstvé tensegrity

Jak již bylo zmíněno, tensegritická struktura se skládá z tyčí a lan. Jejich průsečíky tvoří uzly. Matematický model tedy může být sestaven z uzlů a tím pádem lze získat vztah mezi rozměry jednotlivých komponent a geometrickými parametry. Nejjednodušší tensegritickou strukturou je tzv. „three-bar tensegrity“, neboli tensegrita složená ze tří tyčí. Každá jednovrstvá tensegritická struktura odpovídá válcové obálce o výšce h a poloměru r. Pokud je počet tyčí p, pak se struktura skládá z $2p$ uzlů a $3p$ lan. Zároveň jsou všechny uzly rovnoměrně rozloženy na horní a dolní podstavě válcové obálky. Uzly na horní podstavě mohou být spojeny s jakýmkoliv uzlem v dolní podstavě přes jednu a tu samou tyč. Takováto struktura může zaujmout

p-1 konfigurací. Jak plyne z Obr. 1 Základní typy prostorové tensegritické jednotky, střed dolní podstavy je brán jako počátek kartézského souřadnicového systému. Úhel i-tého uzlu na dolní podstavě je

$$\alpha_{di} = \frac{2\pi}{p} \cdot (i - 1) \tag{3.1}$$

Kde i ∈ [1, 2, …p]. Souřadnice uzlů na dolní podstavě jsou

$$n_{di} = [R \cos \alpha_{di}\,, R \sin \alpha_{di}\,, 0]^T \tag{3.2}$$

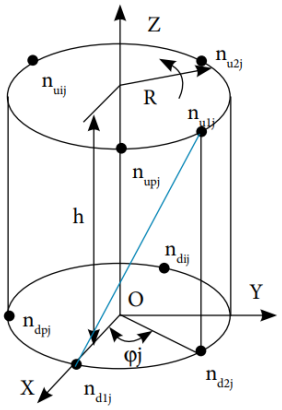
A úhel i-tého uzlu k horní podstavě je

$$\alpha_{uij} = \varphi_j + \frac{2\pi}{p} \cdot (i - 1) \tag{3.3}$$

Kde j ∈ [1, p-1] a udává index konfigurace a φ_j je torzní úhel j-té konfigurace jednovrstvé tensegritické struktury.

$$\varphi_j = \frac{\pi}{2} + \frac{(p - j)\pi}{p} \tag{3.4}$$

Tyto parametry jsou vyznačeny na Obr. 4



Obr. 4 Geometrický popis

Souřadnice uzlů na horní podstavě jsou

$$n_{uij} = \left[R \cos \alpha_{uij}\,, R \sin \alpha_{uij}\,, h\right]^T \tag{3.5}$$

Matici N_j uzlu můžeme psát jako:

$$\boldsymbol{N}_j = [n_{d1}, n_{d2}, \dots, n_{dp}, n_{u1j}, n_{u2j}, \dots n_{upj}]_{3\times 2p} \tag{3.6}$$

Můžeme definovat směrový vektor i-té tyče

$$\vec{b}_{ij} = n_{uij} - n_{di} \tag{3.7}$$

Tyto vektory pro všechny tyče můžeme sdružit do matice soustavy následovně

$$\boldsymbol{B}_j = [\vec{b}_{1j}\ \vec{b}_{2j} \dots \vec{b}_{pj}]_{3\times p} = \boldsymbol{N}_j \boldsymbol{C}_{Bj}^T = \boldsymbol{N}_j \begin{bmatrix} -\boldsymbol{I} \\ \boldsymbol{I} \end{bmatrix}_{2p\times p} \tag{3.8}$$

Kde $\boldsymbol{C}_{Bj}^T$ nazveme maticí propojenosti tyčí a $\boldsymbol{I}$ je jednotková matice p-tého řádu.

$$\boldsymbol{C}_{Bj}^T = \begin{bmatrix} -\boldsymbol{I} \\ \boldsymbol{I} \end{bmatrix}_{2p\times p} \tag{3.9}$$

Dále definujme matici směrových vektorů lan

$$\boldsymbol{S}_j = \left[\vec{s}_{h1}, \dots, \vec{s}_{hi}, \dots, \vec{s}_{h2p}, \vec{s}_{v1j}, \dots \vec{s}_{vkj}, \dots, \vec{s}_{vpj}\right]_{3\times 3p} = \boldsymbol{N}_j \boldsymbol{C}_{sj}^T = \boldsymbol{N}_j [\boldsymbol{C}_{sh}^T \quad \boldsymbol{C}_{svj}^T]_{2p\times 3p} \tag{3.10}$$

Kde $\vec{s}_{hi}$ je směrový vektor horizontálního lana pro i ∈ [1, 2, …, p] a $\vec{s}_{vkj}$ je směrový vektor šikmého lana j-té konfigurace pro k ∈ [1, 2, …, p-1]. $\boldsymbol{N}_j$ je matice uzlovýc souřadnic j-té konfigurace. $\boldsymbol{C}_{Bj}^T$ je matice propojenosti lan, která udává vztah mezi uzly a maticí směrových vektorů lan $\boldsymbol{S}_j$.

Dále pak $\boldsymbol{C}_{sh}^T$, která reprezentuje matici propojenosti horizontálních lan a $\boldsymbol{C}_{svj}^T$, která naopak zastupuje matici propojenosti šikmých lan.

$$\boldsymbol{C}_{sh}^T = \begin{bmatrix} \boldsymbol{E} & 0 \\ 0 & \boldsymbol{E} \end{bmatrix}_{2p\times 2p} \tag{3.11}$$

Kde matice $\boldsymbol{E}$ má následující tvar:

$$\boldsymbol{E} = \begin{bmatrix} -1 & 0 & 0 & \cdots & 0 & 1 \\ 1 & -1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -1 & \cdots & 0 & 0 \\ \vdots & 0 & 1 & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & -1 & 0 \\ 0 & 0 & 0 & \cdots & 1 & -1 \end{bmatrix}_{p \times p} \tag{3.12}$$

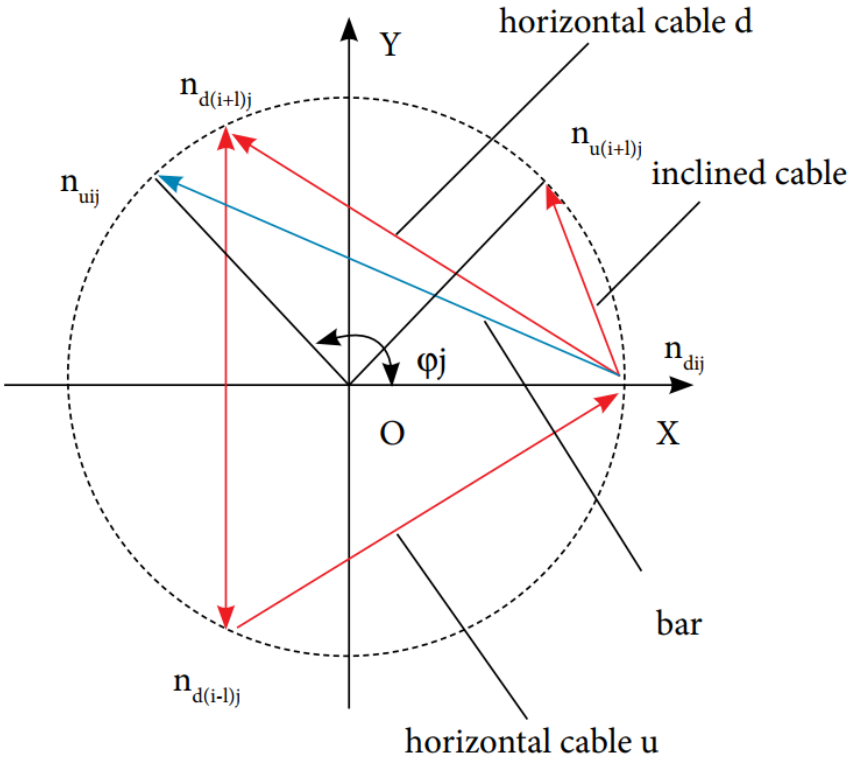
Matice $\boldsymbol{C}_{svj}^{T}$ má poté tvar

$$\boldsymbol{C}_{svj}^{T} = \begin{bmatrix} -\boldsymbol{I} \\ \boldsymbol{H} \end{bmatrix}_{2p \times p} \tag{3.13}$$

Matice $\boldsymbol{H}$ je složena z jednotkových matic následujícího tvaru:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I}_{j \times j} \\ \boldsymbol{I}_{(p-j)\times(p-j)} & \boldsymbol{0} \end{bmatrix}_{p \times p} \tag{3.14}$$

Na základě výše zmíněných vztahů lze odvodit vztah mezi působícími silami a hustotou komponent struktury. Řešení je následující.



Obr. 5 Geometrické uspořádání komponent [3]

Na Obr. 5 modrá šipka reprezentuje tyč a červené šipky reprezentují lana. Každý uzel je spojen se dvěma horizontálními lany a jedním šikmým lanem. Na základě toho je

jednovrstvá tensegritická struktura symetrická a napětí ve všech uzlech je stejné. Pokud je struktura ve vlastní rovnovážné poloze, každý uzel je vyvažován vnitřními silami komponent (lan a tyčí). Vztah síla-hustota (Force-density) udává přepočet mezi vnitřními silami a směrovými vektory komponent („component vector“). Pokud délka komponenty je l a vnitřní síla je F, pak vztah síla-hustota komponenty je F/l. Tento vztah pro tyč se značí λ, šikmého lana r_v, horizontálního lana r_{hd} respektive $r_{hu,}$ podle umístění v horní nebo dolní podstavě válcové obálky struktury.

Rovnice silové rovnováhy v souřadnicích x, y a z pro uzly n_{di} vypadají následovně:

$$\begin{bmatrix} (x_{dij} - x_{uij}) & (x_{u(i+j)j} - x_{dij}) & (x_{d(i+1)j} - x_{dij}) & (x_{d(i-1)j} - x_{dij}) \\ (y_{dij} - y_{uij}) & (y_{u(i+j)j} - y_{dij}) & (y_{d(i+1)j} - y_{dij}) & (y_{d(i-1)j} - y_{dij}) \\ (z_{dij} - z_{uij}) & (z_{u(i+j)j} - z_{dij}) & (z_{d(i+1)j} - z_{dij}) & (z_{d(i-1)j} - z_{dij}) \end{bmatrix} \begin{bmatrix} \lambda \\ r_v \\ r_{hd} \\ r_{hu} \end{bmatrix} = 0 \tag{3.15}$$

Rovnice platí pro $i \in [1, p]$ a $j \in [1, p-1]$. V případě, že je struktura zatěžována vnější silou P, platí rovnice silové rovnováhy v následujícím tvaru:

$$\begin{bmatrix} (x_{dij} - x_{uij}) & (x_{u(i+j)j} - x_{dij}) & (x_{d(i+1)j} - x_{dij}) & (x_{d(i-1)j} - x_{dij}) \\ (y_{dij} - y_{uij}) & (y_{u(i+j)j} - y_{dij}) & (y_{d(i+1)j} - y_{dij}) & (y_{d(i-1)j} - y_{dij}) \\ (z_{dij} - z_{uij}) & (z_{u(i+j)j} - z_{dij}) & (z_{d(i+1)j} - z_{dij}) & (z_{d(i-1)j} - z_{dij}) \end{bmatrix} \begin{bmatrix} \lambda \\ r_v \\ r_{hd} \\ r_{hu} \end{bmatrix} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{3.16}$$

Kde složky vektoru síly P jsou průměty do souřadnicových os. Jedná se o základní úlohu návrhu tensegritické struktury, tzv. form-finding [1]. Jde nám o nalezení takové kombinace sil ve všech elementech, aby bylo dosaženo rovnovážné předepjaté konfigurace. Vystupuje zde matice koeficientů $\boldsymbol{A}_{dij}$ v uzlu n_{dij}

$$\boldsymbol{A}_{dij} = \begin{bmatrix} (x_{dij} - x_{uij}) & (x_{u(i+j)j} - x_{dij}) & (x_{d(i+1)j} - x_{dij}) & (x_{d(i-1)j} - x_{dij}) \\ (y_{dij} - y_{uij}) & (y_{u(i+j)j} - y_{dij}) & (y_{d(i+1)j} - y_{dij}) & (y_{d(i-1)j} - y_{dij}) \\ (z_{dij} - z_{uij}) & (z_{u(i+j)j} - z_{dij}) & (z_{d(i+1)j} - z_{dij}) & (z_{d(i-1)j} - z_{dij}) \end{bmatrix} \tag{3.17}$$

Kde jednotlivé prvky jsou souřadnice uzlů v dolní a horní podstavě jednovrstvé tensegritické struktury. Obecná matice rovnováhy $\mathbf{A}$ jednovrstvé struktury může být odvozena ve tvaru

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \tag{3.18}$$

Kde

$$a_{11} = (R\cos\left(\frac{2\pi}{p}\cdot(i-1)\right) - R\cos(\varphi_j + \frac{2\pi}{p}\cdot(i+j-1)) \tag{3.19}$$

$$a_{12} = R\cos\left(\varphi_j + \frac{2\pi}{p}\cdot(i+j-1)\right) - R\cos\left(\frac{2\pi}{p}\cdot(i-1)\right) \tag{3.20}$$

$$a_{13} = R\cos\left(\frac{2\pi i}{p}\right) - R\cos\left(\frac{2\pi}{p}\cdot(i-1)\right) \tag{3.21}$$

$$a_{14} = R\cos\left(\frac{2\pi(i-2)}{p}\right) - R\cos\left(\frac{2\pi}{p}\cdot(i-1)\right) \tag{3.22}$$

$$a_{21} = (R\sin\left(\frac{2\pi}{p}\cdot(i-1)\right) - R\sin(\varphi_j + \frac{2\pi}{p}\cdot(i+j-1)) \tag{3.23}$$

$$a_{22} = (R\sin(\varphi_j + \frac{2\pi}{p}\cdot(i+j-1)) - R\sin\left(\frac{2\pi}{p}\cdot(i-1)\right) \tag{3.24}$$

$$a_{23} = R\sin\left(\frac{2\pi i}{p}\right) - R\sin\left(\frac{2\pi}{p}\cdot(i-1)\right) \tag{3.25}$$

$$a_{24} = R\sin\left(\frac{2\pi(i-2)}{p}\right) - R\sin\left(\frac{2\pi}{p}\cdot(i-1)\right) \tag{3.26}$$

$$a_{31} = -h \tag{3.27}$$

$$a_{32} = h \tag{3.28}$$

$$a_{33} = 0 \tag{3.29}$$

$$a_{34} = 0 \tag{3.30}$$

Obecně závisí prvky matice na geometrické konfiguraci tensegrity. Je patrné, že už pro mírně složitější konfigurace tensegritické jednotky jsou vztahy nutné k určení prvků matice $\boldsymbol{A}$ náročně získatelné. Po nalezení statické rovnovážné polohy se může přikročit ke statické analýze.

Pokud obecnou matici rovnováhy **A** dosadíme do rovnice (3.15), dostaneme soustavu rovnic řešící problém force-density bez vnějšího zatížení jako:

$$a_{11}\lambda + a_{12}r_v + a_{13}r_{hd} + a_{14}r_{hu} = 0 \tag{3.31}$$

$$a_{21}\lambda + a_{22}r_v + a_{23}r_{hd} + a_{24}r_{hu} = 0 \tag{3.32}$$

$$a_{31}\lambda + a_{32}r_v + a_{33}r_{hd} + a_{34}r_{hu} = 0 \tag{3.33}$$

Vzhledem k tomu, že struktura je symetrická a síly a délky obou horizontálních lan spojené jedním uzlem jsou stejné, platí:

$$r_{hd} = r_{hu} \tag{3.34}$$

Dosazením (3.34) do soustavy rovnic (3.31), (3.32) a (3.33) můžeme psát:

$$r_v = \lambda \tag{3.35}$$

$$r_{hd} = r_{hu} = \frac{\cos\left(\frac{2\pi}{p}\cdot(i-1)\right) - \cos(\varphi_j + \frac{2\pi}{p}\cdot(i+j-1))}{\cos\left(\frac{2\pi i}{p}\right) + \cos\left(\frac{2\pi(i-2)}{p}\right) - 2\cos\left(\frac{2\pi}{p}\cdot(i-1)\right)}\lambda \tag{3.36}$$

$$r_{hd} = r_{hu} = \frac{\cos\left(\frac{\pi(p-j)}{p} + \frac{2\pi(i-1)}{p}\right) - \cos\left(\frac{\pi(p-j)}{p} + \frac{2\pi(i+j-1)}{p}\right)}{\sin\left(\frac{2\pi i}{p}\right) + \sin\left(\frac{2\pi(i-2)}{p}\right) - 2\sin\left(\frac{2\pi}{p}\cdot(i-1)\right)}\lambda \tag{3.37}$$

Rovnice (3.35), (3.36) a (3.37) ukazují, že vztah mezi silami a hustotou komponent jednovrstvé tensegritické struktury závisí pouze na torzním úhlu a nemá žádný vztah s velikostí členu. Torzní úhel je určen počtem tyčí p a konfigurací struktury j. Splněním těchto rovnic dosáhne struktura rovnovážné polohy. Rovnice (3.36) a (3.37) ukazují poměr síla-hustota horizontálních lan. Pokud je posuzovaný uzel na ose x, pak je jmenovatel rovnice (3.37) nulový, musí proto být pro výpočet použita rovnice (3.36). Naopak pokud se uzel nachází na ose y, je jmenovatel rovnice (3.36) nulový a je třeba použít rovnici (3.37). Pro všechny ostatní uzly jsou výsledky obou rovnic totožné. [3]

3.3. Řízení tensegritických struktur

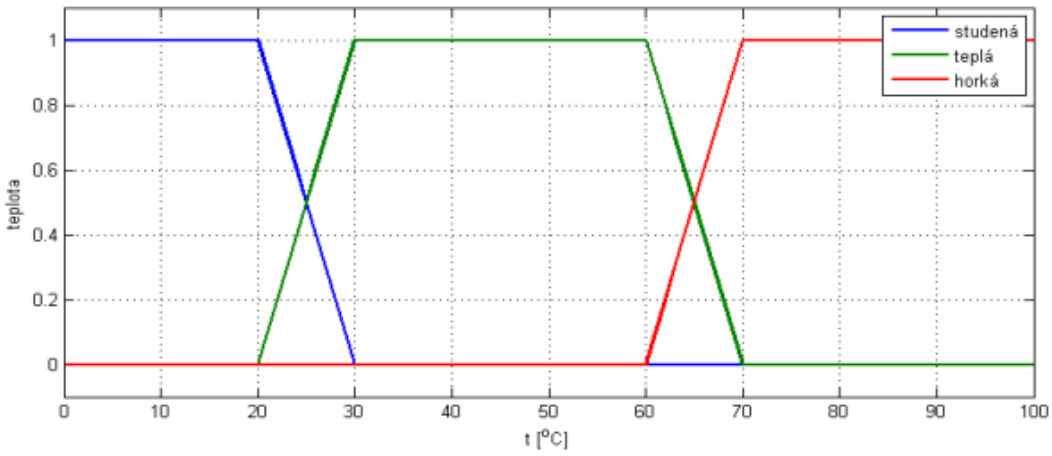
V době psaní diplomové práce (2021), nebyl zatím odvozen žádný obecný zákon řízení pro tensegritické struktury. Hlavní překážkou v tomto ohledu je fakt, že teorie řízení, jak ji známe dnes, dovoluje energeticky efektivní návrh řízení až poté, co je vyřešena otázka fyzikálních a mechanických vlastností jednotlivých komponent. Jednou z hlavních motivací pro využití tensegritních struktur je možnost integrování zákona řízení již ve fázi návrhu rozměrů struktury. Ideou řízené tensegritické struktury je vyšší úroveň spolupráce mezi vlastní dynamikou aktuátoru a samotné struktury. Toho je docíleno využitím řízení pro cílenou změnu vlastní rovnovážné polohy struktury. [1]

4. Seznámení s identifikačními postupy pomocí sítě lokálních lineárních modelů

Hlavní ideou identifikace je aproximace funkce $f(x)$ funkcí $\tilde{f}(x)$, kde minimalizuji určité kritérium. Ve většině případů tímto kritériem bývá součet kvadrátů odchylek funkcí $f(x)$ a $\tilde{f}(x)$. V této práci je implementován algoritmus LOLIMOT, kde je navíc tento kvadrát odchylky ještě přenásoben tzv. funkcí platnosti. Více o tomto postupu v kapitole 5.

4.1. Fuzzy logika

Jak bylo zmíněno již v úvodu, identifikace pomocí lokálních lineárních modelů využívá tzv. fuzzy logiku. Od klasické výrokové logiky se liší tak, že místo pouze dvou pravdivostních hodnot (0 nebo 1) může nabývat všech hodnot v intervalu ⟨0,1⟩. Těchto hodnot je tedy nekonečně mnoho. Přiblížení na příkladu je na Obr. 6



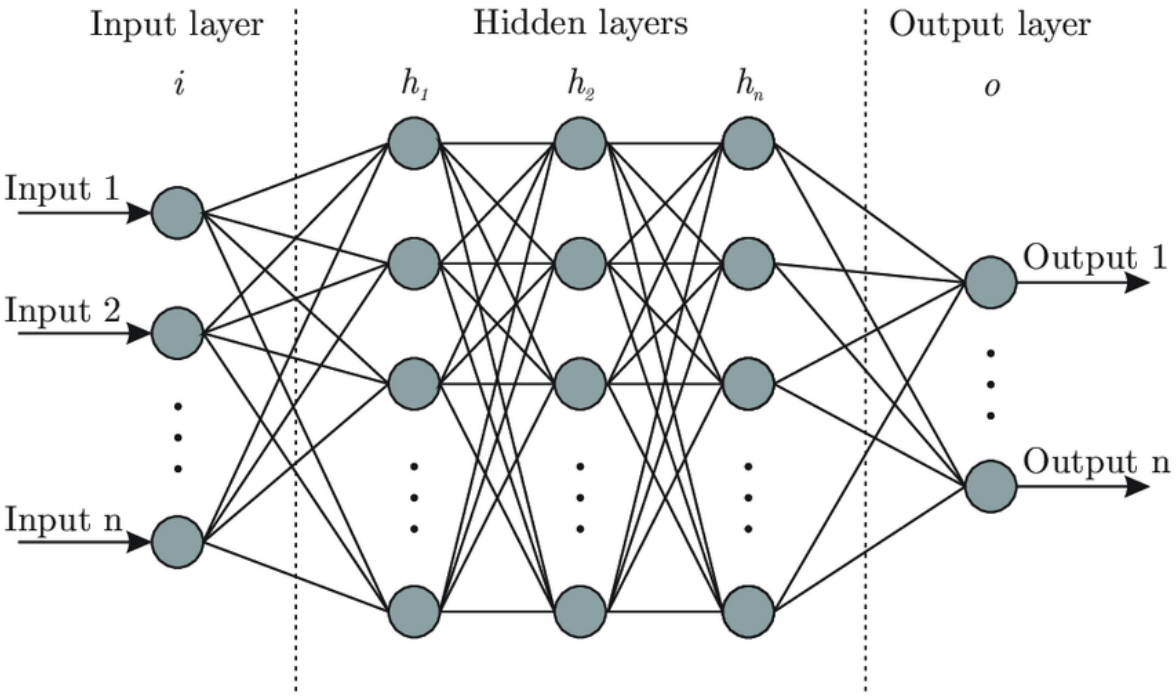
Obr. 6 Příklad fuzzy logiky [5]

Zde je vidět graf pocitové teploty na skutečné teplotě. Při přechodu mezi studenou a teplou neplatí jen jedna teplota, ale s určitým procentuálním příspěvkem platí obě

hodnoty. Čím více se blížíme k pocitově teplé hodnotě, tím méně platí studená pocitová hodnota, až klesne na 0.

4.2. Neuronové sítě

Podobnost názvu se základním prvkem lidského mozku není vůbec náhodná. Počítačové neuronové sítě jsou inspirované těmi biologickými a přejímají základní koncepty, zejména schopnost učit se nebo zpracovávat více informací najednou. Jejich základní schéma je na Obr. 7 Schéma neuronové sítě.



Obr. 7 Schéma neuronové sítě [6]

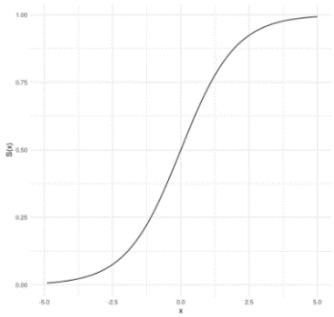
Síť se skládá ze tří vrstev – vstupní vrstvy (Input layer), skrytých vrstev (hidden layers) a výstupní vrstvy (Output layer). Základním stavebním kamenem je neuron. Každý neuron má obecně n vstupů a 1 výstup, který vstupuje do všech neuronů v následující vrstvě. Výstup neuronů ve výstupní vrstvě je výstup celé sítě. Příkladem takového výstupu může být např. procentuální pravděpodobnost, zda se na obrázku, který síť zpracovává, nachází určitý objekt, nebo ne. K výpočtu výstupu neuronu postupujeme dle následující rovnice:

$$y = \sigma\left(\sum_{i=1}^{n} w_i x_i - \theta\right) \tag{4.1}$$

Kde w_i jsou váhy jednotlivých neuronů, θ je tzv. bias. Jde vlastně o offset vážené hodnoty neuronu. Tyto parametry jsou předmětem učení. Dále x_i je hodnota ve vstupním vektoru, popř. v předchozí vrstvě sítě. Každá vrstva obsahuje aktivační funkci σ. Ta se volí nelineární. Pokud bychom zvolili aktivační funkci lineární, síť by nedokázala aproximovat nelineární funkce. Jednou z nejčastějších aktivačních funkcí je hyperbolický tangens, známý také jako sigmoid. Má rovnici

$$\sigma = \frac{1}{(1 + e^{-x})} \tag{4.2}$$

Graf vypadá následovně:

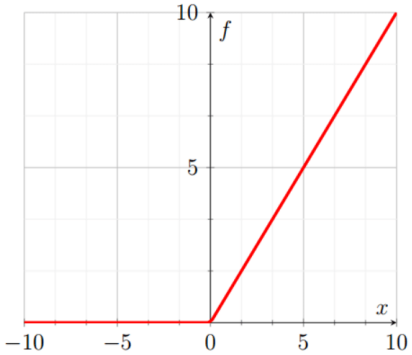


Obr. 8 Graf funkce sigmoid [7]

Obecně se v oblasti umělé inteligence často uplatňuje aktivační funkce ReLU (Rectified Linear Unit) definovaná jako:

$$f(x) = x^+ = \max(0, x) \tag{4.3}$$

A graf vypadá takto:



Obr. 9 Aktivační funkce ReLU [8]

Kde x je vstup neuronu. Tato aktivační funkce se používá zejména ve zpracování obrazu nebo rozpoznání řeči. Mezi výhody použití této funkce patří zejména výpočtová náročnost, je zde využito pouze porovnání, sčítání a odčítání. Dále např. tzv. řídká aktivace. Při náhodné inicializaci sítě se aktivuje pouze polovina skrytých neuronů. Mezi nevýhody patří její necentrovanost v nule a neomezenost.

4.3. Neuro-fuzzy sítě

Hlavní výhodou neuro-fuzzy sítí, oproti klasickým hlubokým neuronovým sítím, je přiblížení k lidskému způsobu myšlení. Toho je dosaženou použitím nebinární fuzzy logiky. Největší výhodou neuro-fuzzy sítí je, že jsou univerzální aproximátory. Právě této vlastnosti se využívá při identifikaci systémů. Vezměme v potaz dva protipóly, tedy dosažitelnost a přesnost. Jedna z těchto vlastností v praxi převládne. Na dosažitelnost se zaměřuje tzv. jazykové fuzzy modelování (Mamdani model [9]) a na přesnost pak Takagi-Sugeno-Kang (TSK) modely [10].

4.4. Trénování neuronových sítí

Neuron je vlastně matematická funkce s N vstupy. Ke každému vstupu x_k pro $k \in (1,2,3,\dots,N)$ je přiřazena jeho váha w_k. Tyto váhy jsou klíčové pro celkový výstup neuronové sítě a jeho přesnost. Při testování jsou tyto parametry konstantní, ale při tréninku se v každém kroku mění tak, aby se celkový výstup sítě co nejvíce blížil požadované hodnotě. Jak již bylo zmíněno, výstup sítě je lineární kombinace hodnot vstupů a jejich vah, popř. zde figurují ještě jednotlivé offsety (bias) neuronů.

Nejdůležitější prvkem při trénování neuronové sítě je její chybová funkce (anglicky loss function). Ta nám říká, jak spolehlivá je naše síť v určité operaci. Vezměme každý testovací vzorek (například hodnotu pixelu v obrázku) a vypočtěme výstup sítě. Ten porovnejme se vstupní hodnotou. Výraz chybové funkce může být následující:

$$L(y,\hat{y}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y_i})^2, \tag{4.4}$$

Kde y_i je vstupní přesná hodnota, $\hat{y_i}$ je aproximovaná hodnota a N je počet vstupních vzorků. Pokud například používáme neuronovou síť pro detekci zvířete na obrázku, pro jednoduchost budeme uvažovat pouze 2 možnosti. Pes, nebo kočka. Určíme si, že pokud je na obrázku přítomen pes, má být výstup naší sítě 0. Pokud je ovšem na obrázku kočka, chceme, aby byl výstup roven 1. Tato hodnota odpovídá y, tedy číslu, které chceme ze sítě dostat. Pro výpočet chybové funkce půjdeme přes celý dataset (v tomto případě soubor obrázků). Pokud hodnota chybové funkce je velká, síť není dostatečně spolehlivá.

Při první iteraci tréninku inicializujeme váhy neuronů náhodně. To samozřejmě nebude podávat dobré výsledky, ale jednak výpočet určité hodnoty potřebuje a zadruhé chceme v průběhu tréninku hodnotu chybové funkce snižovat a tím zvyšovat přesnost sítě. Náš cíl je nalezení jiných váhových koeficientů za účelem snížení chybové funkce. Tento problém je ekvivalentní s principem minimalizace funkce N proměnných.

Existuje mnoho postupů a algoritmů pro minimalizaci, jsou buď gradientní, nebo negradientní. Jednou z nejjednodušších funkcí je gradientní spád (anglicky gradient descent). Vektor gradientu je vektor parciálních derivací podle všech proměnných. U nás jsou tyto proměnné váhové koeficienty *w*.

$$w_j = w_j - \mu \frac{\partial L}{\partial w_j}, \tag{4.5}$$

Důležitý parametr, který tento algoritmus využívá, je tzv. learning rate μ. Ten udává, jak velkým krokem postupujeme v každé iteraci. Pokud zvolíme learning rate příliš velký, jednoduše bod minima přeskočíme a algoritmus zdiverguje, naopak pokud bude learning rate moc malý, může se násobně prodloužit čas nalezení lokálního minima. [11]

Posledním bodem je výpočet gradientu. Ten probíhá v algoritmu „Backpropagation". Výpočet postupuje odzadu od výstupní vrstvy, přes všechny skryté vrstvy až ke vstupní vrstvě a pro odvození parciálních derivací se využívá řetězové pravidlo. Váhové parametry w se mění podle jejich vlivu na hodnotu chybové funkce a tím probíhá proces učení.

5. Algoritmus Local Linear Model Tree (LOLIMOT)

Idea algoritmu LOLIMOT je strategie „rozděl a panuj", čili štěpení složitého nelineárního problému na více jednodušších, lineárních problémů. Toto dělení je realizováno osově ortogonálními řezy celého n-dimenzionálního vstupního prostoru. Svou inkrementální výpočtovou složitostí patří mezi tzv. rostoucí algoritmy. V každé iteraci algoritmu jsou vypočítány funkce platnosti každého lokálního lineárního modelu (dále jen LLM). Jediný parametr, který závisí na volbě uživatele, je stupeň proporcionality k_σ. Oliver Nelles ve své disertaci [10] navrhl jako optimální hodnotu $k_\sigma = \frac{1}{3}$. Tento parametr ovlivňuje ostrost funkcí platnosti, jinými slovy určuje, jak moc se prolínají a na jak široké oblasti platí více funkcí platnosti najednou, každá s menší mírou vlastního příspěvku. [10]

5.1. Popis algoritmu

Algoritmus se skládá ze dvou smyček – vnitřní a vnější. Ve vnitřní smyčce jsou vyhodnocovány parametry lokálně metodou nejmenších čtverců. Vnější smyčka kontroluje splnění ukončovací podmínky, ta může mít buď tvar minimální změny v přírůstku, maximální počet lokálních lineárních modelů, nebo kontrolu konvergence. [5]

1. V prvním kroku algoritmu se vytvoří platnostní funkce celého neděleného vstupního prostoru a vyhodnotí se parametry LLM metodou nejmenších

čtverců. Mám tedy celou funkci aproximovanou jednou úsečkou/rovinou nebo odpovídajícím útvarem ve vyšších dimenzích podle dimenze vstupního prostoru. Hodnota funkce platnosti je v celém prostoru rovna jedné.

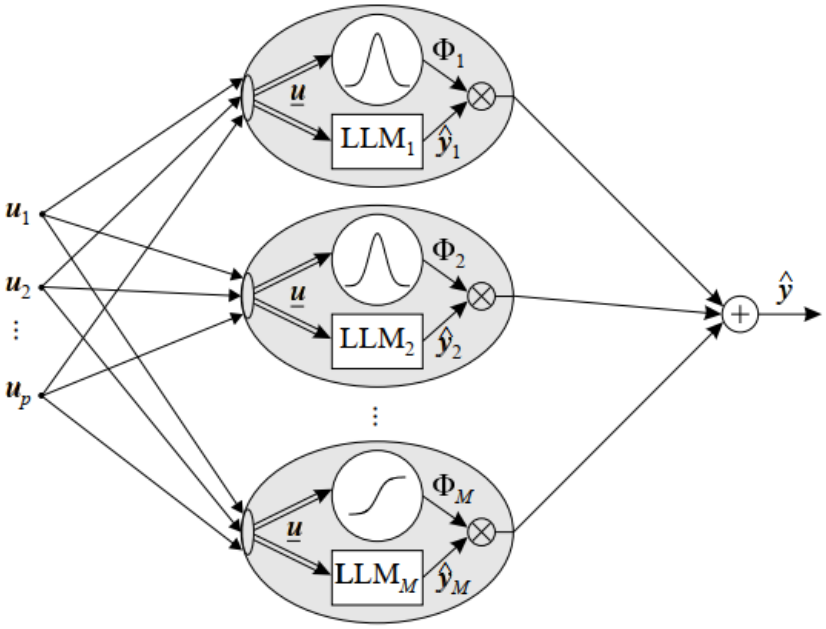
2. Nalezení nejhoršího LLM: Vypočtěte chybovou funkci pro všechny LLM a vyberte největší z nich. Tento LLM je určen pro dělení v dalším kroku. Chybová funkce má tvar:

$$E_i = \sum_{j=1}^{N} e^2(j)\phi_i(u(j)), \tag{5.1}$$

Kde $i = 1$. M je počet LLM, e je rozdíl mezi přesnou hodnotou a aproximovanou hodnotou a ϕ_i je funkce platnosti i-tého LLM, j je dimenze problému a $\underline{u}$ je vektor vstupních hodnot.

3. Proveď veškerá možná dělení a vyber nejvhodnější: Dělení probíhá osově ortogonálními řezy vždy na dvě poloviny prostoru. Pro každou možnost dělení je vypočítána funkce příslušnosti, funkce platnosti, vyhodnoceny parametry nově vzniklých LLM a vypočítána chybová funkce pro celý model. Dělení s nejmenší chybou je zvoleno jako provedené, zvýší se počet LLM: $M = M+1$. Zkontroluj ukončovací podmínky, pokud neplatí, pokračuj krokem 2.

Struktura sítě neuro fuzzy modelu je na Obr. 10



Obr. 10 Struktura algoritmu LOLIMOT [10]

Každý neuron obsahuje LLM_M a k němu odpovídající aktivační funkci Φ_M. Výstup LLM je součet hodnot všech neuronů,

$$\widehat{y_i} = w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \cdots + w_{ip}u_p, \tag{5.2}$$

kde w_{ij} jsou parametry LLM i-tého neuronu.

Funkce platnosti jsou normované, tzn. v každém bodě vstupního prostoru je celkový součet všech funkcí platnosti roven jedné podle rovnice:

$$\sum_{i=1}^{M} \Phi_i(u) = 1 \tag{5.3}$$

Tato vlastnost zajišťuje, že příspěvek všech LLM je v součtu 100 %. Příklad funkcí platnosti pro model s 6 neurony a dvoudimenzionální vstupní prostor při aproximaci funkce $y = \sin(x_1) + \cos(x_2)$ je na Obr. 11. Celkový výstup modelu neuro-fuzzy sítě je tedy součet vážených hodnot výstupů všech LLM,

$$\hat{y} = \sum_{i=1}^{M}(w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \cdots + w_{ip}u_p)\Phi_i(u), \tag{5.4}$$

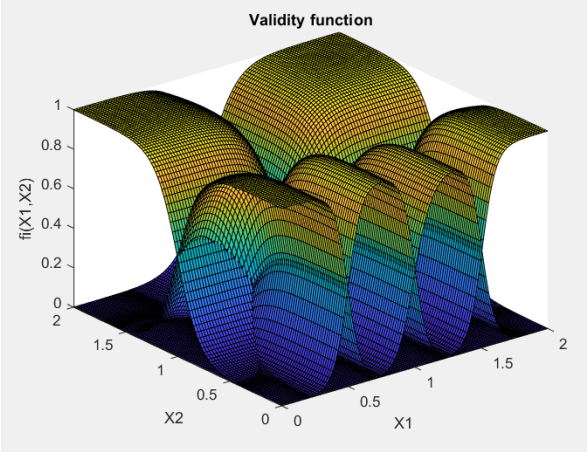
Model interpoluje mezi jednotlivými lokálními modely přes funkce platnosti, ty jsou voleny jako normalizované gaussiány, pokud jsou navíc osově ortogonální, můžeme psát:

$$\Phi_i(u) = \frac{\mu_i(u)}{\sum_{j=1}^{M} \mu_j(u)}, \tag{5.5}$$

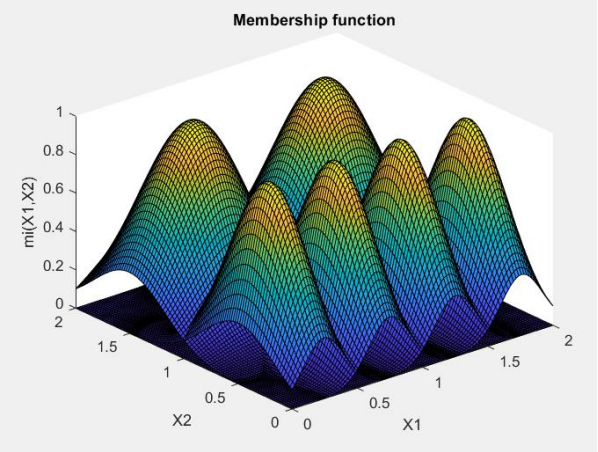
kde

$$\mu_i(u) = \exp\left(-0.5\left(\frac{(u_1 - c_{i1})^2}{\sigma_{i1}^2} + \frac{(u_2 - c_{i2})^2}{\sigma_{i2}^2} + \cdots + \frac{\left(u_p - c_{ip}\right)^2}{\sigma_{ip}^2}\right)\right) \tag{5.6}$$

Funkci μ_i nazýváme funkcí příslušnosti a je závislá na středech dělených oblastí c_{ij} a na směrodatných odchylkách σ_{ij}. Tyto parametry jsou nelineární a reprezentují skryté vrstvy neuronové sítě. Na Obr. 12 je vidět příklad funkcí příslušnosti pro model s 6 neurony a dvoudimenzionální vstupní prostor při aproximaci funkce $y = \sin(x_1) + \cos(x_2)$.

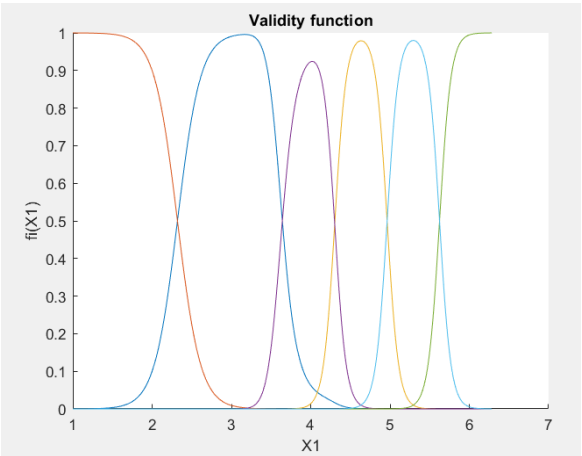


Obr. 12 Funkce platnosti (3D)

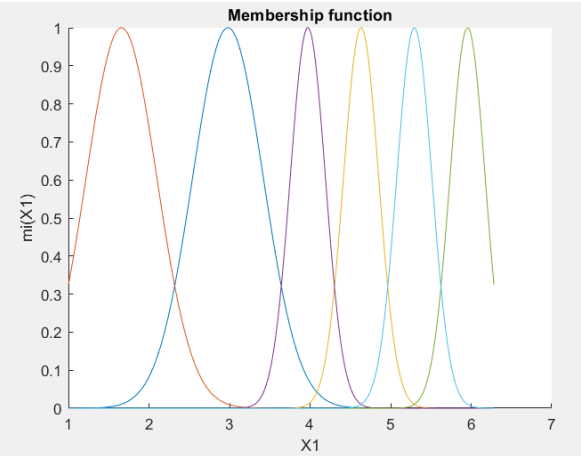


Obr. 11 Funkce příslušnosti (3D)

Na Obr. 13 je tvar funcí platnosti pro aproximaci funkce y = sin(x) šesti neurony a na Obr. 14 tomu odpovídající funkce příslušnosti.



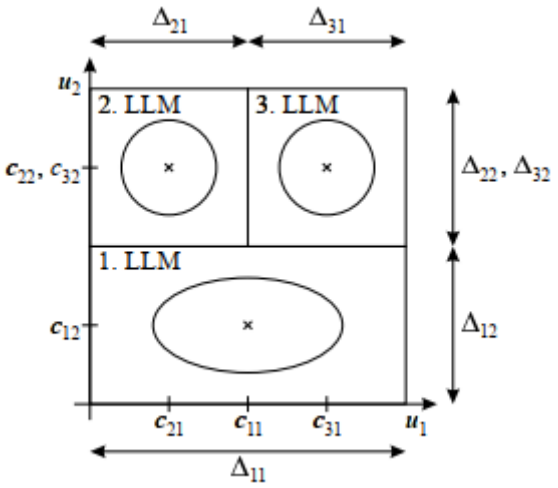
Obr. 13 Funkce platnosti (2D)



Obr. 14 Funkce příslušnosti (2D)

5.2. Strategie dělení prostoru

Počet funkcí platnosti a jejich parametrů (středy c_{ij} a směrodatné odchylky σ_{ij}) definují dělení vstupního prostoru. Na Obr. 15 Příklad dělení vstupního prostoru je vidět dělení dvoudimenzionálního vstupního prostoru na 3 oblasti, kde v každé z nich je střed jednoho lokálního lineárního modelu. Stojí za to připomenout, že jednotlivé lokální lineární modely jsou definovány vždy v celém vstupním prostoru, ovšem se vzdáleností od jejich středu příspěvek do výstupu celého modelu klesá s hodnotou jeho funkce platnosti.



Obr. 15 Příklad dělení vstupního prostoru
[10]

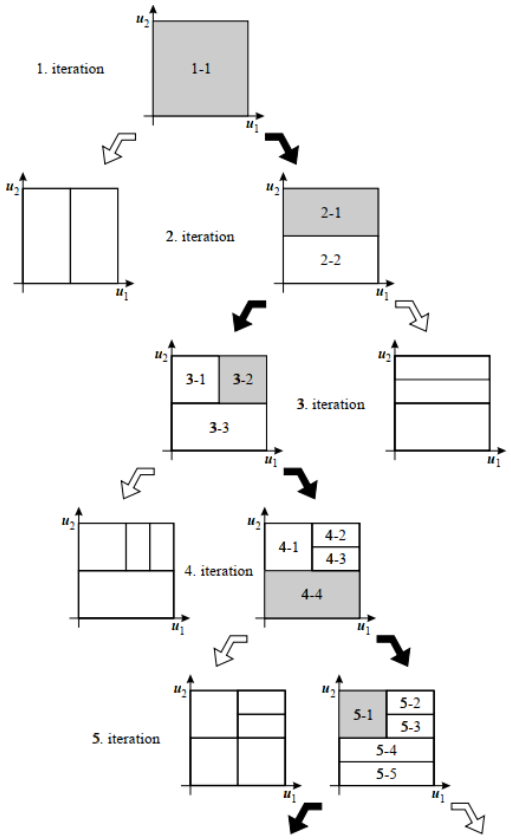
Kruhy a elipsa na Obr. 15 Příklad dělení vstupního prostoru zobrazují konturu funkce příslušnosti korespondujících lokálních lineárních modelů, Δ_{ij} je pak šířka *i*-té oblasti v *j*-té dimenzi. Oblast je dělena vždy na obdélníky popř. jim odpovídající útvary ve vyšších dimenzích řezy kolmými na osy vstupního prostoru.

Určení parametrů funkcí platnosti je problém nelineární optimalizace. Cesty k jejich nalezení mohou být následující: [10]

- *Dělení mřížky*: Počet funkcí příslušnosti je zvolen s ohledem na znalost systému. V tomto přístupu se ale naplno projeví problém multidimenzionality, tedy paměťová náročnost na alokaci vícedimenzionálních polí. Problém je třeba řešit s nižším počtem vstupních parametrů.
- *Clusterování vstupního prostoru*: Funkce platnosti jsou umístěny na základě rozmístění vstupních dat. Ovšem tento postup většinou nepodává uspokojivé výsledky, protože nebere v potaz komplexitu problému.
- *Nelineární lokální optimalizace*: Cílem je optimalizace globální metodou nejmenších čtverců. Tento přístup je výpočtově náročný, avšak zpravidla podává velice přesné výsledky.
- *Genetické algoritmy*: Evoluční algoritmy poskytují širokou paletu různých přístupů, všechny ovšem zaostávají v extrémně pomalé konvergenci oproti klasickým metodám.
- *Heuristické konstrukční algoritmy*: Zřejmě nejvíce rozšířená skupina algoritmů. Začínají s hrubým dělením prostoru a iteračně zvětšují rozlišení dělení vstupního prostoru. Zvětšují tedy komplexnost lokálních lineárních modelů při tréninku.

Na Obr. 16 je vidět iterativní postup dělení dvourozměrného vstupního prostoru. Vždy jsou provedena všechna možná osově ortogonální dělení oblasti určené k dělení, ostatní oblasti zůstávají nedotčené. Pro každou nově vzniklou oblast je vypočítána

funkce příslušnosti, z ní funkce platnosti, dále jsou určeny parametry lokálního lineárního modelu a chyba tohoto LLM. Pro celkový model je vybrán ten lokální lineární model, jehož chyba je nejmenší. Poté je určena chyba všech ostatních LLM, vybrán tn nejhorší a označen pro dělení v další iteraci algoritmu.



Obr. 16 Příklad možného postupu dělení oblastí [10]

5.3. Výpočtová náročnost

V každé iteraci algoritmu musí být zvolený nejhorší lokální lineární model rozdělen na poloviny v n dimenzích, takže vzroste počet určovaných parametrů o $2n$. Z toho vyplývá časová náročnost:

$$O(2n(n+1)^3) \cong O(2n^4) \tag{5.7}$$

Je zajímavé, že časová náročnost nezávisí na aktuálním indexu iterace, což je důsledkem přístupu lokální interpolace. To má za následek, že algoritmus při tréninku nezpomaluje a i velice složité modely mohou být vytvořeny efektivně. Pro celý algoritmus platí, že musí být provedeno M iterací. Z toho plyne, že:

$$O(2Mn(n+1)^3) \cong O(2Mn^4) \tag{5.8}$$

Výpočtová složitost stoupá lineárně s rozsahem modelu (počtem LLM). Navíc výpočtová složitost neroste exponenciálně dimenzí problému, což by působilo velké problémy. [10]

5.4. Globální interpolace

Výpočet parametrů lokálního lineárního modelu je problém lineární optimalizace za předpokladu, že jsou známy hodnoty funkcí platnosti. Je možné použít dva různé přístupy ke zjištění parametrů LLM, těmi jsou globální a lokální interpolace.

Při globální interpolaci jsou všechny parametry lokálních lineárních modelů určeny současně v jediném kroku. Vektor parametrů obsahuje $M(p + 1)$ parametrů, kde M je počet neuronů a p počet vstupů.

$$\underline{w} = [w_{10}\ w_{11} \dots w_{1p}\ w_{20}\ w_{21} \dots\ w_{2p} \dots\ w_{M0}\ w_{M1} \dots w_{Mp}]^T \tag{5.9}$$

S ním spojená regresní matice **X** pro N měřených vzorků je

$$\boldsymbol{X} = [\boldsymbol{X}_1^{sub}\ \boldsymbol{X}_2^{sub} \dots \boldsymbol{X}_M^{sub}] \tag{5.10}$$

Jednotlivé submatice jsou definovány jako:

$$\boldsymbol{X_i^{sub}} = \begin{bmatrix} \Phi_i(\underline{u}(1)) & u_1(1)\Phi_i(\underline{u}(1)) & u_2(1)\Phi_i(\underline{u}(1)) & \cdots & u_p(1)\Phi_i(\underline{u}(1)) \\ \Phi_i(\underline{u}(2)) & u_1(2)\Phi_i(\underline{u}(2)) & u_2(1)\Phi_i(\underline{u}(2)) & \cdots & u_p(1)\Phi_i(\underline{u}(2)) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Phi_i(\underline{u}(N)) & u_1(N)\Phi_i(\underline{u}(N)) & u_2(N)\Phi_i(\underline{u}(N)) & \cdots & u_p(1)\Phi_i(\underline{u}(N)) \end{bmatrix} \tag{5.11}$$

Výstup modelu $\hat{y}$ je potom

$$\underline{\hat{y}} = \boldsymbol{X}\underline{w} \tag{5.12}$$

V globální optimalizaci je minimalizována chybová funkce vzhledem k parametrům $\underline{w}$

$$I = \sum_{j=1}^{N} e^2(j) \rightarrow \min(\underline{w}) \tag{5.13}$$

Kde $\mathrm{e(j)} = \mathrm{y(j)} - \hat{y}(j)$ reprezentuje chybu modelu pro vzorky vstupních dat $\underline{u}(j)$.

Parametry jsou vypočítány metodou nejmenších čtverců, tedy pomocí pseudoinverze.

$$\underline{\widehat{w}} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T}\underline{y} \tag{5.14}$$

Kde $\underline{y} = [y(1)\, y(2) \ldots y(N)]^T$ obsahuje měřené výstupy. Problémem je ovšem výpočtová složitost, která roste se třetí mocninou počtu neuronů

$$\mathrm{O}(M^3(p+1)^3) \cong \mathrm{O}(M^3 p^3) \tag{5.15}$$

Oproti tomu lokální interpolace nabízí větší časovou optimalitu. [10]

5.5. Lokální interpolace

Ideou je, oproti globální optimalizaci, posuzovat vyhodnocování parametrů LLM jako separátní problémy. Navíc jsou zanedbány průniky mezi jednotlivými lokálními modely, ideálně je toto křížení nulové. Výstup modelu je posuzován pro daný vstup $\underline{u}$ pouze jedním lokálním lineárním modelem, zatímco všechny ostatní jsou neaktivní. Namísto určení všech $n = M(p+1)$ parametrů najednou, jak je tomu při globální interpolaci, je provedeno M lokálních interpolací pro p+1 parametrů každého lokálního lineárního modelu. Vektor parametrů pro každý $i = 1, \ldots, M$ neuron je:

$$\underline{w_i} = [w_{i0}\, w_{i1} \ldots w_{ip}]^T \tag{5.16}$$

Tomu odpovídají regresní matice:

$$\boldsymbol{X}_i = \begin{bmatrix} 1 & u_1(1) & u_2(1) & \cdots & u_p(1) \\ 1 & u_1(2) & u_2(2) & \cdots & u_p(2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & u_1(N) & u_2(N) & \cdots & u_p(N) \end{bmatrix} \tag{5.17}$$

V tomto případě jsou všechny regresní matice pro lokální modely $i = 1, \ldots, M$ stejné, protože elementy $\boldsymbol{X}_i$ nezávisí na i. Výstup i-tého lokálního lineárního modelu bude

$$\underline{\hat{y}_i} = \boldsymbol{X}_i \underline{w_i}, \tag{5.18}$$

Ten je však platný pouze v oblasti spojené s jeho funkcí platnosti Φ_i, přesněji tam, kde se hodnota Φ_i blíží jedničce. Tato varianta nastane v místech blízkých středu funkce platnosti. Hodnoty v okolí tohoto bodu jsou vysoce relevantní k určení parametrů $\underline{w_i}$. S poklesem hodnoty funkce platnosti se i relevance dat pro daný LLM snižuje a naopak se zvyšuje pro sousední LLM. Navíc je snadné použít metodu vážených nejmenších čtverců, kde proporčními koeficienty jsou hodnoty funkce platnosti Φ_i.

$$I_i = \sum_{j=1}^{N} \Phi_i(\underline{u}(j)) e^2(j) \rightarrow \min(\underline{w_i}) \tag{5.19}$$

Zde opět $\mathrm{e(j)} = \mathrm{y(j)} - \hat{y}(j)$ udává chybu modelu. Pro zvláštní případy, kdy $\Phi_i(\underline{u}(j)) = 0$ nebo $\Phi_i(\underline{u}(j)) = 1$, je pro estimaci použita pouze podmnožina dat,

v ostatních případech jsou využita všechna data, i když ta s hodnotou funkce platnosti rovnou nule jsou doslova ignorována.

Pro výpočet hodnot parametrů lokálně lineárních modelů je opět použita pseudoinverze

$$\underline{\widehat{w_i}} = (\boldsymbol{X_i^T Q_i X_i})^{-1} \boldsymbol{X_i^T Q_i} \underline{y} \tag{5.20}$$

Kde $\boldsymbol{Q_i}$ je váhová matice:

$$\boldsymbol{Q}_i = \begin{bmatrix} \Phi_i(\underline{u}(1)) & 0 & \cdots & 0 \\ 0 & \Phi_i(\underline{u}(2)) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Phi_i(\underline{u}(N)) \end{bmatrix} \tag{5.21}$$

Výpočet $\underline{\widehat{w_i}}$ musí být proveden pro všech $i = 1, \dots, M$ lokálních lineárních modelů. Hlavní výhodou tohoto přístupu je jeho nízká výpočtová náročnost. Oproti globální interpolaci je zde zapotřebí určit pouze p+1 parametrů bez ohledu na počet neuronů. Jelikož lokální interpolace musí být provedena pro každý lokálně lineární model zvlášť, roste výpočtová náročnost pouze lineárně s počtem neuronů

$$\mathrm{O}(\mathrm{M}(p+1)^3) \cong \mathrm{O}(p^3) \tag{5.22}$$

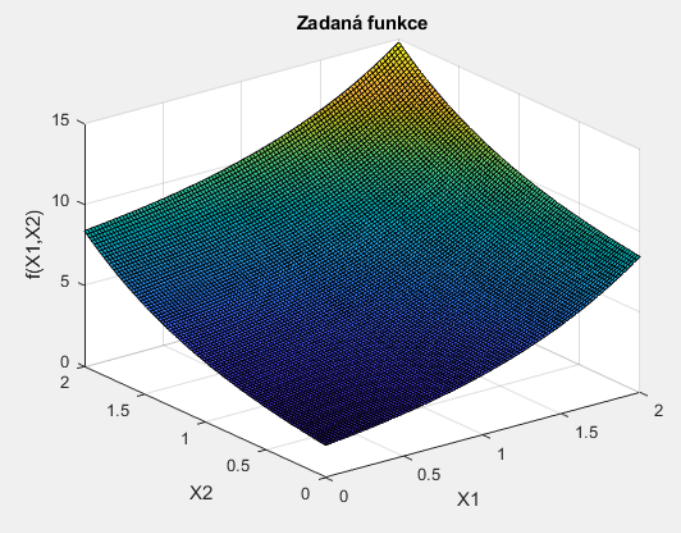
Tato výhoda roste kvadraticky s velikostí modelu. Cenou za takto nízkou výpočtovou složitost je zavedení systematické chyby při zanedbání vztahů mezi lokálními modely. [10]

5.6. Příklad použití algoritmu LOLIMOT v prostředí MATLAB

Pro demonstraci jsem zvolil aproximovanou funkci dvou proměnných

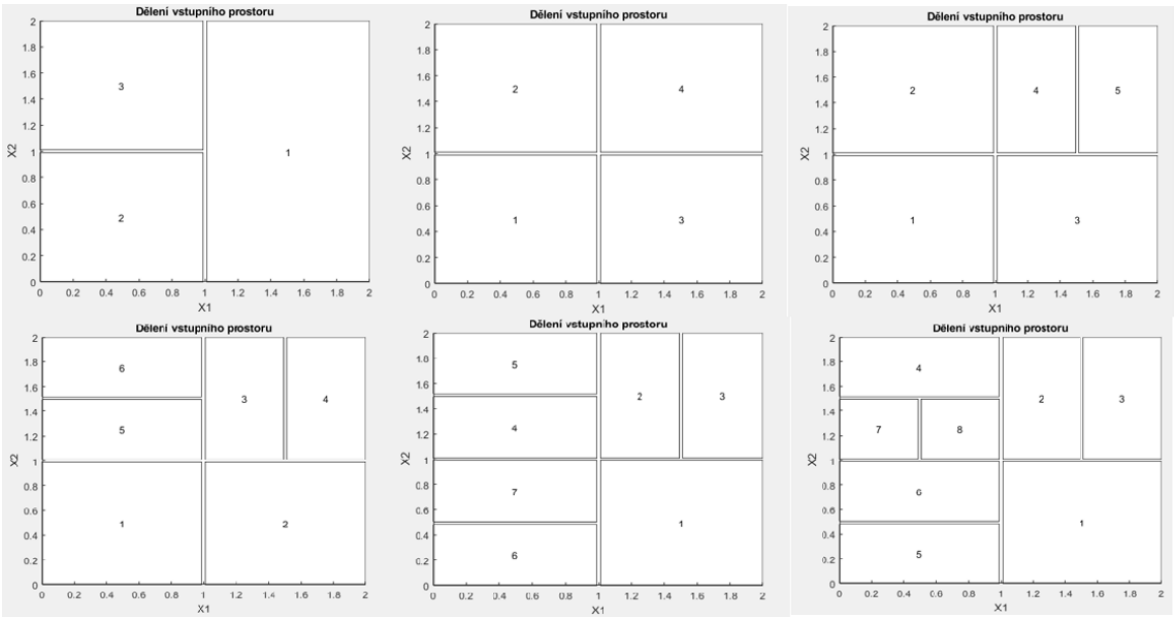
$$\mathrm{f}(x_1, x_2) = e^{x_1} + e^{x_2} \tag{5.23}$$

Pro $x_1 \in \langle 0; 2 \rangle$ a stejně tak $x_2 \in \langle 0; 2 \rangle$. Graf této funkce je na Obr. 17. Je jasné, že tato funkce je nelineární. Pokusím se tedy tuto nelineární funkci aproximovat sítí lokálních modelů. Pro dvoudimenzionální vstupní prostor to znamená množinu rovinných ploch.



Obr. 17 Graf aproximované funkce

Z Obr. 18 je patrný postup dělení vstupního prostoru. Je vybrán nejhorší lokální lineární model, provedeny všechny možné osově ortogonální řezy a vybrán ten nejvhodnější a nahrazení původního LLM dvěma novými.



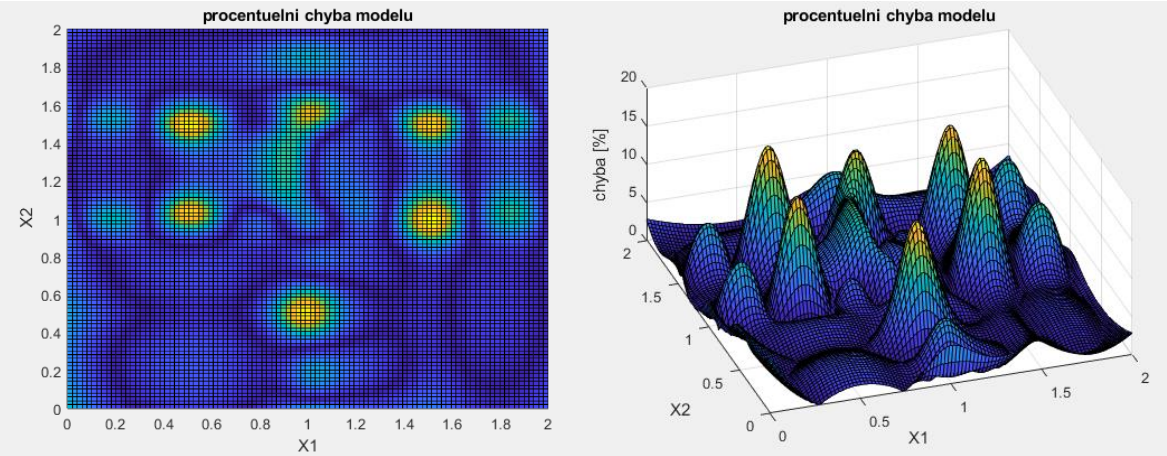
Obr. 18 Postupné dělení prostoru

Při tvorbě modelu může ovšem nastat problém, tzv. „over-fitting“. Tedy že se pro aproximaci použije příliš velký počet lokálních lineárních modelů a celková chyba modelu oproti přesné funkci bude větší, než kdyby se použil nižší počet neuronů. Demonstrujme tento problém na zadané funkci (5.23). Při aproximaci pomocí 10 neuronů je chyba modelu $E_c = 645{,}7$.

Tato chyba je počítána pomocí vzorce:

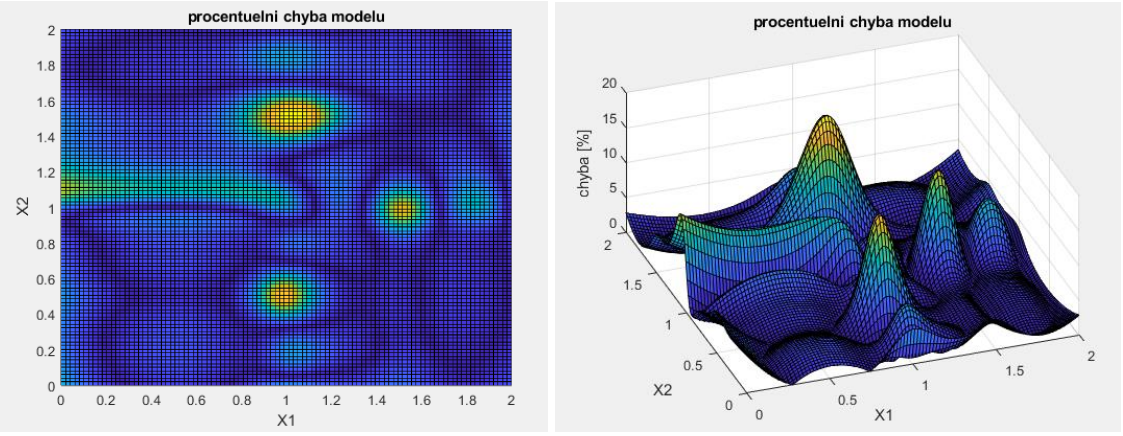
$$E_c = \sum_{i=1}^{M} (\mathrm{f}(x_1, x_2) - \hat{\mathrm{f}}(x_1, x_2))^2 \cdot \Phi_i(x_1, x_2), \tag{5.24}$$

kde $\hat{\mathrm{f}}(x_1, x_2)$ je hodnota výstupu modelu. Na Obr. 19 je graf procentuální chyby modelu oproti zadané funkci. Chyba se pohybuje v jednotkách procent, ovšem jsou místa, kde chyba vzroste až na 16 %.



Obr. 19 Procentuální chyba modelu s 10 neurony

Pokud bychom stejnou funkci aproximovali pomocí 8 neuronů, globální chyba by byla pouze $E_c = 577{,}4$. Tedy asi o 11 % nižší. I graf rozložení procentuální chyby v celém vstupním prostoru by vypadal lépe, přestože maximální hodnota procentuální chyby by vzrostla na 18 %, v celém vstupním prostoru by se ale celkově zmenšila.

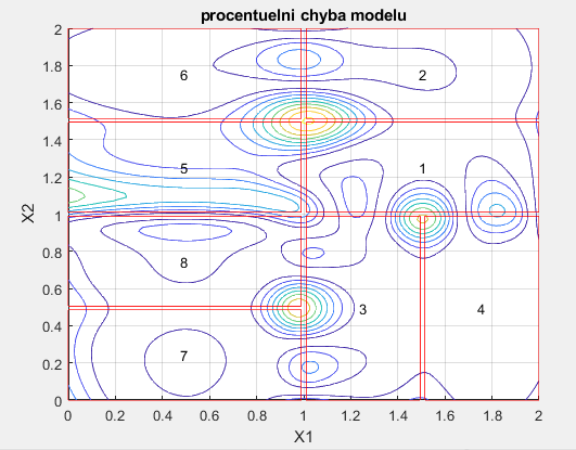


Obr. 20 Procentuální chyba modelu s 8 neurony

Je tedy nutné sledovat vývoj chyby modelu a v případě souvislého nárůstu mezi iteracemi aproximaci ukončit. Důležitá je poznámka, že hodnota chyby E_c se jako číslo zdá vysoká, ale je třeba se podívat na její výpočet. Jde o součet rozdílu modelu a

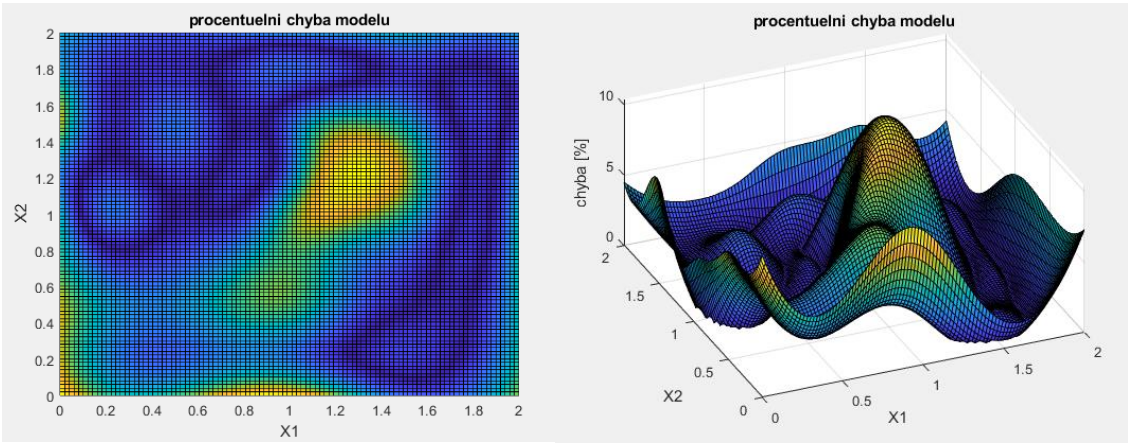
zadané funkce v každém bodě vstupního prostoru. Čím hustší mřížku (grid) zvolíme, tím existuje více bodů, ve kterých se chyba počítá. Aby byla chyba imunní vůči změně znaménka, je tento rozdíl umocněn na druhou. To hodnotu ještě zvětšuje. Posuzujeme-li tedy kvalitu modelu podle tohoto parametru, musíme sledovat pouze procentuální změnu napříč iteracemi, ne absolutní hodnotu.

Dalším zajímavým fenoménem, který je zřejmý již z grafů procentuální chyby, jsou vysoké hodnoty chyb v určitých oblastech. Při bližším posouzení dojdeme k závěru, že tyto oblasti jsou místa křížení hranic jednotlivých lokálních lineárních modelů. Tento poznatek je patrný z Obr. 21, kde jsou červenými obdélníky naznačeny ony hranice LLM. Graf pochází z předchozího případu aproximace zadané funkce pomocí 8 neuronů.



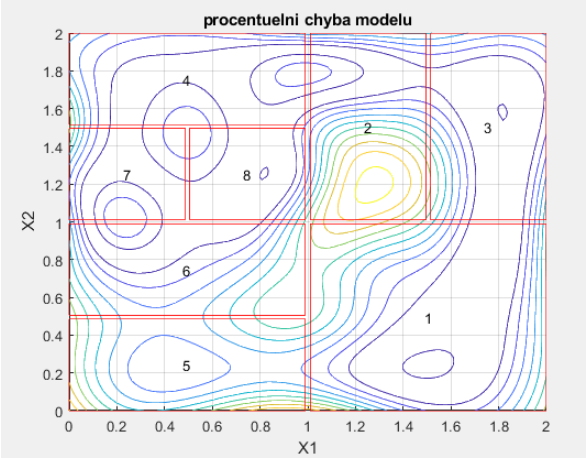
Obr. 21 Koncentrace chyby modelu

Tento problém lze adresovat změnou ostrosti funkcí příslušnosti μ_i a z nich počítaných funkcí platnosti Φ_i, tedy parametrem k_σ. Výše zmíněný model je vypočítán s doporučenou hodnotou $k_\sigma = \frac{1}{3}$ [10]. Pokud změníme jeho hodnotu na $k_\sigma = \frac{1}{2}$, dostaneme kvalitativně lepší model s menší chybou. Maximální procentuální chyba modelu bude pouze 10 %, oproti dřívějším 18 %.



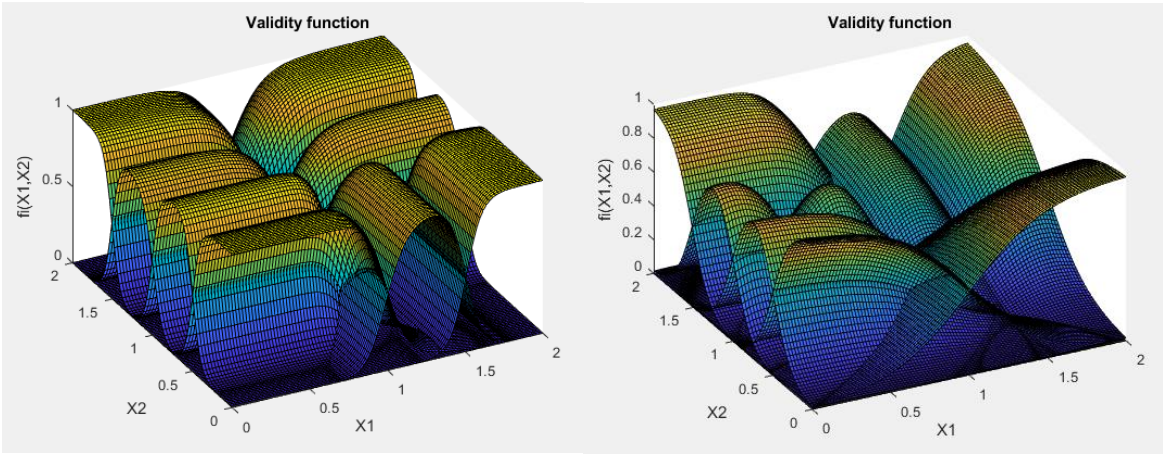
Obr. 22 Procentuální chyba modelu, $k_\sigma = 0{,}5$

Zároveň se zmenšil počet píků, to je dáno větším stupněm prolnutí funkcí platnosti. Existuje tedy větší oblast, kde ve větší míře platí více lokálních lineárních modelů najednou. Posun píků od míst křížení lokálních lineárních modelů je patrný z Obr. 23.



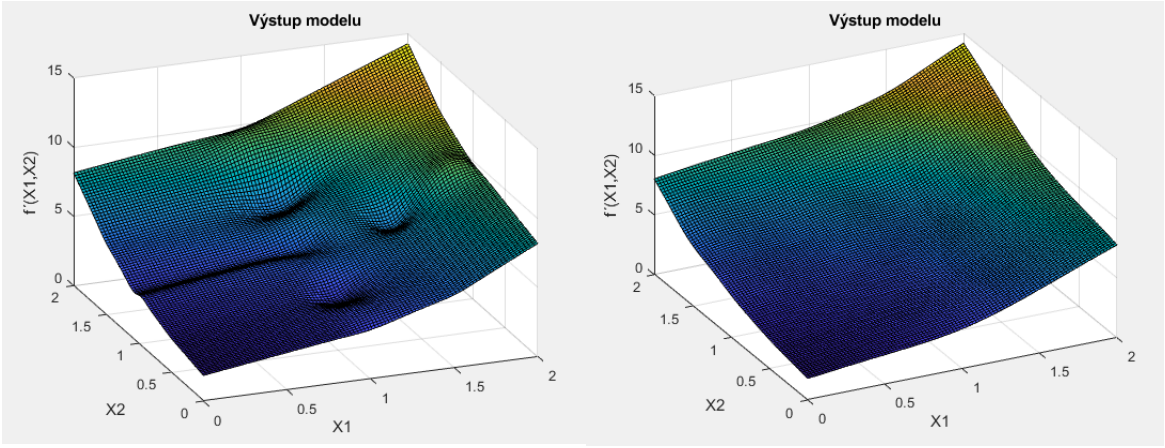
Obr. 23 Procentuální chyba v kontextu dělení prostoru

Změna tvaru funkcí platnosti je vidět z Obr. 24. V levé části je tvar funkcí platnosti pro hodnotu $k_\sigma = \frac{1}{3}$, vpravo pak pro $k_\sigma = \frac{1}{2}$.



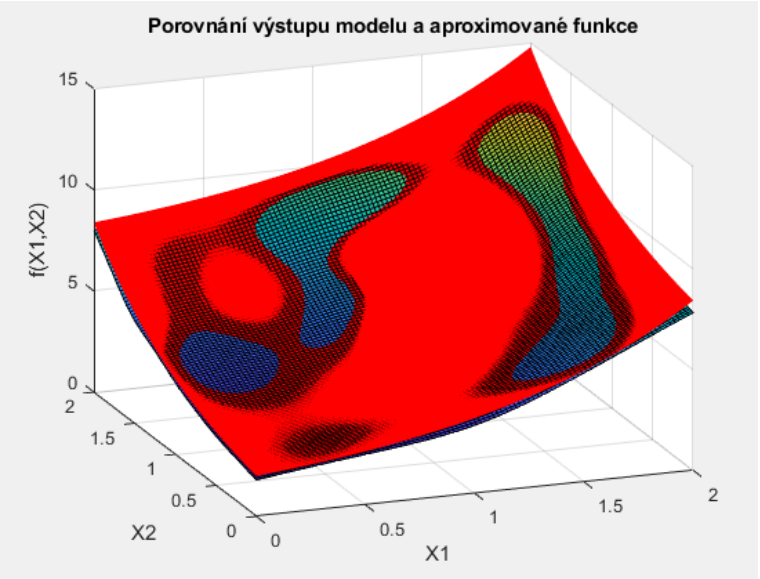
Obr. 24 Funkce platnosti, $k_\sigma = 0{,}33$ *vs* $k_\sigma = 0{,}5$

Rozdíl v těchto hodnotách můžeme pozorovat i na tvaru výstupní funkce modelu. Vlevo na Obr. 25 je opět model pro $k_\sigma = \frac{1}{3}$ a vpravo pro $k_\sigma = \frac{1}{2}$. Model pro vyšší hodnotu k_σ má obecně hladší tvar a nevyskytují se v něm náhlé změny snižující kvalitu modelu jako ve druhé variantě. Obecně ovšem nelze říct, jaká je optimální hodnota parametru k_σ. Záleží na znalosti aproximovaného systému.



Obr. 25 Výstup modelu, $k_\sigma = 0{,}33$ *vs* $k_\sigma = 0{,}5$

Celkový výstup, společně s porovnáním se zadanou funkcí, je na Obr. 26. Červeně je znázorněna hodnota zadané funkce, barevně pak výstup modelu. Z tohoto porovnání je zřejmé, že algoritmus LOLIMOT dokáže být velice přesným aproximátorem kombinujícím výhody neuronových sítí, tedy schopnost učit se, a velice efektivní výpočtovou náročnost.



Obr. 26 Porovnání zadané funkce a modelu

5.7. Implementace v prostředí MATLAB

Ve mnou vytvořené implementaci tohoto algoritmu v prostředí MATLAB jsou veškeré důležité hodnoty ukládány do objektu *model*. V terminologii programu MATLAB se objekt nazývá struktura. Na Obr. 27 jsou vidět vnořené parametry objektu pro tvorbu modelu ze zadané funkce (5.23) .

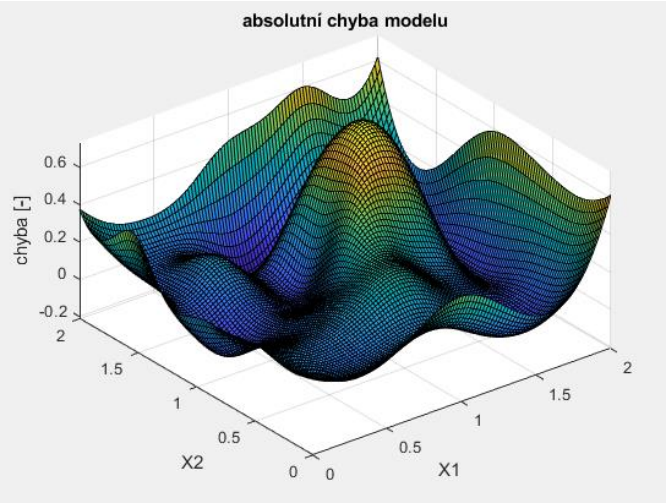
```
model =

  struct with fields:

                          w: {8×1 cell}
                         mi: {8×1 cell}
                         fi: {8×1 cell}
                          X: {1×8 cell}
                          Y: {1×8 cell}
    historie_absolutni_chyba: {[2.2917e+03]  [1.2727e+03]  [561.6860]  [1.2630e+03]  [618.4589]  [719.3628]  [485.9797]}
                     vystup: [100×100 double]
                      chyba: [137.2180 145.9165 76.8648 34.5920 25.8074 36.5916 6.6301 22.3592]
                  max_chyba: 145.9165
              globalni_chyba: 485.9797
    globalni_chyba_relativni: 0.0486
             absolutni_chyba: [100×100 double]
          procentuelni chyba: [100×100 double]
```

Obr. 27 Objekt *model*

Struktura obsahuje následující parametry:

- *w*: parametry lokálních lineárních modelů, zde koeficienty rovnice plochy $w_1 + xw_2 + yw_3 = 0$. Trojice parametrů pro každý lokální lineární model
- *mi*: hodnoty funkcí příslušnosti pro každý LLM
- *fi*: hodnoty funkcí platnosti pro každý LLM
- *X*: x-ové souřadnice každého LLM

- *Y*: y-ové souřadnice každého LLM
- *historie_absolutni_chyba*: vývoj chyby v každé iteraci, dle tohoto parametru lze kontrolovat problém over-fittingu
- *vystup*: hodnota výstupu modelu, modelový příklad je v mřížce 100 x 100, čili hodnota v každém bodě vstupního prostoru
- *chyba*: chyba každého lokálního lineárního modelu zvlášť. Tento parametr určuje LLM vhodný pro dělení. Pořadí v poli odpovídá i-tému LLM, jak je znázorněno na Obr. 18
- *max_chyba*: chyba nejhoršího LLM
- *globalni_chyba*: součet chyb všech LLM v dané iteraci
- *globalni_chyba_relativni*: průměrná chyba v každém bodě vstupního prostoru
- *absolutni chyba*: hodnota rozdílu zadané funkce a výstupu modelu v každém bodě. Pro demonstrativní příklad vypadá graf následovně:



Obr. 28 Absolutní chyba modelu

- *procentuelni_chyba*: Rozdíl výstupu modelu a zadané funkce v procentech. Graf na Obr. 22

6. Manipulovatelnost

Pojem manipulovatelnost, neboli dexterita, je v podstatě zobecnění převodu mezi rychlostí pohonů a koncového efektoru. Vyjadřuje, jak efektivně jsou pohony využity v aktuální konfiguraci mechanismu. Nabývá hodnot v intervalu $\langle 0,1 \rangle$, přičemž vyšší hodnota znamená lepší manipulovatelnost.

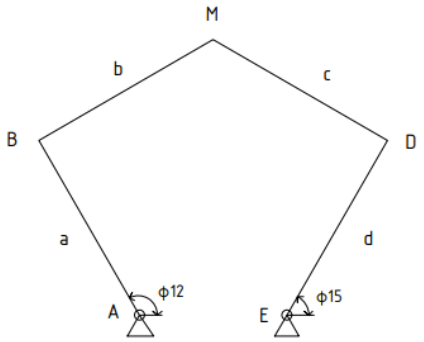
Výpočet manipulovatelnosti probíhá dle vzorce

$$D = \frac{1}{cond(\boldsymbol{J})}, \tag{6.1}$$

tedy převrácená hodnota podmíněnosti Jacobiho matice **J**. Její rozměr je obecně $n \times 2$, nebo $n \times 3$ podle toho, jestli se pohybujeme v rovině, nebo v prostoru. Číslo n udává počet pohonů mechanismu. V následující kapitole se omezíme pouze na rovinu.

$$\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \boldsymbol{J} \begin{bmatrix} \dot{x_M} \\ \dot{y_M} \end{bmatrix} \tag{6.2}$$

Podmíněnost matice je zde vyjádřena jako poměr vlastních čísel. Jakobiho matice určuje vztah rychlostmi pohonů a koncového efektoru. Jako modelový příklad je použit mechanismus na Obr. 29.



Obr. 29 pětikloubový mechanismus

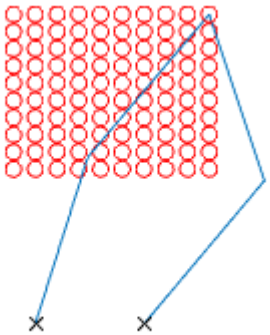
Tento jednoduchý pětikloubový mechanismus se skládá ze 4 ramen stejné délky. V rotačních podpěrách jsou umístěny pohony A a E, koncovým efektorem je bod M. Aby matice **J** pro tento příklad splňovala předpis

$$\begin{bmatrix} v_B \\ v_D \end{bmatrix} = \boldsymbol{J} \begin{bmatrix} \dot{x_M} \\ \dot{y_M} \end{bmatrix}, \tag{6.3}$$

má tvar

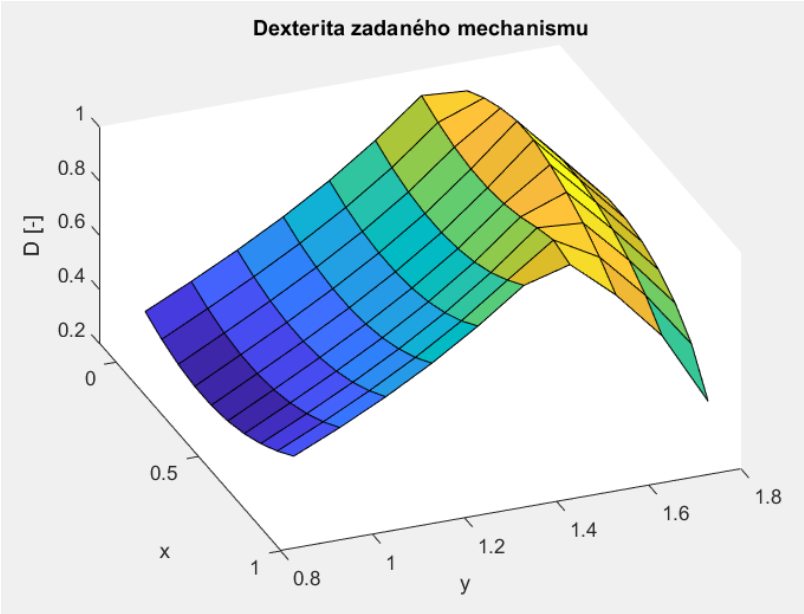
$$\boldsymbol{J} = \begin{bmatrix} \dfrac{x_M - x_B}{-(x_M - x_B)\sin\varphi_{12} + (y_M - y_B)\cos\varphi_{12}} & \dfrac{y_M - y_B}{-(x_M - x_B)\sin\varphi_{12} + (y_M - y_B)\cos\varphi_{12}} \\ \dfrac{x_M - x_D}{-(x_M - x_D)\sin\varphi_{15} + (y_M - y_D)\cos\varphi_{15}} & \dfrac{y_M - y_D}{-(x_M - x_D)\sin\varphi_{15} + (y_M - y_D)\cos\varphi_{15}} \end{bmatrix} \tag{6.4}$$

Byl zvolen pracovní prostor mechanismu (100 poloh, na Obr. 30 je každá poloha koncového efektoru znázorněna jako červený puntík) a konfigurace mechanismu je znázorněna v poslední poloze.



Obr. 30 Pracovní prostor mechanismu

Pro každý bod pracovního prostoru byla podle vztahu (6.1) vypočítána manipulovatelnost a byla vynesena do grafu na Obr. 31.



Obr. 31 Manipulovatelnost v definovaném pracovním prostoru

Z grafu je patrné, že na krajích pracovního prostoru není energie pohonů dostatečně efektivně využita, hodnota manipulovatelnosti je nízká. V další kapitole se budu zabývat optimalizací tohoto mechanismu pro navržení ideální struktury maximalizující hodnotu manipulovatelnosti v celém pracovním prostoru.

6.1. Singularita prvého typu

Při pohledu na Jacobiho matici (6.4) je patrné, že mohou nastat dvě extrémní situace. Tou první je nulovost jmenovatele. Vznikne tedy soustava rovnic

$$\begin{bmatrix}(-(x_M - x_B)\sin\varphi_{12} + (y_M - y_B)\cos\varphi_{12})\cdot v_B\\(-(x_M - x_D)\sin\varphi_{15} + (y_M - y_D)\cos\varphi_{15})\cdot v_D\end{bmatrix} = \begin{bmatrix}x_M - x_B & y_M - y_B\\x_M - x_D & y_M - y_D\end{bmatrix}\begin{bmatrix}\dot{x_M}\\\dot{y_M}\end{bmatrix} \quad (6.5)$$

Pokud platí výraz

$$(x_M - x_B)\sin\varphi_{12} + (y_M - y_B)\cos\varphi_{12} = 0 \quad (6.6)$$

Znamená to, že v_B je nadbytečný parametr. Z toho vyplývá, že

$$\frac{x_M - x_B}{y_M - y_B} = -\frac{\dot{y_M}}{\dot{x_M}} \quad (6.7)$$

V praxi to znamená, že vektor rychlosti bodu B je rovnoběžný s vektorem rychlosti bodu M. Tato podmínka je splněna na krajích pracovního prostoru.

6.2. Singularita druhého typu

Druhá extrémní situace je nulovost determinantu matice

$$\begin{bmatrix}x_M - x_B & y_M - y_B\\x_M - x_D & y_M - y_D\end{bmatrix} \quad (6.8)$$

Při úpravě do tvaru (6.5). Znamená to, že tato matice je singulární a neexistuje k ní inverzní matice. Platí

$$\begin{bmatrix}\frac{y_M - y_D}{\Delta} & \frac{-y_M + y_B}{\Delta}\\\frac{-x_M + x_D}{\Delta} & \frac{x_M - x_B}{\Delta}\end{bmatrix}\begin{bmatrix}(-(x_M - x_B)\sin\varphi_{12} + (y_M - y_B)\cos\varphi_{12})\cdot v_B\\(-(x_M - x_D)\sin\varphi_{15} + (y_M - y_D)\cos\varphi_{15})\cdot v_D\end{bmatrix} = \begin{bmatrix}\dot{x_M}\\\dot{y_M}\end{bmatrix} \quad (6.9)$$

kde

$$\Delta = (x_M - x_B)(y_M - y_D) - (x_M - x_D)(y_M - y_B) \quad (6.10)$$

Pokud $\Delta = 0$, jsou nadbytečnými parametry $\dot{x_M}$ a $\dot{y_M}$. Tento typ singularity lze geometricky popsat jako tuhost úsečky. Vektor rychlosti v_C je kolmý na vektor v_D. V teorii bez uvažování jakéhokoliv tření a nehomogenit těles by to znamenalo zaseknutí mechanismu v dané poloze.

7. Optimalizace manipulovatelnosti rekonfigurovatelného rovinného manipulátoru

Použijeme stejný pětikloubový mechanismus jako v předchozí kapitole, ovšem s tím rozdílem, že délka jednoho ramene bude proměnná v určitém rozsahu. Tímto ramenem bude rameno b. V praxi to znamená, že přidáme další pohon. V původní konfiguraci má mechanismus 2 stupně volnosti dle vzorce:

$$n = 3(t-1) - 2r - 2p - 3v = 3(4-1) - 2 \cdot 5 - 0 - 0 = 2° \tag{7.1}$$

Po přidání pohonu nám ale vznikne nová nezávislá proměnná, kromě dvou nezávislých úhlů φ_{12} a φ_{15} ještě b. Počet stupňů volnosti se zvýší na 3. Bude tedy existovat více možných konfigurací celého mechanismu pro dosažení požadovaných souřadnic koncového efektoru x_M a y_M. Nadefinujme si napřed pracovní prostor, vezměme 100 poloh v rozsahu $x \in \langle -0.1; 0.8 \rangle$ a $y \in \langle 0.9; 1.8 \rangle$ s krokem 0.1 v obou osách. Fyzickou dosažitelnost těchto souřadnic mechanismem budeme kontrolovat při výpočtu inverzní kinematiky. Body pracovního prostoru generuje jednoduchý skript na Obr. 32

```
1 -   clc; close all; clear all;
2 -   polohyM2=[];
3 -   for ii=-0.1:0.1:0.8      %x
4 -       for kk=0.9:0.1:1.8   %y
5 -           polohyM2=[polohyM2; ii   kk];
6 -       end
7 -   end
8 -   save polohyM2
```

Obr. 32 Generátor pracovního prostoru

Body B a D se musejí pohybovat po kružnicích z rotačních podpěr A resp. E s konstantním poloměrem zvoleným $a = d = 1$. Stejně tak se musejí pohybovat po kružnicích z bodu M. Analyticky lze úlohu polohy bodů B a D řešit jako průsečík kružnic. Z analytické geometrie vyplývá, že průsečík nemusí být žádný, může být jeden, nebo mohou být dva. To odpovídá řešení kvadratické rovnice. Pokud je diskriminant záporný, kružnice nemají společný průsečík a mechanismus není sestavitelný. Pokud je diskriminant roven nule, průsečík je právě jeden a pokud je diskriminant kladný, průsečíky jsou právě dva. V tomto případě vezmu pro bod B menší kořen (více vlevo) a pro bod D větší kořen (víc vpravo) a jim příslušné soustavy kvadratických rovnic. Soustava dvou rovnic pro výpočet bodu B vypadá následovně:

$$y_B{}^2 = a^2 - x_B{}^2 \tag{7.2}$$

$$b^2 = (x_B - x_M)^2 + (y_B - y_M)^2 \tag{7.3}$$

K řešení byl použit symbolický toolbox MATLAB. V tomto kroku byla provedena kontrola smontovatelnosti mechanismu, tedy pozitivnost diskriminantu.

Stejným způsobem proběhl i výpočet polohy bodu D dle rovnic:

$$d^2 = (x_D - x_E)^2 + (y_D - y_E)^2 \tag{7.4}$$

$$c^2 = (x_D - x_M)^2 + (y_D - y_M)^2 \tag{7.5}$$

Toto bylo provedeno pro pevně zadanou velikost b. Jelikož je ale tato hodnota proměnná, můžeme toho využít pro nalezení optimální konfigurace mechanismu, tedy maximalizovat manipulovatelnost.

V kapitole 6 byla uvedena Jakobiho matice pro výpočet manipulovatelnosti daného pětikloubového mechanismu. Tento mechanismu ovšem není totožný, má rozdílný počet stupňů volnosti. Je tedy třeba odvodit novou Jakobiho matici, která odpovídá předpisu:

$$\begin{bmatrix} v_C \\ v_D \\ \dot{b} \end{bmatrix} = \hat{\boldsymbol{J}} \begin{bmatrix} \dot{x_M} \\ \dot{y_M} \end{bmatrix} \tag{7.6}$$

Pro úplnost, vztah dává proti sobě rychlosti pohonů a rychlost koncového efektoru, proto musí být v rovnici uvedena časová derivace polohy b. Pro snadnější odvození ale vezměme tento předpis inverzně, tedy:

$$\begin{bmatrix} \dot{x_M} \\ \dot{y_M} \end{bmatrix} = \boldsymbol{J} \begin{bmatrix} v_C \\ v_D \\ \dot{b} \end{bmatrix} \tag{7.7}$$

Vycházejme opět ze stejných rovnic, ovšem s tím rozdílem, že b je proměnná závislá na čase.

$$b^2 = (x_B - x_M)^2 + (y_B - y_M)^2 \tag{7.8}$$

Tuto rovnici zderivujme podle času

$$2b \cdot \dot{b} = 2(x_B - x_M)(\dot{x_B} - \dot{x_M}) + 2(y_B - y_M)(\dot{y_B} - \dot{y_M}) \tag{7.9}$$

Zaveďme substituci:

$$x_B = a \cos \varphi_{12} \tag{7.10}$$

$$y_B = a \sin \varphi_{12} \tag{7.11}$$

A tyto dvě rovnice zderivujme podle času:

$$\dot{x_B} = -a \cdot \dot{\varphi}_{12} \cdot \sin \varphi_{12} \tag{7.12}$$

$$\dot{y_B} = a \cdot \dot{\varphi}_{12} \cdot \cos \varphi_{12} \tag{7.13}$$

Postupnými úpravami dostaneme výraz

$$\boldsymbol{J} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{bmatrix}, \tag{7.14}$$

Kde

$$A_{11} = \frac{\left((x_B - x_M)\cdot \sin \varphi_{12} - (y_B - y_M)\cdot \cos \varphi_{12}\right) + \frac{(-\sin \varphi_{12}\cdot (x_M - x_D)\cdot (x_B - x_M) + \cos \varphi_{12}\cdot (y_B - y_M)\cdot (x_M - x_D))\cdot (y_B - y_M)}{-y_M\cdot (x_M - x_D) + y_B\cdot (x_M - x_D) + y_M\cdot (x_M - x_B) - y_D\cdot (x_M - x_B)}}{x_M - x_B} \tag{7.15}$$

$$A_{12} = \frac{\dfrac{(x_M - x_B)\cdot (x_D - x_M)\sin \varphi_{15} + (y_D - y_M)\cdot (x_M - x_B)\cdot (\cos \varphi_{15})}{-y_M\cdot (x_M - x_D) + y_B\cdot (x_M - x_D) + y_M\cdot (x_M - x_B) - y_D\cdot (x_M - x_B)}(y_B - y_M)}{x_M - x_B} \tag{7.16}$$

$$A_{13} = \frac{b + \dfrac{-b\cdot (x_M - x_D)}{-y_M\cdot (x_M - x_D) + y_B\cdot (x_M - x_D) + y_M\cdot (x_M - x_B) - y_D\cdot (x_M - x_B)}(y_B - y_M)}{x_M - x_B} \tag{7.17}$$

$$A_{21} = \frac{(x_D - x_M)\cdot (x_B - x_M)\cdot \sin \varphi_{12} + (y_B - y_M)\cdot (x_M - x_D)\cdot \cos \varphi_{12}}{-y_M\cdot (x_M - x_D) + y_B\cdot (x_M - x_D) + y_M\cdot (x_M - x_B) - y_D\cdot (x_M - x_B)} \tag{7.18}$$

$$A_{22} = \frac{(x_M - x_D) \cdot (x_D - x_M) \cdot \sin\varphi_{15} - (y_D - y_M) \cdot (x_M - x_B) \cdot \cos\varphi_{15}}{-y_M \cdot (x_M - x_D) + y_B \cdot (x_M - x_D) + y_M \cdot (x_M - x_B) - y_D \cdot (x_M - x_B)} \tag{7.19}$$

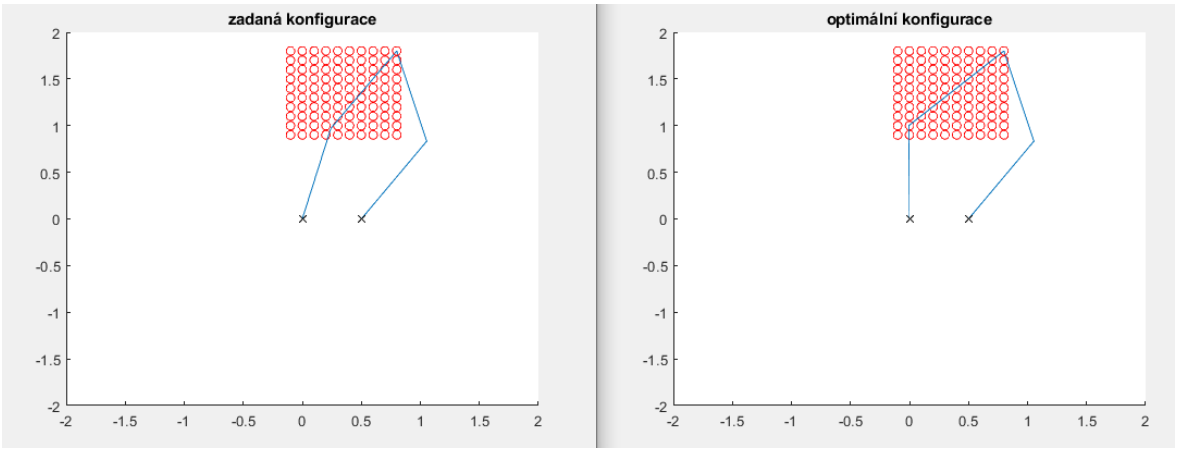
$$A_{23} = \frac{-b \cdot (x_M - x_D)}{-y_M \cdot (x_M - x_D) + y_B \cdot (x_M - x_D) + y_M \cdot (x_M - x_B) - y_D \cdot (x_M - x_B)} \tag{7.20}$$

Podmíněnost matice $\boldsymbol{J}$ je číslo v intervalu $cond(\boldsymbol{J}) \in \langle 1; \infty)$. Chceme-li maximalizovat manipulovatelnost, počítanou dle vzorce (6.1), aby se co nejvíce blížila 1, musíme podmíněnost matice minimalizovat. Proměnné v Jakobiho matici budou parametry pohonů v levé části mechanismu, tedy φ_{12} a b. Jelikož rameno u pohonu E nemá proměnnou délku a je předepsán pohyb bodu M, optimalizace i souřadnice φ_{15} by nepřinesla žádné změny. Nastavme hranice změny členu b na $\pm$ 50 % a φ_{12} na $\pm$ 0,3 rad. Aby se během výpočtu zachovala konstantní délka a, zaveďme nelineární podmínku:

$$0 = (a \cdot \sin\varphi_{12})^2 + (a \cdot \cos\varphi_{12})^2 - a^2 \tag{7.21}$$

Protože při výpočtu jsou počítány souřadnice bodů, ne jednotlivé délky, tato podmínka zaručí konstantní velikost a. Bez této podmínky by optimalizace proběhla s předpokladem, že i délka toho členu je proměnná.

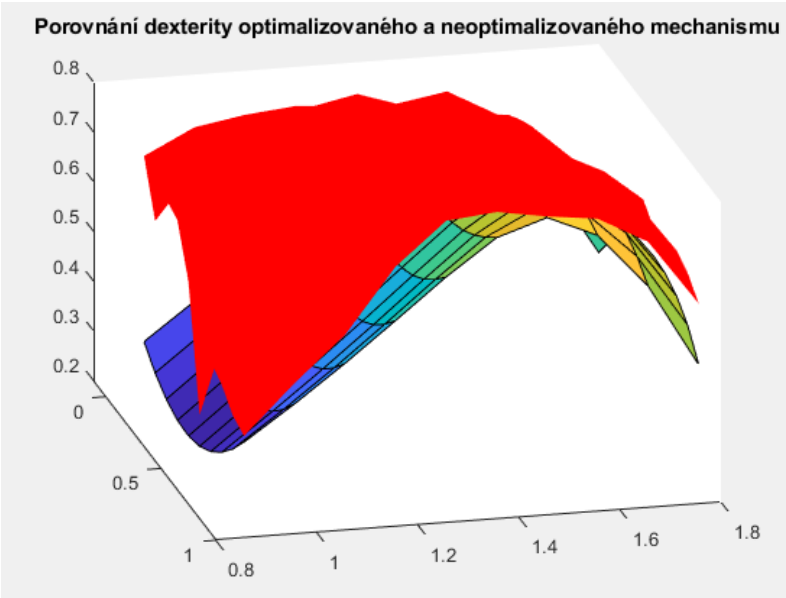
Problém nelineární optimalizace je řešen v prostředí MATLAB funkcí *fmincon*. Ta hledá lokální minimum zadané funkce nejblíže zadaných počátečních podmínek. Hrozí proto, jako u všech gradientních metod, nález nevhodného lokálního minima při špatné volbě počátečních podmínek.



Obr. 33 Porovnání původní a optimální konfigurace mechanismu

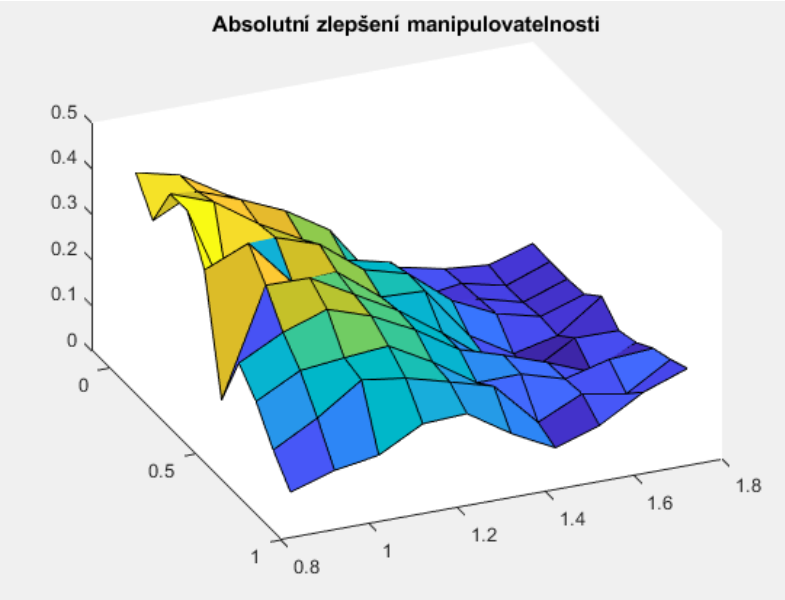
Na Obr. 33 je patrná konfigurace mechanismu v koncové poloze po provedení optimalizace. Pro tento konkrétní mechanismus lze „kvalitu manipulovatelnosti“ odhadnout již z obrázku. Pokud je mechanismus v „rozkročenější“ poloze, je využití sily z pohonů efektivnější než v případě užšího postoje.

Porovnání manipulovatelnosti optimalizovaného (červeně) a neoptimalizovaného mechanismu je na Obr. 34



Obr. 34 Porovnání manipulovatelnosti optimalizovaného a neoptimalizovaného mechanismu

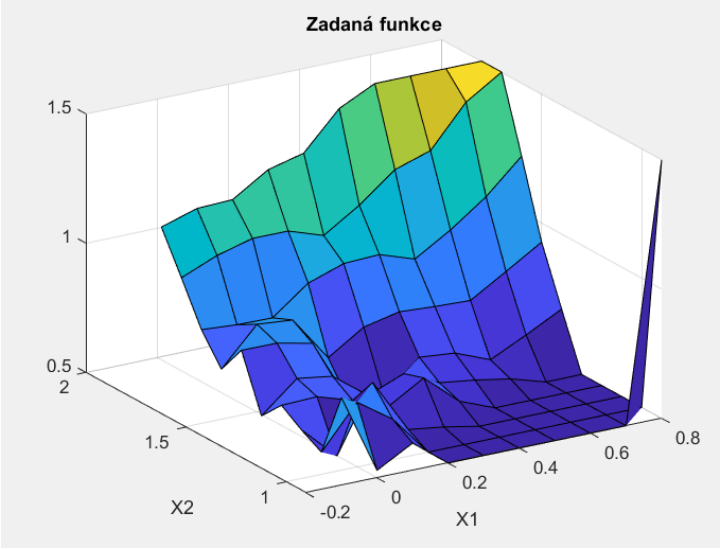
Absolutní změna je vidět na Obr. 35. Zlepšení je, zvlášť na kraji pracovního prostoru, poměrně značné.



Obr. 35 Absolutní změna manipulovatelnosti

7.1. Tvorba modelu

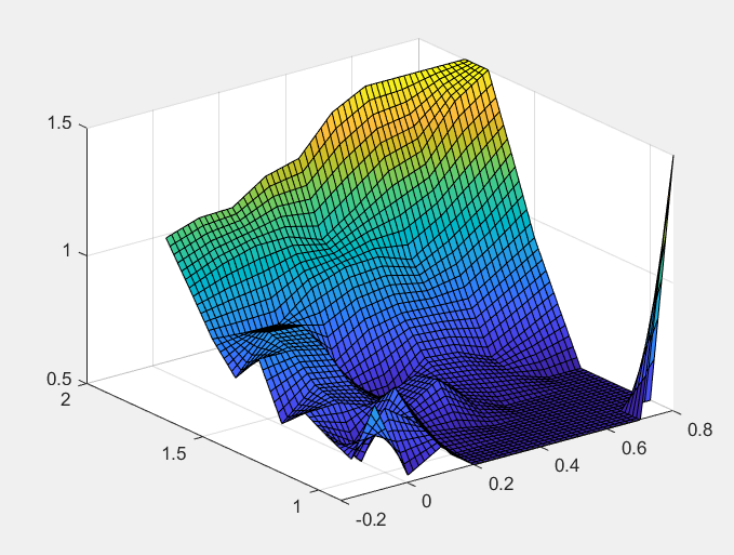
Následně byl vytvořen model manipulovatelnosti mechanismu pomocí sítě lokálně lineárních modelů. Jako vstupy modelu byly zvoleny souřadnice koncového efektoru x_M a y_M. Hodnota výstupu je délka proměnného členu *b*. Tato optimalizovaná hodnota byla vypočítána v celém pracovním prostoru pro 100 poloh. Graf funkce, kterou se budeme snažit modelovat pomocí algoritmu LOLIMOT, je na Obr. 36



Obr. 36 Funkce optimální hodnoty délky členu b

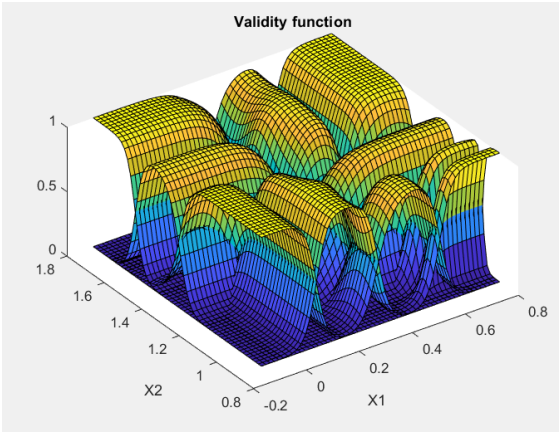
Aby mohla být hodnota optimalizované délky odečtena i mimo měřené polohy, byla vstupní síť zhuštěna. Toto zhuštění je provedeno parametricky a záleží na uživateli,

jak hustou síť chce zvolit. Pro tento případ jsem zvolil pětinásobek prvků původní sítě, viz Obr. 37

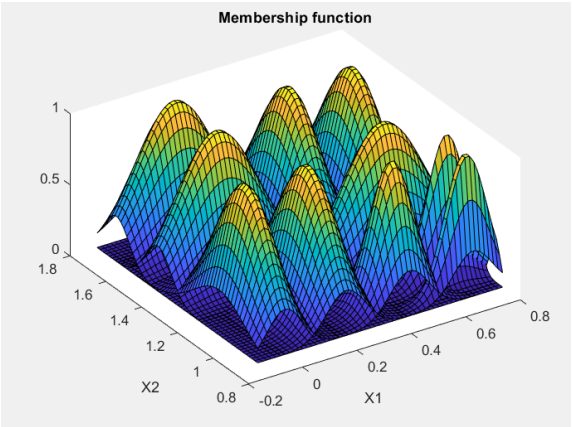


Obr. 37 Zhuštěná síť

Na Obr. 39 resp. Obr. 38 je vidět vypočtená funkce příslušnosti (membership function) a funkce platnosti (validity function). Vzhledem k minimalizaci chyby modelu byla zvolena hodnoty ostrosti těchto funkcí $k_\sigma = \frac{1}{3}$.

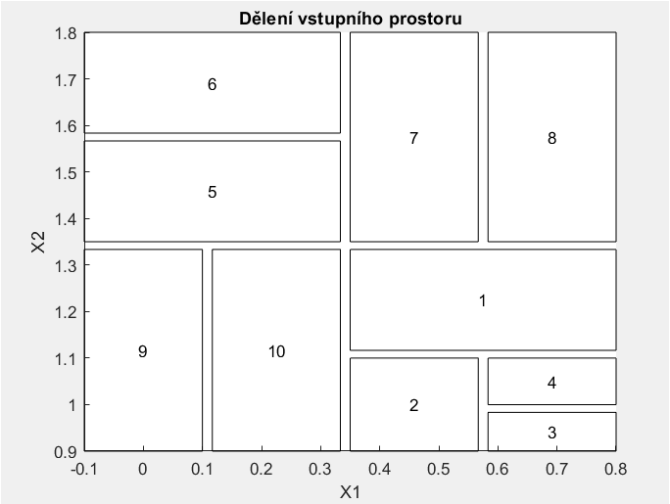


Obr. 38 Funkce platnosti



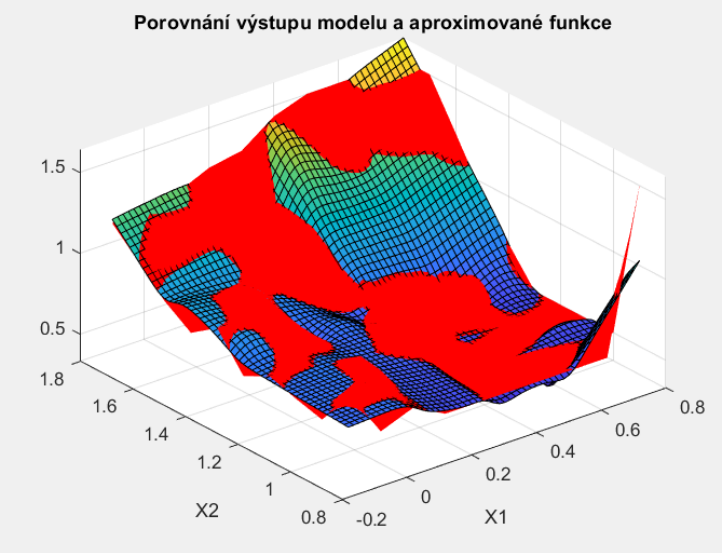
Obr. 39 Funkce příslušnosti

Model byl aproximován pomocí 10 neuronů. Dělení prostoru je vidět na Obr. 40. Dělen je vždy lokálně lineární model s největší chybou.



Obr. 40 Dělení vstupního prostoru, $k_\sigma = 0.5$

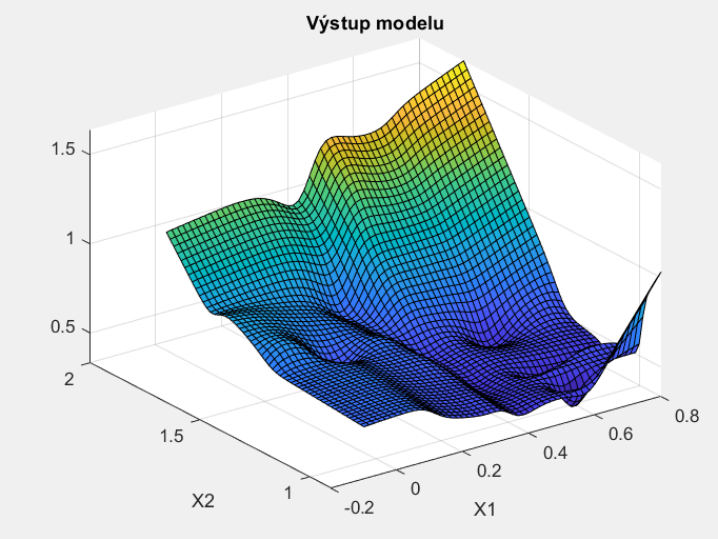
Konečné porovnání výstupu modelu (barevně) a aproximované funkce (červeně) je na Obr. 41. Na základě tohoto modelu je možné zvolit ideální délku proměnného členu bez nutnosti složitého výpočtu v daném pracovním prostoru. V případě více proměnlivých členů by se takovýto model vytvořil pro každý člen zvlášť. Největší nevýhodou algoritmu LOLIMOT je nutnost držet v paměti informace pro více dimenzí vstupního prostoru. Problém nastává již pro 5 rozměrů, kdy paměť potřebná pouze na alokaci takovéhoto pole je v řádech stovek GB.



Obr. 41 Porovnání modelu a zadané funkce

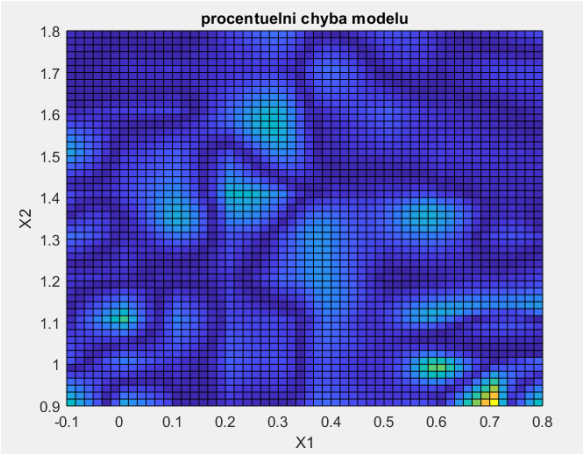
Samotný graf výstupu modelu je na Obr. 42. Vzhledem ke zvolené hodnotě k_σ je vidět určitá „vlnitost“ výsledné funkce. Ovšem i přes to podává v tomto konkrétním případě lepší výsledky než vyšší hodnota k_σ, která by tuto vlnitost odstranila, a model by

vycházel hladší, jelikož by každý jednotlivý lokální lineární model platil na větším prostoru.

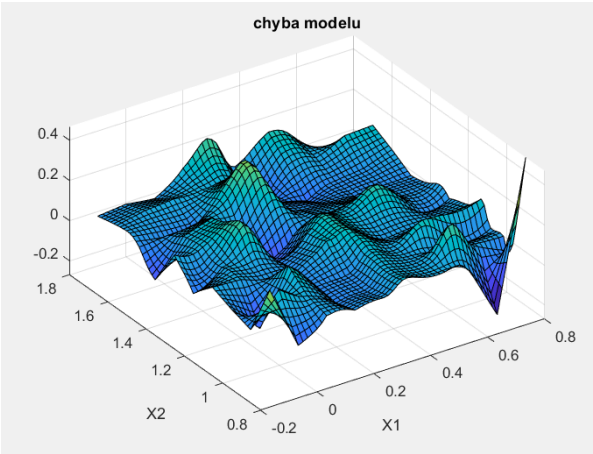


Obr. 42 Výstup modelu

Pokud bychom analyzovali chybu modelu, je jasné, že nejvyšší chyba by se nacházela v oblasti pravého dolního kraje pracovního prostoru. Tato oblast je ovšem poměrně malá a na zbytku pracovního prostoru podává model vcelku solidní výsledky. Chyba se pohybuje v řádu jednotek procent.

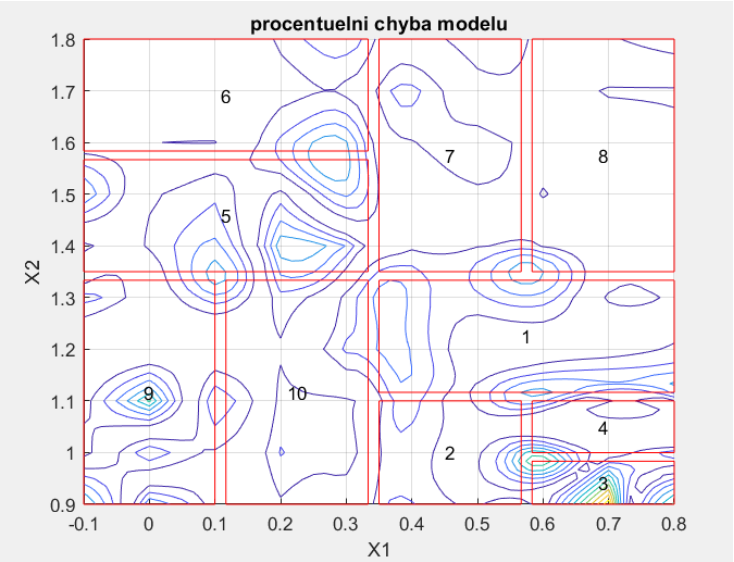


Obr. 43 Procentuální chyba modelu



Obr. 44 Absolutní chyba modelu

Jak bylo zmíněno v předchozí kapitole, tato hodnota k_σ způsobuje kumulaci chyby zejména v oblastech křížení lokálních lineárních modelů, jak je patrné z Obr. 45



Obr. 45 Procentuální chyba v kontextu dělení prostoru, $k_\sigma = 0.33$

Pokud se podíváme na parametr *historie_absolutni_chyba* na Obr. 46, je vidět, že celková chyba modelu se s přibývajícími iteracemi, tudíž zvětšujícím se počtem lokálních lineárních modelů, snižuje.

```
struct with fields:

                         w: {10×1 cell}
                        mi: {10×1 cell}
                        fi: {10×1 cell}
                         X: {1×10 cell}
                         Y: {1×10 cell}
  historie_absolutni_chyba: {[24.9910]  [19.0376]  [13.6780]  [11.3712]  [10.1648]  [8.7074]  [9.9557]  [9.0599]  [8.9806]}
                   vystup: [55×55 double]
                    chyba: [0.9314 0.2895 1.3334 0.1300 1.6727 1.3022 1.2207 0.5851 0.8861 0.6294]
                max_chyba: 1.6727
           globalni_chyba: 8.9806
  globalni_chyba_relativni: 0.0030
           absolutni_chyba: [55×55 double]
       procentuelni_chyba: [55×55 double]
```

Obr. 46 Objekt *model* - $k_\sigma = 0.33$

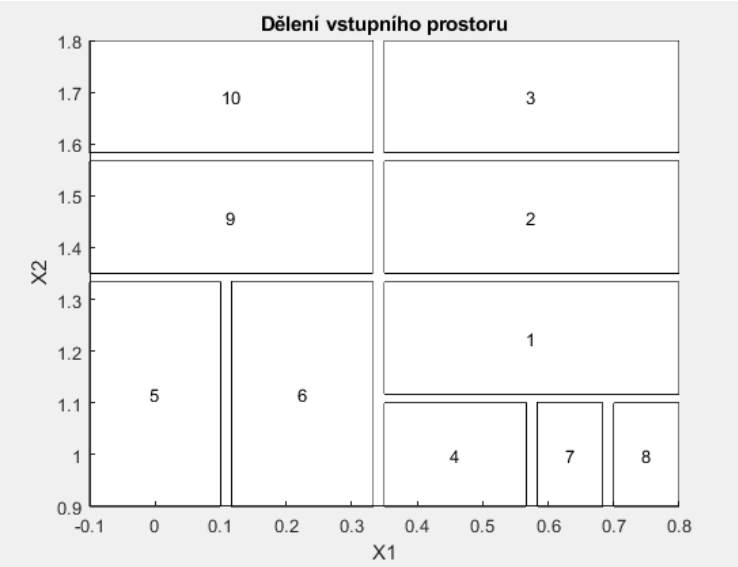
V případě dalšího dělení je zřejmé, že největší chybu má v aktuální iteraci lokální lineární model označený číslem 5. Ten by byl v případě dalšího postupu algoritmu zvolen k dělení v následující iteraci.

Pokud bychom zvolili hodnotu $k_\sigma = 0.5$, celková chyba modelu by byla napříč iteracemi větší než v předchozím případě, jak je vidět z Obr. 47.

```
struct with fields:

                        w: {10×1 cell}
                       mi: {10×1 cell}
                       fi: {10×1 cell}
                        X: {1×10 cell}
                        Y: {1×10 cell}
  historie_absolutni_chyba: {[26.7318]  [19.9406]  [14.2381]  [14.8487]  [11.7339]  [10.7915]  [11.8039]  [10.6680]  [11.4142]}
                   vystup: [55×55 double]
                    chyba: [2.5614 0.9721 0.8304 0.9551 1.5276 1.5384 0.2045 1.3411 1.1158 0.3679]
                max_chyba: 2.5614
           globalni_chyba: 11.4142
  globalni_chyba_relativni: 0.0038
           absolutni_chyba: [55×55 double]
        procentuelni_chyba: [55×55 double]
```

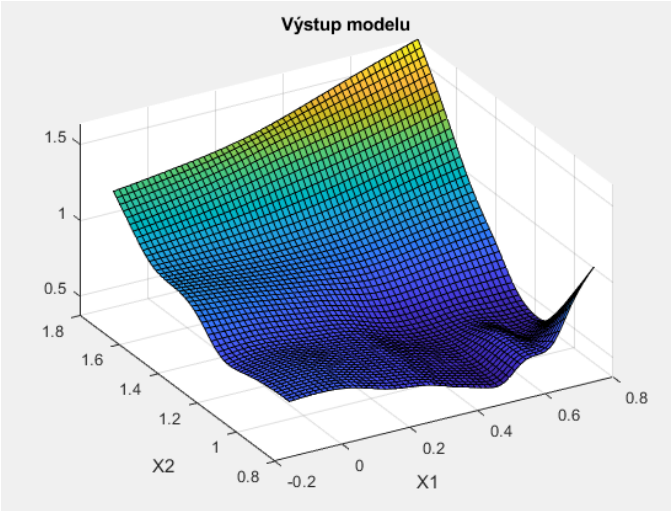
Obr. 47 Objekt *model* - $k_\sigma = 0.5$

Z různého výpočtu chyby, resp. funkcí platnosti, vychází i jiné dělení prostoru. Pro porovnání pro $k_\sigma = 0.5$ vychází následovně:



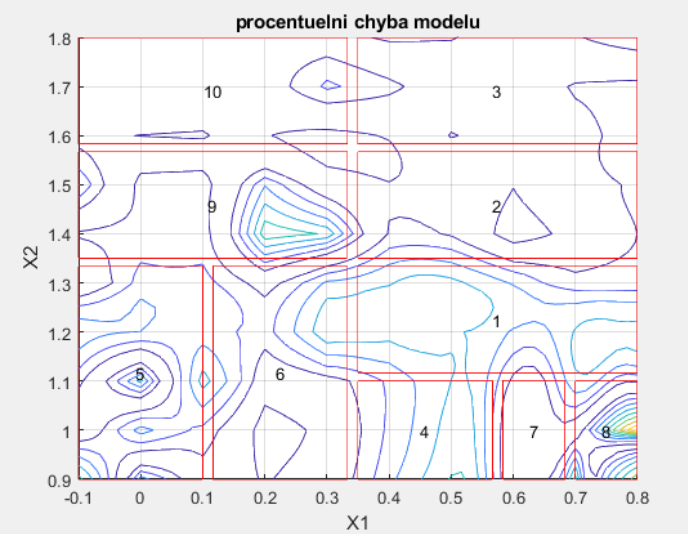
Obr. 48 Dělení vstupního prostoru, $k_\sigma = 0.33$

Je patrný rozdíl oproti Obr. 40. V dalším kroku by byla zvolena oblast s číslem 1. To opět vychází z parametru *max_chyba* na Obr. 47, kde se tato hodnota nachází v parametru *chyba* na první pozici. A jak již bylo zmíněno, výstupní funkce (Obr. 49) je hladší, bez zmiňované „vlnitosti“.



Obr. 49 Výstup modelu, $k_\sigma = 0.5$

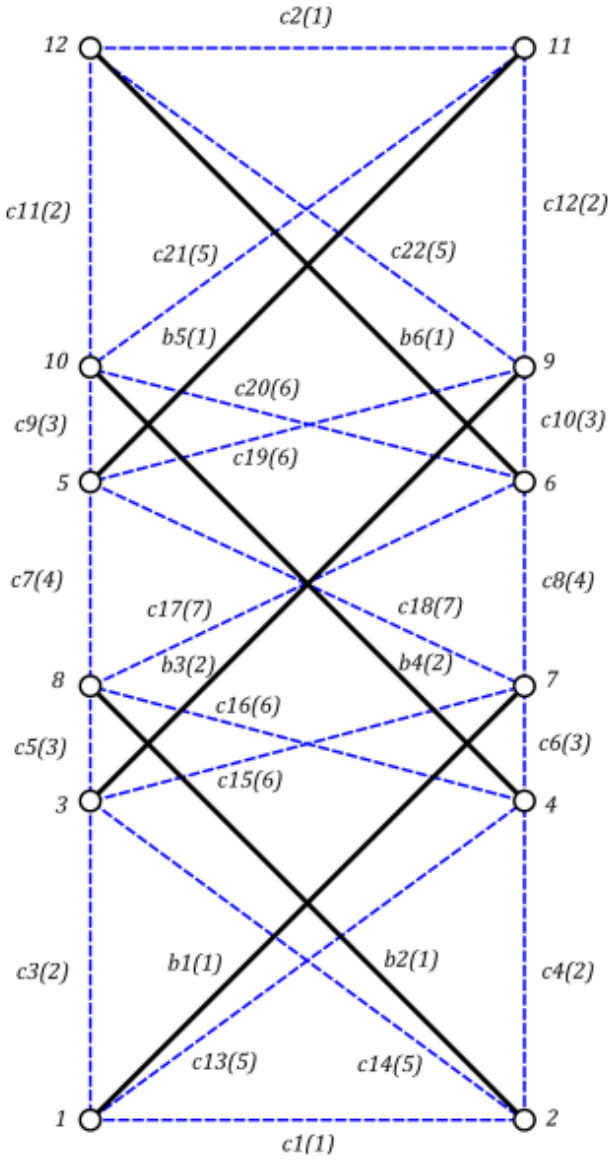
Zároveň opět došlo k posunu center chyb od míst křížení lokálních lineárních modelů, viz Obr. 50



Obr. 50 Procentuální chyba v kontextu dělení prostoru, $k_\sigma = 0.5$

8. Optimalizace manipulovatelnosti tensegritické struktury

Pro účely optimalizace byla zvolena 3-patrová rovinná tensegritická věž s diagonálním uložením lan. Matematický model této struktury a jeho realizaci v prostředí MATLAB vytvořil Ing. Aleš Balon ve své diplomové práci. [4]

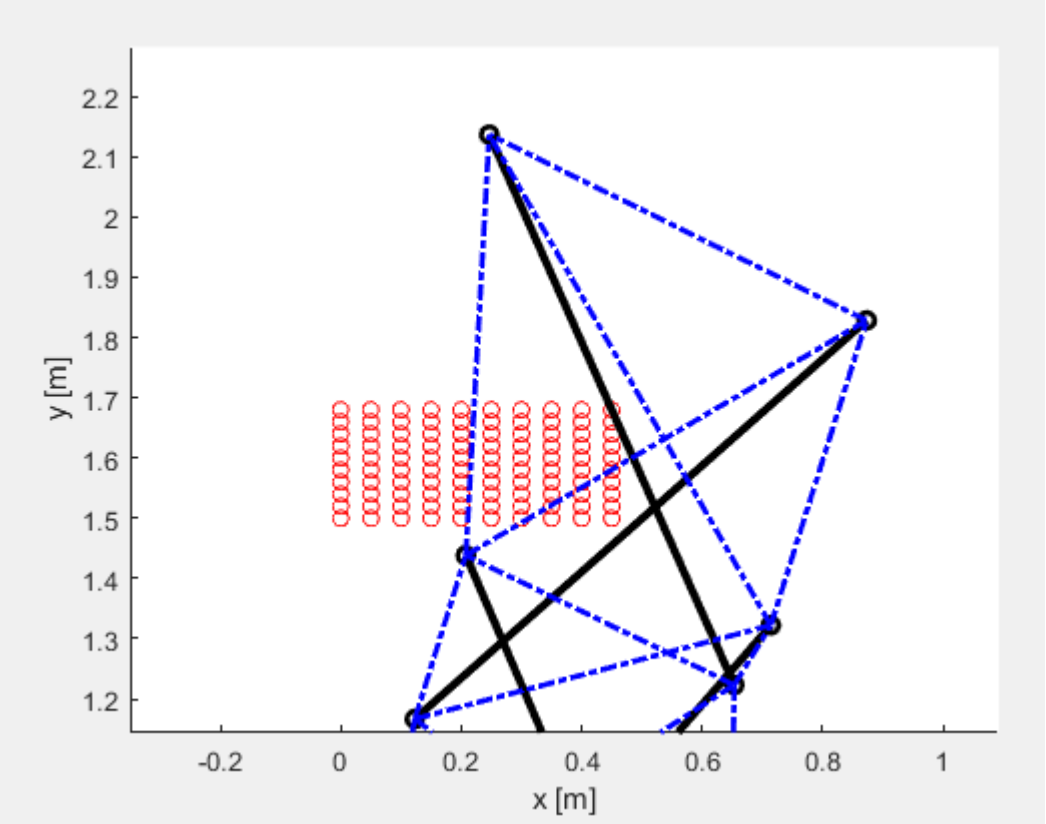


Obr. 51 tensegritická věž s diagonálním uložením lan [4]

Struktura se skládá z 6 tyčí a 22 lan proměnné délky. Celkově je tensegrita popsána 15 nezávislými souřadnicemi.

Jako koncový efektor byl zvoleno těžiště tyče b6(1). Pro tento bod se souřadnicemi x_{s13}, y_{s13} byl určen pracovní prostor. Ten je vidět na Obr. 52, jedná se o 100 poloh

označených červenými kolečky. Ostatních 13 nezávislých souřadnic bylo určeno jako optimalizované parametry.



Obr. 52 Pracovní prostor tensegrity

V každém bodě pracovního prostoru proběhla optimalizace 13 parametrů uvedených ve vektoru q

$$\vec{q} = [\varphi_1, x_2, \varphi_2, x_{s3}, y_{s3}, \varphi_3, x_{s4}, y_{s4}, \varphi_4, x_{s5}, y_{s5}, \varphi_5, \varphi_6]^T \tag{8.1}$$

Pro minimalizaci byla opět zvolena funkce *fmincon*, obsažená v prostředí MATLAB. Rozsah optimalizovaných parametrů byl zvolen jako

$$q_{i\_opt} \in \langle q_i - 0{,}5;\ q_i + 0{,}5 \rangle \tag{8.2}$$

Jako Jacobiho matice převodu byla využita modifikovaná matice relativních rychlostí. Její výpočet pomocí form-finding algoritmu je popsán v [4]. Pro naše účely bylo ovšem nutno tuto matici upravit do podoby

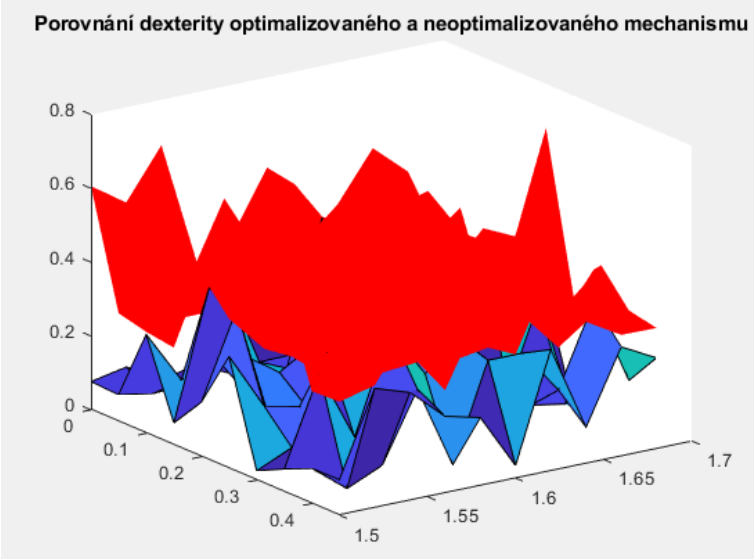
$$\boldsymbol{J} \cdot \dot{\vec{l}} = \begin{bmatrix} \dot{x}_{s13} \\ \dot{y}_{s13} \end{bmatrix} \tag{8.3}$$

Kde

$$\dot{\vec{l}} = \begin{bmatrix} \dot{l}_1 \\ \dot{l}_2 \\ \vdots \\ \dot{l}_{22} \end{bmatrix} \tag{8.4}$$

Toho je docíleno postupnou eliminací nezávislých proměnných ve vektoru nezávislých rychlostí $\dot{\vec{q}}$, kdy v posledním kroku eliminace zbydou na pravé straně pouze dvě proměnné, určující rychlost koncového efektoru. Matice $\boldsymbol{J}$ poté správně vyjadřuje převod mezi rychlostmi pohonů (lan) a rychlostí koncového efektoru. Problém minimalizace této matice tedy vede k výpočtu optimální manipulovatelnosti dle vzorce (6.1).

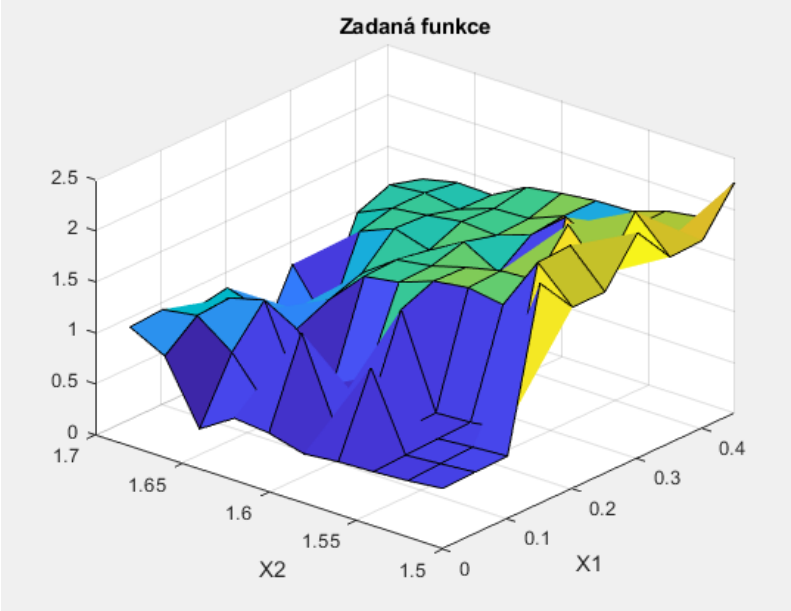
Porovnání manipulovatelnosti původního mechanismu a optimalizované struktury je na Obr. 53. Červená plocha odpovídá zlepšené konfiguraci.



Obr. 53 Porovnání manipulovatelnosti optimalizované a neoptimalizované tensegrity

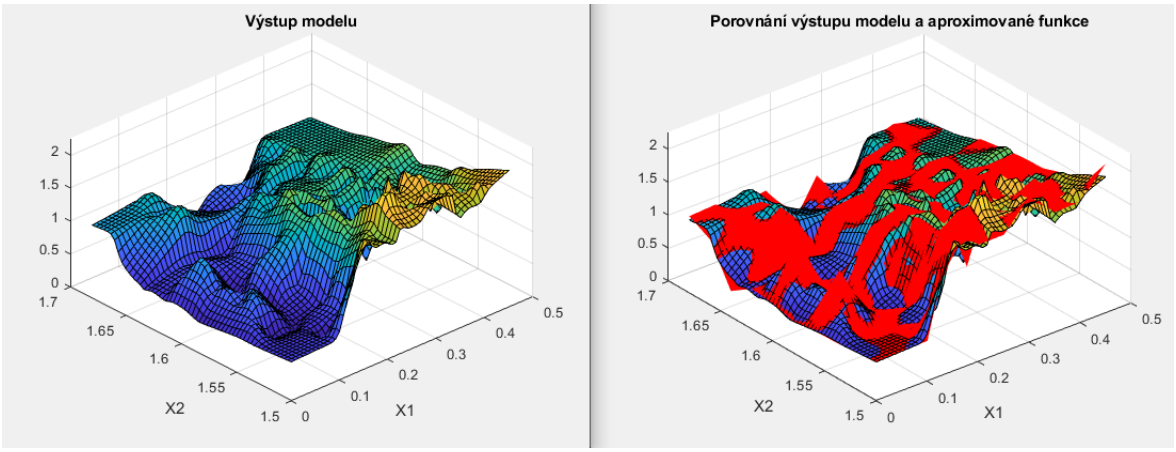
Následně byl vytvořen model pomocí algoritmu LOLIMOT. Jako vstupní prostor posloužily souřadnice pracovního prostoru. Jelikož LOLIMOT patří mezi MISO (multiple input - single output) systémy, byl vytvořen pro každé lano vlastní model. Vynášená funkce je tedy ideální délka příslušného lana. Pro ukázku použiji např. lano c10.

Závislost délky lana c10 na poloze v pracovním prostoru je na Obr. 54



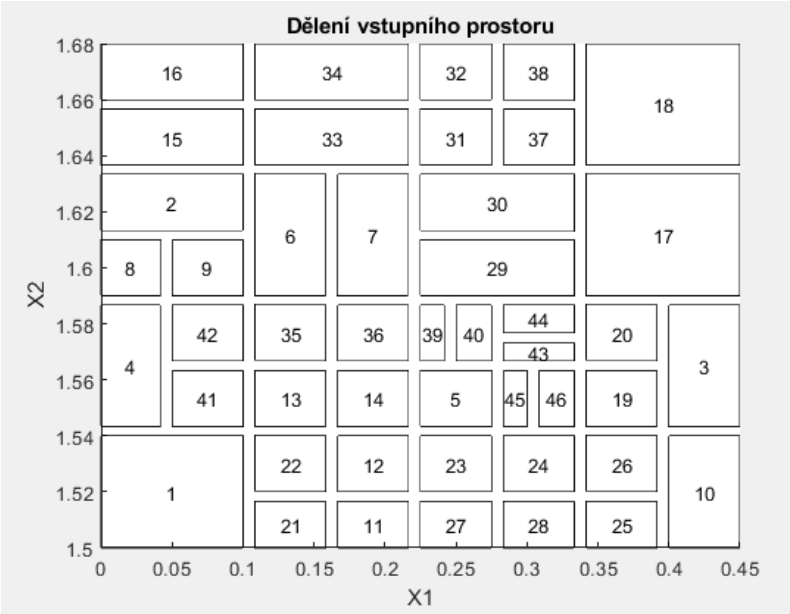
Obr. 54 Optimální délka lana c10 v pracovním prostoru

K aproximaci této funkce bylo využito 46 neuronů a hodnota ostrosti funkcí platnosti $k_\sigma = \frac{1}{3}$. V levé části Obr. 55 je vidět samotný výstup tohoto modelu a v pravé částí porovnání s původní funkcí.



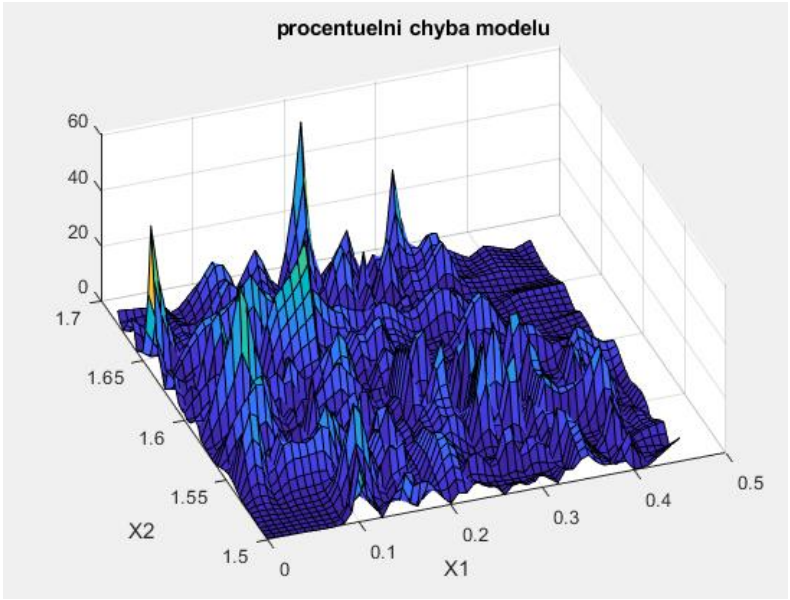
Obr. 55 Výstup modelu a porovnání se aproximovanou funkcí

Optimální dělení vstupního prostoru je na Obr. 56. Z obrázku je názorné, že čím větší nelinearita se nachází v určité oblasti, tím více lokálních modelů je určeno k lepší aproximaci.



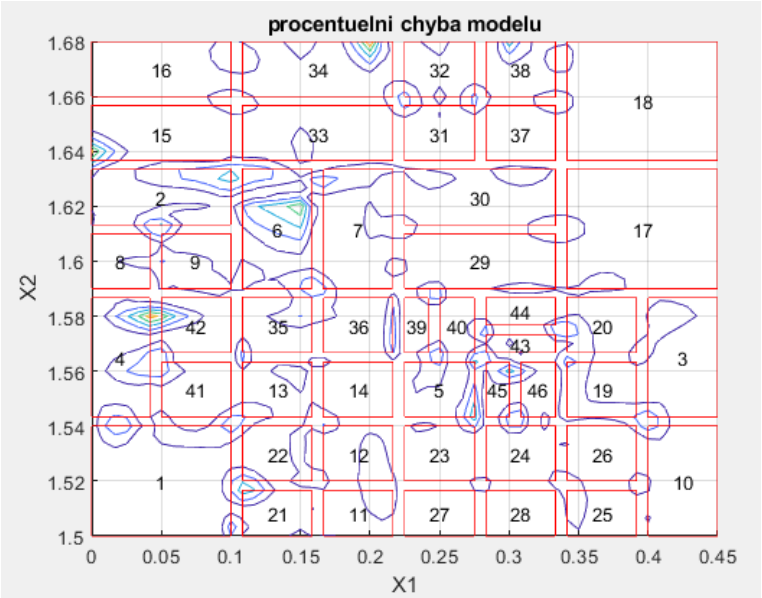
Obr. 56 Dělení vstupního prostoru

Procentuální rozdíl mezi výstupem modelu a měřené funkce je na Obr. 57. Vzhledem k vysokým nelinearitám je v určitých místech chyba značná. V průměru je ovšem odchylka dvou funkcí 0.0097 (viz *globalni_chyba_relativni* na Obr. 59), lze tedy celkovou přesnost modelu považovat za dostatečnou.



Obr. 57 Procentuální chyba modelu

Umístění chyby v kontextu dělení vstupního prostoru je pak na Obr. 58. Vzhledem ke zvolené hodnotě ostrosti funkcí platnosti je pak chyba koncentrována zejména v místech křížení lokálních lineárních modelů.



Obr. 58 Procentuální chyba v kontextu dělení prostoru, $k_\sigma = 0.33$

Celkový model je pak uložen v objektu model.

```
model =

  struct with fields:

                          w: {46×1 cell}
                         mi: {46×1 cell}
                         fi: {46×1 cell}
                          X: {1×46 cell}
                          Y: {1×46 cell}
    historie_absolutni_chyba: {1×45 cell}
                     vystup: [55×55 double]
                      chyba: [1×46 double]
                  max_chyba: 1.6074
              globalni_chyba: 29.4595
    globalni_chyba_relativni: 0.0097
             absolutni_chyba: [55×55 double]
          procentuelni_chyba: [55×55 double]
```

Obr. 59 Parametry modelu, $k_\sigma = 0.33$

Z parametru *historie_absolutni_chyba* je patrné postupné zpřesňování modelu při každém dělení a vzniku nového lokálního modelu. Konkrétně z hodnoty 161.7475 až na 29.4595. Zpřesnění je tedy téměř šestinásobné.

Na experimentálním příkladu se potvrdilo, že síť lokálních lineárních modelů je skutečně dobrý aproximátor i velice složitých nelineárních funkcí, kombinující výhodu nízkého výpočtového času. Naopak zásadní nevýhodou je řešení problému ve vyšších dimenzích [10]. V našem případě se ovšem prokázalo, že výše zmíněné výhody mnohonásobně převýší možné slabiny tohoto přístupu k tvorbě modelu.

9. Závěr

Obsahem práce bylo základní seznámení s problematikou tensegritických struktur, seznámení s identifikačními postupy pomocí sítě lokálních lineárních modelů a v neposlední řadě také optimalizace manipulovatelnosti.

V Kapitole 3 byl prozkoumán dosavadní vývoj a tensegritických struktur a zároveň byl odvozen základní návrhový výpočet, tzv. form-finding algoritmus, sloužící k nalezení rovnovážné předepjaté konfigurace.

V Kapitolách 4 a 5 jsem se seznámil se základy neuronových a neuro-fuzzy sítí, zejména pak s identifikačním algoritmem LOLIMOT. Tento algoritmus je důkladně popsán a je implementován v prostředí MATLAB. Implementace je popsána na zvoleném příkladu.

Kapitola 6 pojednává o manipulovatelnosti (dexteritě). Je vytvořen jednoduchý simulační příklad, na kterém je tento koncept demonstrován. Z výsledku simulace vyplývá, že nejhorší hodnota manipulovatelnosti se nachází na krajích pracovního prostoru. Jsou zde popsány i krajní případy, tzv. singularity. Konkrétně singularita prvého a druhého druhu.

V Kapitole 7 jsou znalosti nabyté z předchozích kapitol využity k optimalizaci pětikloubového mechanismu s redundantními pohony. Tento přístup simuluje tensegritický nůžkový mechanismus. Absolutní zlepšení manipulovatelnosti v levé části pracovního prostoru, kde se nachází proměnný člen, je až 0,4 [-].

Kapitola 8 obsahuje aplikaci veškerých znalostí na model konkrétní tensegritické struktury, je optimalizována její manipulovatelnost a vytvořen model pomocí neuro-fuzzy sítě. Zlepšení manipulovatelnosti je zde až o 0,6 [-] a odchylka modelu je dostatečně malá, konkrétní průměr chyby je 0.0097 v každém bodě pracovního prostoru.

Závěrem lze říci, že práce splnila všechny úkoly, které si v zadání definovala.

10. Reference

[1] R. Skelton a M. de Oliveira, Tensegrity Systems, San Diego: University of California, 2009.

[2] T. Kaňka, „Tensegritní mechanismy pro náhradu prostorových sériových robotů, Diplomová práce," CVUT, Praha, 2021.

[3] H. Yang, R. Liu, A. Luo, H. Liu a C. Li, „Force Density Relation and Lightweight Modeling of Single Layer Tensegrity Structures," J. Aerosp. Technol. Manag., 2020.

[4] A. Balon, „Optimalizace a řízení mechatronické tensegrity pro robotiku, Diplomová práce," CVUT, Praha, 2019.

[5] T. Skopec, „Kalibrace paralelních mechanismů s adaptivní složitostí modelu, Disertační práce," *CVUT,* 2012.

[6] F. Bre, G. J. M. a F. V. D., „Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks," *Energy and Buildings,* 11 2017.

[7] T. Wood, „deepai.org," 2018. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/sigmoid-function.

[8] S. Romanazzi, „BolognaNN - Photo Geolocation in Bologna with Convolutional Neural Networks," Universitz of Bologna, Bologna, 2018.

[9] E. Mamdani a S. Assilian, „An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller," *International Journal of Man-Machine Studies 7,* 1975.

[10] O. Nelles, „Nonlinear System Identification with Local Linear Neuro-Fuzzy Models, Ph.D. Thesis," Technische Universitat Darmstadt, 1998.

[11] V. Bushaev, „towardsdatascience.com," 27 November 2017. [Online]. Available: https://towardsdatascience.com/how-do-we-train-neural-networks-edd985562b73.

[12] L. MOHOLY-NAGY, Von material zu architektur, 1968.